Package 'idealstan'

December 6, 2018

Type Package

Title Generalized IRT Ideal Point Models with 'Stan'

Version 0.5.1

Date 2018-11-26

BugReports https://github.com/saudiwin/idealstan/issues

Description Offers item-response theory (IRT) ideal-point estimation for binary, ordinal, counts and continuous responses with time-varying and missing-data inference. Full and approximate Bayesian sampling with 'Stan' (www.mc-stan.org).

License GPL

Depends methods, R (>= 3.4.0), Rcpp (>= 0.12.18)

Imports rstan (>= 2.18.2), rstantools (>= 1.5.1), dplyr, tidyr, stringr, bayesplot, ggplot2, lazyeval, rlang, shinystan, gghighlight, forcats, ggrepel

Suggests pscl, loo, lubridate, R.rsp

LinkingTo StanHeaders (>= 2.18.0), rstan (>= 2.18.2), BH (>= 1.66), Rcpp (>= 0.12.7), RcppEigen (>= 0.3.2.9.0)

Encoding UTF-8

LazyData true

NeedsCompilation yes

SystemRequirements GNU make

VignetteBuilder R.rsp

RoxygenNote 6.1.0

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

Maintainer Robert Kubinec <rmk7@nyu.edu>

Repository CRAN

Date/Publication 2018-12-06 21:50:04 UTC

R topics documented:

derive_chain . . . 2
idealdata-class . . . 3
idealstan . . . 3
idealstan-class . . . 4
id_estimate . . . 4
id_extract . . . 10
id_extract,idealstan-method . . . 11
id_make . . . 11
id_plot . . . 14
id_plot,idealstan-method . . . 14
id_plot_all_hist . . . 15
id_plot_compare . . . 16
id_plot_cov . . . 16
id_plot_legis . . . 17
id_plot_legis_dyn . . . 19
id_plot_legis_var . . . 21
id_plot_ppc . . . 23
id_plot_ppc,idealstan-method . . . 23
id_plot_rhats . . . 24
id_plot_sims . . . 24
id_post_pred . . . 25
id_post_pred,idealstan-method . . . 25
id_sim_coverage . . . 26
id_sim_gen . . . 27
id_sim_resid . . . 28
id_sim_rmse . . . 28
launch_shinystan . . . 29
launch_shinystan,idealstan-method . . . 29
release_questions . . . 30
senate114 . . . 30
stan_trace . . . 31
stan_trace,idealstan-method . . . 31
summary,idealstan-method . . . 32

Index **33**

`derive_chain` *Helper Function for 'loo' calculation*

Description

This function accepts a log-likelihood matrix produced by 'id_post_pred' and extracts the IDs of the MCMC chains. It is necessary to use this function as the second argument to the 'loo' function along with an exponentiated log-likelihood matrix. See the package vignette How to Evaluate Models for more details.

Usage

```
derive_chain(ll_matrix = NULL)
```

Arguments

ll_matrix	A log-likelihood matrix as produced by the id_post_pred function

---

idealdata-class *Data and Identification for* id_estimate

Description

idealdata objects contain the relevant legislator/bill (person/item) matrix of data along with slots containing information about the kind of identification used in the estimation.

See Also

id_make to create an idealdata object suitable for estimation with id_estimate.

---

idealstan *idealstan package*

Description

R Interface to Stan for Item-Response Theory Ideal Point Models

Details

See the README on GitHub

References

1. Kubinec, Robert. Generalized Ideal Point Models for Time-Varying and Missing-Data Inference. Working Paper.
2. Clinton, J., Jackman, S., & Rivers, D. (2004). The Statistical Analysis of Roll Call Data. *The American Political Science Review*, 98(2), 355-370. doi:10.1017/S0003055404001194
3. Bafumi, J., Gelman, A., Park, D., & Kaplan, N. (2005). Practical Issues in Implementing and Understanding Bayesian Ideal Point Estimation. *Political Analysis*, 13(2), 171-187. doi:10.1093/pan/mpi010

idealstan-class *Results of* id_estimate *function*

Description

The idealstan objects store the results of estimations carried out by the id_estimate function. These objects include the full results of Bayesian sampling performed by the stan function in the **rstan** package.

id_estimate *Estimate an* idealstan *model*

Description

This function will take a pre-processed idealdata vote/score dataframe and run one of the available IRT/latent space ideal point models on the data using Stan's MCMC engine.

Usage

```
id_estimate(idealdata = NULL, model_type = 2, inflate_zero = FALSE,
  vary_ideal_pts = "none", use_subset = FALSE, sample_it = FALSE,
  subset_group = NULL, subset_person = NULL, sample_size = 20,
  nchains = 4, niters = 2000, use_vb = FALSE,
  restrict_ind_high = NULL, id_diff = 4, id_diff_high = 2,
  restrict_ind_low = NULL, fixtype = "vb_full", prior_fit = NULL,
  warmup = floor(niters/2), ncores = 4, use_groups = FALSE,
  discrim_reg_sd = 1, discrim_miss_sd = 1, person_sd = 1,
  time_sd = 0.1, sample_stationary = FALSE, ar_sd = 2,
  diff_reg_sd = 1, diff_miss_sd = 1, restrict_sd = 0.01,
  restrict_mean = NULL, restrict_var = NULL,
  restrict_mean_val = NULL, restrict_mean_ind = NULL,
  restrict_var_high = 0.1, ...)
```

Arguments

idealdata	An object produced by the id_make containing a score/vote matrix for use for estimation & plotting
model_type	An integer reflecting the kind of model to be estimated. See below.
inflate_zero	If the outcome is distributed as Poisson (count/unbounded integer), setting this to TRUE will fit a traditional zero-inflated model. To use correctly, the value for zero must be passed as the miss_val option to id_make before running a model so that zeroes are coded as missing data.
vary_ideal_pts	Default 'none'. If 'random_walk' or 'AR1', a time-varying ideal point model will be fit with either a random-walk process or an AR1 process. See documentation for more info.

use_subset — Whether a subset of the legislators/persons should be used instead of the full response matrix

sample_it — Whether or not to use a random subsample of the response matrix. Useful for testing.

subset_group — If person/legislative data was included in the id_make function, then you can subset by any value in the $group column of that data if use_subset is TRUE.

subset_person — A list of character values of names of persons/legislators to use to subset if use_subset is TRUE and person/legislative data was included in the id_make function with the required $person.names column

sample_size — If sample_it is TRUE, this value reflects how many legislators/persons will be sampled from the response matrix

nchains — The number of chains to use in Stan's sampler. Minimum is one. See stan for more info.

niters — The number of iterations to run Stan's sampler. Shouldn't be set much lower than 500. See stan for more info.

use_vb — Whether or not to use Stan's variational Bayesian inference engine instead of full Bayesian inference. Pros: it's much faster. Cons: it's not quite as accurate. See vb for more info.

restrict_ind_high — If fixtype is not "vb", the particular indices of legislators/persons or bills/items to constrain high

id_diff — The fixed difference between the high/low person/legislator ideal points used to identify the model. Set at 4 as a standard value but can be changed to any arbitrary number without affecting model results besides re-scaling.

id_diff_high — The fixed intercept of the high ideal point used to constrain the model.

restrict_ind_low — If fixtype is not "vb", the particular indices of legislators/persons or bills/items to constrain low. (Note: not used if values are pinned).

fixtype — Sets the particular kind of identification used on the model, could be one of 'vb_full' (identification provided exclusively by running a variational identification model with no prior info), 'vb_partial' (two indices of ideal points to fix are provided but the values to fix are determined by the identification model), 'constrain' (two indices of ideal points to fix are provided–only sufficient for model if restrict_var is FALSE, and 'prior_fit' (a previous identified idealstan fit is passed to the prior_fit option and used as the basis for identification). See details for more information.

prior_fit — If a previous idealstan model was fit *with the same* data, then the same identification constraints can be recycled from the prior fit if the idealstan object is passed to this option. Note that means that all identification options, like restrict_var, will also be the same

warmup — The number of iterations to use to calibrate Stan's sampler on a given model. Shouldn't be less than 100. See stan for more info.

ncores — The number of cores in your computer to use for parallel processing in the Stan engine. See stan for more info.

use_groups — If TRUE, group parameters from the person/legis data given in id_make will be estimated instead of individual parameters.

discrim_reg_sd — Set the prior standard deviation of the bimodal prior for the discrimination parameters for the non-inflated model.

discrim_miss_sd — Set the prior standard deviation of the bimodal prior for the discrimination parameters for the inflated model.

person_sd — Set the prior standard deviation for the legislators (persons) parameters

time_sd — The precision (inverse variance) of the over-time component of the person/legislator parameters. A higher value will allow for less over-time variation (useful if estimates bounce too much). Default is 4.

sample_stationary — If TRUE, the AR(1) coefficients in a time-varying model will be sampled from an unconstrained space and then mapped back to a stationary space. Leaving this TRUE is slower but will work better when there is limited information to identify a model. If used, the ar_sd parameter should be increased to 5 to allow for wider sampling in the unconstrained space.

ar_sd — If an AR(1) model is used, this defines the prior scale of the Normal distribution. A lower number can help identify the model when there are few time points.

diff_reg_sd — Set the prior standard deviation for the bill (item) intercepts for the non-inflated model.

diff_miss_sd — Set the prior standard deviation for the bill (item) intercepts for the inflated model.

restrict_sd — Set the prior standard deviation for constrained parameters

restrict_mean — Whether or not to restrict the over-time mean of an ideal point (additional identification measure when standard fixes don't work). TRUE by default for random-walk models.

restrict_var — Whether to limit variance to no higher than 0.5 for random-walk time series models. If left blank (the default), will be set to TRUE for random-walk models and FALSE for AR(1) models if identification is still a challenge (note: using this for AR(1) models is probably overkill).

restrict_mean_val — For random-walk models, the mean of a time-series ideal point to constrain. Should not be set a priori (leave blank) unless you are absolutely sure. Otherwise it is set by the identification model.

restrict_mean_ind — For random-walk models, the ID of the person/group whose over-time mean to constrain. Should be left blank (will be set by identification model) unless you are really sure.

restrict_var_high — The upper limit for the variance parameter (if restrict_var=TRUE & model is a random-walk time-series). If left blank, either defaults to 0.1 or is set by identification model.

... — Additional parameters passed on to Stan's sampling engine. See stan for more information.

Details

To run an IRT ideal point model, you must first pre-process your data using the `id_make` function. Be sure to specify the correct options for the kind of model you are going to run: if you want to run an unbounded outcome (i.e. Poisson or continuous), the data needs to be processed differently. Also any hierarchical covariates at the person or item level need to be specified in `id_make`. If they are specified in `id_make`, than all subsequent models fit by this function will have these covariates.

Note that for static ideal point models, the covariates are only defined for those persons who are not being used as constraints.

As of this version of `idealstan`, the following model types are available. Simply pass the number of the model in the list to the `model_type` option to fit the model.

1. IRT 2-PL (binary response) ideal point model, no missing-data inflation
2. IRT 2-PL ideal point model (binary response) with missing- inflation
3. Ordinal IRT (rating scale) ideal point model no missing-data inflation
4. Ordinal IRT (rating scale) ideal point model with missing-data inflation
5. Ordinal IRT (graded response) ideal point model no missing-data inflation
6. Ordinal IRT (graded response) ideal point model with missing-data inflation
7. Poisson IRT (Wordfish) ideal point model with no missing data inflation
8. Poisson IRT (Wordfish) ideal point model with missing-data inflation
9. unbounded (Gaussian) IRT ideal point model with no missing data
10. unbounded (Gaussian) IRT ideal point model with missing-data inflation
11. Positive-unbounded (Log-normal) IRT ideal point model with no missing data
12. Positive-unbounded (Log-normal) IRT ideal point model with missing-data inflation
13. Latent Space (binary response) ideal point model with no missing data
14. Latent Space (binary response) ideal point model with missing-data inflation

In addition, each of these models can have time-varying ideal point (person) parameters if a column of dates is fed to the `id_make` function. If the option `vary_ideal_pts` is set to `'random_walk'`, `id_estimate` will estimate a random-walk ideal point model where ideal points move in a random direction. If `vary_ideal_pts` is set to `'AR1'`, a stationary ideal point model is estimated where ideal points fluctuate around long-term mean. In general, the stationary model is preferred when the time series is of short absolute duration (such as days or hours) while the random-walk model is preferable when the time series is of very long duration and there are no natural limits to the ideal points. Please see the package vignette and associated paper for more detail about these time-varying models.

The inflation model used to account for missing data assumes that missingness is a function of the persons' (legislators') ideal points. In other words,the model will take into account if people with high or low ideal points tend to have more/less missing data on a specific item/bill. Missing data is whatever was passed as `miss_val` to the `id_make` function. If there isn't any relationship between missing data and ideal points, then the model assumes that the missingness is ignorable conditional on each item, but it will still adjust the results to reflect these ignorable (random) missing values. The inflation is designed to be general enough to handle a wide array of potential situations where strategic social choices make missing data important to take into account.

The missing data is assumed to be any possible value of the outcome. The well-known zero-inflated Poisson model is a special case where missing values are known to be all zeroes. To fit a zero-inflated Poisson model, change `inflate_zeroes` to `TRUE` and also make sure to set the value for zero as `miss_val` in the `id_make` function. This will only work for outcomes that are distributed as Poisson variables (i.e., unbounded integers or counts).

To leave missing data out of the model, simply choose a version of the model in the list above that is non-inflated.

Models can be either fit on the person/legislator IDs or on group-level IDs (as specified to the `id_make` function). If group-level parameters should be fit, set `use_groups` to `TRUE`.

Value

A fitted `idealstan` object that contains posterior samples of all parameters either via full Bayesian inference or a variational approximation if `use_vb` is set to `TRUE`. This object can then be passed to the plotting functions for further analysis.

Identification

Identifying IRT models is challenging, and ideal point models are still more challenging because the discrimination parameters are not constrained. As a result, more care must be taken to obtain estimates that are the same regardless of starting values. The parameter `fixtype` enables you to change the type of identification used. The default, 'vb_full', does not require any further information from you in order for the model to be fit. In this version of identification, an unidentified model is run using variational Bayesian inference (see `vb`). The function will then select two persons/legislators that end up on either end of the ideal point spectrum, and pin their ideal points to those specific values. This is sufficient to identify all of the static models and also the AR(1) time-varying models. For random-walk time-varying models, identification is more difficult (see vignette). Setting the option `restrict_mean` to `TRUE` will implement additional identification constraints on random-walk models. A particularly convenient option for `fixtype` is `'vb_partial'`. In this case, the user should pass the IDs (as a character vector) of the persons to constrain high (`restrict_ind_high`) and low (`restrict_ind_low`). A model will then be fit to find the likely positions of these parameters, which will then be used to fit an identified model. In this way, the user can achieve a certain shape of the ideal point distribution without needing to choose specific values ahead of time to pin parameters to. If a prior model has been estimated with the same data, the user can re-use those identification settings by passing the fitted `idealstan` object to the `prior_fit` option.

References

1. Clinton, J., Jackman, S., & Rivers, D. (2004). The Statistical Analysis of Roll Call Data. *The American Political Science Review*, 98(2), 355-370. doi:10.1017/S0003055404001194

2. Bafumi, J., Gelman, A., Park, D., & Kaplan, N. (2005). Practical Issues in Implementing and Understanding Bayesian Ideal Point Estimation. *Political Analysis*, 13(2), 171-187. doi:10.1093/pan/mpi010

3. Kubinec, R. "Generalized Ideal Point Models for Time-Varying and Missing-Data Inference". Working Paper.

See Also

id_make for pre-processing data, id_plot_legis for plotting results, summary for obtaining posterior quantiles, posterior_predict for producing predictive replications.

Examples

```
# First we can simulate data for an IRT 2-PL model that is inflated for missing data
library(ggplot2)
library(dplyr)

# This code will take at least a few minutes to run
## Not run:
bin_irt_2pl_abs_sim <- id_sim_gen(model_type='binary',inflate=T)

# Now we can put that directly into the id_estimate function
# to get full Bayesian posterior estimates
# We will constrain discrimination parameters
# for identification purposes based on the true simulated values

bin_irt_2pl_abs_est <- id_estimate(bin_irt_2pl_abs_sim,
                       model_type=2,
                       restrict_ind_high =
                       sort(bin_irt_2pl_abs_sim@simul_data$true_person,
                       decreasing=TRUE,
                       index=TRUE)$ix[1],
                       restrict_ind_low =
                       sort(bin_irt_2pl_abs_sim@simul_data$true_person
                       decreasing=FALSE,
                       index=TRUE)$ix[1],
                       fixtype='vb_partial',
                       ncores=2,
                       nchains=2)

# We can now see how well the model recovered the true parameters

id_sim_coverage(bin_irt_2pl_abs_est) %>%
         bind_rows(.id='Parameter') %>%
         ggplot(aes(y=avg,x=Parameter)) +
           stat_summary(fun.args=list(mult=1.96)) +
           theme_minimal()

## End(Not run)

# In most cases, we will use pre-existing data
# and we will need to use the id_make function first
# We will use the full rollcall voting data
# from the 114th Senate as a rollcall object

data('senate114')

# Running this model will take at least a few minutes, even with
# variational inference (use_vb=T) turned on
```

```
## Not run:

to_idealstan <-   id_make(score_data = senate114,
outcome = 'cast_code',
person_id = 'bioname',
item_id = 'rollnumber',
group_id= 'party_code',
time_id='date',
high_val='Yes',
low_val='No',
miss_val='Absent')

sen_est <- id_estimate(senate_data,
model_type = 2,
use_vb = TRUE,
fixtype='vb_partial',
restrict_ind_high = "BARRASSO, John A.",
restrict_ind_low = "WARREN, Elizabeth")

# After running the model, we can plot
# the results of the person/legislator ideal points

id_plot_legis(sen_est)

## End(Not run)
```

`id_extract` *Generic Method for Extracting Posterior Samples*

Description

This is a generic function.

Usage

```
id_extract(object, ...)
```

Arguments

`object`	A fitted `idealstan` object
`...`	Other arguments passed on to underlying functions

id_extract,idealstan-method *Extract stan joint posterior distribution from idealstan object*

Description

This convenience function allows you to extract the underlying rstan posterior estimates for the full parameters estimates of the idealstan model object. See extract for the underlying function and more options.

You can use this function to access a matrix or array of the full posterior estimates of each of the parameters in an idealstan object. There are available options to pick certain parameters of the model, such as the person (legislator) ideal points or item (bill) discrimination scores. Alternatively, you can leave the extract_type option blank and receive a list of all of the available parameters. Please note that the list of parameters do not have particularly informative names.

All parameters are returned in the order in which they were input into the id_make function.

Usage

```
## S4 method for signature 'idealstan'
id_extract(object, extract_type = "persons", ...)
```

Arguments

object	A fitted idealstan object (see id_estimate)
extract_type	Can be one of 'persons' for person/legislator ideal points, 'reg_discrim' for non-inflated item (bill) discrimination scores, 'reg_diff' for non-inflated item (bill) difficulty scores, 'miss_discrim' for inflated item (bill) discrimination scores, and 'miss_diff' for inflated item (bill) difficulty scores.
...	Any additional arguments passed on to the extract function.

id_make *Create data to run IRT model*

Description

To run an IRT model using idealstan, you must first process your data using the id_make function.

Usage

```
id_make(score_data = NULL, outcome = "outcome",
  person_id = "person_id", item_id = "item_id", time_id = "time_id",
  group_id = "group_id", simul_data = NULL, person_cov = NULL,
  group_cov = NULL, item_cov = NULL, item_cov_miss = NULL,
  miss_val = NA, high_val = NULL, low_val = NULL,
  middle_val = NULL, unbounded = FALSE, exclude_level = NA,
  simulation = FALSE)
```

Arguments

score_data	A data frame in long form, i.e., one row in the data for each measured score or vote in the data or a rollcall data object from package pscl.
outcome	Column name of the outcome in score_data, default is "outcome"
person_id	Column name of the person/legislator ID index in score_data, default is 'person_id'. Should be integer, character or factor.
item_id	Column name of the item/bill ID index in score_data, default is 'item_id'. Should be integer, character or factor.
time_id	Column name of the time values in score_data: optional, default is 'time_id'. Should be a date or date-time class, but can be an integer (i.e., years in whole numbers).
group_id	Optional column name of a person/legislator group IDs (i.e., parties) in score_data. Optional, default is 'group_id'. Should be integer, character or factor.
simul_data	Optionally, data that has been generated by the id_sim_gen function.
person_cov	A one-sided formula that specifies the covariates in score_data that will be used to hierarchically model the person/legislator ideal points
group_cov	A one-sided formula that specifies the covariates in score_data that will be used to hierarchically model the person/legislator ideal points at the group level. Use this in place of person_cov if you intend to run a group-level model.
item_cov	A one-sided formula that specifies the covariates in score_data that will be used to hierarchically model the item/bill discrimination parameters for the regular model
item_cov_miss	A one-sided formula that specifies the covariates in the dataset that will be used to hierarchically model the item/bill discrimination parameters for the missing data model.
miss_val	The value (numeric or character) that indicate missing data/absences in the data. If missing data is coded as NA, simply leave this parameter at the default, NA.
high_val	The value (numeric or character) that indicate the highest discrete outcome possible, such as yes in a vote dataset or correct in a test examination.
low_val	The value (numeric or character) that indicates the lowest discrete outcome possible, such as no votes in a vote dataset or incorrect in a test examination.
middle_val	The value (numeric or character) that indicate values between the lowest and highest categories, such as abstention in voting data or "Neither Agree nor Disagree" in Likert scales. If there are multiple possible values, pass along a numeric or character vector of all such values in correct order (lower to higher values). If there are no middle values (binary outcome), leave empty.
unbounded	Whether or not the outcome/response is unbounded (i.e., continuous or Poisson). If it is, miss_val is recoded as the maximum of the outcome + 1.
exclude_level	A vector of any values that should be treated as NA in the response matrix. Unlike the miss_val parameter, these values will be dropped from the data before estimation rather than modeled explicitly.
simulation	If TRUE, simulated values are saved in the idealdata object for later plotting with the id_plot_sims function

Details

This function can accept either a `rollcall` data object from package `pscl` or a long data frame where one row equals one item-person (bill-legislator) observation with associated outcome. The preferred method is the long data frame as passing a long data frame permits the inclusion of a wide range of covariates in the model, such as person-varying and item-varying (bill-varying) covariates. If a `rollcall` object is passed to the function, the `rollcall` data is converted to a long data frame with data from the `vote.data` matrix used to determine dates for bills. If passing a long data frame, you should specify the names of the columns containing the IDs for persons, items and groups (groups are IDs that may have multiple observations per ID, such as political parties or classes) to the `id_make` function, along with the name of the response/outcome. The only required columns are the item/bill ID and the person/legislator ID along with an outcome column.

The preferred format for the outcome column for discrete variables (binary or ordinal) is to pass a factor variable with levels in the correct order, i.e., in ascending order. For example, if using legislative data, the levels of the factor should be `c('No','Yes')`. If a different kind of variable is passed, such as a character or numeric variable, you should consider specifying `low_val`,`high_val` and `middle_val` to determine the correct order of the discrete outcome. Specifying `middle_val` is only necessary if you are estimating an ordinal model.

If you do not specify a value for `miss_val`, then any `NA` are assumed to be missing. If you do specify `miss_val` and you also have `NA` in your data (assuming `miss_val` is not `NA`), then the function will treat the data coded as `miss_val` as missing data that should be modeled and will treat the `NA` data as ignorable missing data that will be removed (list-wise deletion) before estimating a model.

To run a time-varying model, you need to include the name of a column with dates (or integers) that is passed to the `time_id` option.

If the outcome is unbounded i.e. a continuous or an unbounded discrete variable like Poisson, simply set `unbounded` to `TRUE`. You can ignore the options that specify which values should be `high_val` or `low_val`. You can either specify a particular value as missing using `miss_val`, or all missing values (`NA`) will be recoded to a specific value out of the range of the outcome to use for modeling the missingness.

Value

A `idealdata` object that can then be used in the `id_estimate` function to fit a model.

Examples

```
# You can either use a pscl rollcall object or a vote/score matrix
# where persons/legislators are in the rows
# and items/bills are in the columns

library(dplyr)

# First, using a rollcall object with the 114th Senate's rollcall votes:

data('senate114')

to_idealstan <-   id_make(score_data = senate114,
                outcome = 'cast_code',
                person_id = 'bioname',
```

```
                item_id = 'rollnumber',
                group_id= 'party_code',
                time_id='date',
                high_val='Yes',
                low_val='No',
                miss_val='Absent')
```

`id_plot` *Generic Function for Plotting* `idealstan` *objects*

Description

This generic function will run all the plotting functions associated with fitted `idealstan` objects.

Usage

```
id_plot(object, ...)
```

Arguments

`object`	An `idealstan` object
`...`	Other options passed onto the underlying plot function

`id_plot,idealstan-method`
Plot Results of `id_estimate`

Description

This function allows you to access the full range of plotting options for fitted `idealstan` models.

Usage

```
## S4 method for signature 'idealstan'
id_plot(object, plot_type = "persons", ...)
```

Arguments

`object`	A fitted `idealstan` object
`plot_type`	Specify the plot as a character string. Currently 'persons' for legislator/person ideal point plot and 'histogram' for a histogram of model estimates for given parameters.
`...`	Additional arguments passed on to the underlying functions. See individual function documentation for details.

Details

id_plot is a wrapper function that can access the various plotting functions available in the idealstan package. Currently, the options are limited to a plot of legislator/person ideal points with bills/item midpoints as an optional overlay. Additional plots will be available in future versions of idealstan.

Value

A ggplot object

See Also

id_plot_legis for a legislator/person ideal point plot, id_plot_all_hist for a standard histogram plot, id_plot_compare for an ideal point plot of two different models of the same data, id_plot_rhats for a histogram of Rhat values, id_plot_sims for plotting true versus estimated values, id_estimate for how to estimate an idealstan object.

---

id_plot_all_hist *Density plots of Posterior Parameters*

Description

This function produces density plots of the different types of parameters in an idealstan model: item (bill) difficulty and discrimination parameters, and person (legislator) ideal points.

Usage

```
id_plot_all_hist(object, params = "person", param_labels = NULL,
  dens_type = "all", return_data = FALSE, func = median, ...)
```

Arguments

object	A fitted idealstan object
params	Select the type of parameter from the model to plot. 'person' for person/legislator ideal points, 'miss_diff' and 'miss_discrim' for difficulty and discrimination parameters from the missing/inflated item/bill parameters, and 'regular_diff' and 'regular_discrim' for difficulty and discrimination parameters from the non-missing/non-inflated item/bill parameters.
param_labels	A vector of labels equal to the number of parameters. Primarily useful if return_data is TRUE.
dens_type	Can be 'all' for showing 90 Or to show one of those posterior estimates at a time, use 'high' for 90 'low' for 10 in func (median by default).
return_data	Whether or not to return the plot as a ggplot2 object and the data together in a list instead of plotting.
func	The function to use if 'dens_type' is set to 'function'.
...	Other options passed on to the plotting function, currently ignored.

id_plot_compare *Function to compare two fitted idealstan models by plotting ideal points. Assumes that underlying data is the same for both models.*

Description

Function to compare two fitted idealstan models by plotting ideal points. Assumes that underlying data is the same for both models.

Usage

```
id_plot_compare(model1 = NULL, model2 = NULL, scale_flip = FALSE,
  return_data = FALSE, labels = NULL, hjust = -0.1,
  palette = "Set1", color_direction = 1, text_size_label = 2,
  rescale = FALSE)
```

Arguments

model1	The first model to compare
model2	The second model to compare
scale_flip	This parameter is set to true if you have two models that are reflected around the ideal point axis. This can happen as a result of identification and is harmless.
return_data	Whether to return the underlying data
labels	TRUE or FALSE, whether to use labels for points
hjust	The horizontal adjustment of point labels
palette	colorbrewer palette name
color_direction	Whether to reverse the color scale
text_size_label	Size of point labels
rescale	Whether to rescale the estimates from two models so they will match regardless of arbitrary scale shifts in the ideal points

id_plot_cov *Display Coefficient Plot of Hierarchical Covariates*

Description

This function will pull the estimates of the hierarchical covariates (whether at the person or item-discrimination level) and then plot them on a vertical coefficient plot. Names of the parameters are taken from the levels of the factor if a categorical variable and the column names otherwise.

Usage

```
id_plot_cov(object, cov_type = "person_cov", filter_cov = NULL, ...)
```

Arguments

object	A fitted idealstan object
cov_type	Either 'person_cov' for person-level hierarchical parameters, 'discrim_reg_cov' for bill/item discrimination parameters from regular (non-inflated) model, and 'discrim_infl_cov' for bill/item discrimination parameters from inflated model.
filter_cov	A character vector of coefficients from covariate plots to exclude from plotting (should be the names of coefficients as they appear in the plots)
...	Any additional parameters passed on to mcmc_intervals

Value

A ggplot2 plot that can be further customized with ggplot2 functions if need be.

---

id_plot_legis *Plot Legislator/Person and Bill/Item Ideal Points*

Description

This function can be used on a fitted idealstan object to plot the relative positions and uncertainties of legislator/persons and bills/items.

Usage

```
id_plot_legis(object, return_data = FALSE, include = NULL,
  high_limit = 0.95, low_limit = 0.05, item_plot = NULL,
  item_plot_type = "non-inflated", text_size_label = 2,
  text_size_group = 2.5, point_size = 1, hjust_length = -0.7,
  person_labels = TRUE, group_labels = F, person_ci_alpha = 0.2,
  show_true = FALSE, group_color = TRUE, hpd_limit = 10,
  sample_persons = NULL, ...)
```

Arguments

object	A fitted idealstan object
return_data	If true, the calculated legislator/bill data is returned along with the plot in a list
include	Specify a list of person/legislator IDs to include in the plot (all others excluded)
high_limit	The quantile (number between 0 and 1) for the high end of posterior uncertainty to show in plot
low_limit	The quantile (number between 0 and 1) for the low end of posterior uncertainty to show in plot
item_plot	The IDs (character vector) of the bill/item midpoints to overlay on the plot

item_plot_type	Whether to show the 'non-inflated' item/bill midpoints, the 'inflated' item/bill midpoints, or produce plots for 'both' kinds of models. Defaults to 'non-inflated' and will only display an item/bill midpoint if one has been specified in item_plot.
text_size_label	ggplot2 text size for legislator labels
text_size_group	ggplot2 text size for group text used for points
point_size	If person_labels and group_labels are set to FALSE, controls the size of the points plotted.
hjust_length	horizontal adjustment of the legislator labels
person_labels	if TRUE, use the person_id column to plot labels for the person (legislator) ideal points
group_labels	if TRUE, use the group column to plot text markers for the group (parties) from the person/legislator data
person_ci_alpha	The transparency level of the dot plot and confidence bars for the person ideal points
show_true	Whether to show the true values of the legislators (if model has been simulated)
group_color	If TRUE, give each group/bloc a different color
hpd_limit	The greatest absolute difference in high-posterior density interval shown for any point. Useful for excluding imprecisely estimated persons/legislators from the plot. Leave NULL if you don't want to exclude any.
sample_persons	If you don't want to use the full number of persons/legislators from the model, enter a proportion (between 0 and 1) to select only a fraction of the persons/legislators.
...	Other options passed on to plotting function, currently ignored

Details

This plot shows the distribution of ideal points for the legislators/persons in the model. It will plot them as a vertical dot plot with associated high-density posterior interval (can be changed with high_limit and low_limit options). In addition, if item/bill IDs as a character vector is passed to the item_plot option, then an item/bill midpoint will be overlain on the ideal point plot, showing the point at which legislators/persons are indifferent to voting/answering on the bill/item. Note that because this is an ideal point model, it is not possible to tell from the midpoint itself which side will be voting which way. For that reason, the legislators/persons are colored by their votes/scores to make it clear.

Examples

```
## Not run:

# First create data and run a model

to_idealstan <-   id_make(score_data = senate114,
```

```
outcome = 'cast_code',
person_id = 'bioname',
item_id = 'rollnumber',
group_id= 'party_code',
time_id='date',
high_val='Yes',
low_val='No',
miss_val='Absent')

sen_est <- id_estimate(senate_data,
model_type = 2,
use_vb = TRUE,
fixtype='vb_partial',
restrict_ind_high = "BARRASSO, John A.",
restrict_ind_low = "WARREN, Elizabeth")

# After running the model, we can plot
# the results of the person/legislator ideal points

id_plot_legis(sen_est)

## End(Not run)
```

`id_plot_legis_dyn` *Function to plot dynamic ideal point models*

Description

This function can be used on a fitted `idealstan` object to plot the relative positions and uncertainties of legislator/persons and bills/items when the legislator/person ideal points are allowed to vary over time.

Usage

```
id_plot_legis_dyn(object, return_data = FALSE, include = NULL,
  item_plot = NULL, text_size_label = 2, text_size_group = 2.5,
  high_limit = 0.95, low_limit = 0.05, line_size = 1,
  highlight = NULL, plot_text = TRUE, use_ci = TRUE,
  person_line_alpha = 0.3, person_ci_alpha = 0.8,
  item_plot_type = "non-inflated", show_true = FALSE,
  group_color = TRUE, hpd_limit = 10, sample_persons = NULL,
  plot_sim = FALSE, ...)
```

Arguments

`object`	A fitted `idealstan` object
`return_data`	If true, the calculated legislator/bill data is returned along with the plot in a list

include	Specify a list of person/legislator IDs to include in the plot (all others excluded)
item_plot	The value of the item/bill for which to plot its midpoint (character value)
text_size_label	ggplot2 text size for legislator labels
text_size_group	ggplot2 text size for group text used for points
high_limit	A number between 0 and 1 showing the upper limit to compute the posterior uncertainty interval (defaults to 0.95).
low_limit	A number between 0 and 1 showing the lower limit to compute the posterior uncertainty interval (defaults to 0.05).
line_size	Sets the size of the line of the time-varying ideal points.
highlight	A character referring to one of the persons in person_labels that the plot can highlight relative to other persons
plot_text	If TRUE, will plot person_labels over the lines.
use_ci	Whether or not high-posterior density intervals (credible intervals) should be plotted over the estimates (turn off if the plot is too busy)
person_line_alpha	The transparency level of the time-varying ideal point line
person_ci_alpha	The transparency level of ribbon confidence interval around the time-varying ideal points
item_plot_type	Whether to show the 'non-inflated' item/bill midpoints, the 'inflated' item/bill midpoints, or produce plots for 'both' kinds of models. Defaults to 'non-inflated' and will only display an item/bill midpoint if one has been specified in item_plot.
show_true	Whether to show the true values of the legislators (if model has been simulated)
group_color	If TRUE, use the groups instead of individuals to plot colours
hpd_limit	The greatest absolute difference in high-posterior density interval shown for any point. Useful for excluding imprecisely estimated persons/legislators from the plot. Leave NULL if you don't want to exclude any.
sample_persons	If you don't want to use the full number of persons/legislators from the model, enter a proportion (between 0 and 1) to select only a fraction of the persons/legislators.
plot_sim	Whether to plot the true values of parameters if a simulation was used to generate data (see id_sim_gen)
...	Other options passed on to plotting function, currently ignored

Details

This plot shows the distribution of ideal points for the legislators/persons in the model, and also traces the path of these ideal points over time. It will plot them as a vertical line with associated high-density posterior interval (10% to 90%). In addition, if the column index for a bill/item from the response matrix is passed to the item_plot option, then an item/bill midpoint will be overlain on the ideal point plot, showing the point at which legislators/persons are indifferent to voting/answering on the bill/item. Note that because this is an ideal point model, it is not possible to tell from the midpoint itself which side will be voting which way. For that reason, the legislators/persons are colored by their votes/scores to make it clear.

Examples

```
## Not run:

# First create data and run a model

to_idealstan <-   id_make(score_data = senate114,
outcome = 'cast_code',
person_id = 'bioname',
item_id = 'rollnumber',
group_id= 'party_code',
time_id='date',
high_val='Yes',
low_val='No',
miss_val='Absent')

sen_est <- id_estimate(senate_data,
model_type = 2,
use_vb = TRUE,
vary_ideal_pts='random_walk',
fixtype='vb_partial',
restrict_ind_high = "BARRASSO, John A.",
restrict_ind_low = "WARREN, Elizabeth")

# After running the model, we can plot
# the results of the person/legislator ideal points

id_plot_legis_dyn(sen_est)

## End(Not run)
```

---

id_plot_legis_var *Plot Legislator/Person Over-time Variances*

Description

This function can be used on a fitted idealstan object to plot the over-time variances (average rates of change in ideal points) for all the persons/legislators in the model.

Usage

```
id_plot_legis_var(object, return_data = FALSE, include = NULL,
  high_limit = 0.95, low_limit = 0.05, text_size_label = 2,
  text_size_group = 2.5, point_size = 1, hjust_length = -0.7,
  person_labels = TRUE, group_labels = F, person_ci_alpha = 0.1,
  group_color = TRUE, ...)
```

Arguments

object	A fitted idealstan object
return_data	If true, the calculated legislator/bill data is returned along with the plot in a list
include	Specify a list of person/legislator IDs to include in the plot (all others excluded)
high_limit	The quantile (number between 0 and 1) for the high end of posterior uncertainty to show in plot
low_limit	The quantile (number between 0 and 1) for the low end of posterior uncertainty to show in plot
text_size_label	ggplot2 text size for legislator labels
text_size_group	ggplot2 text size for group text used for points
point_size	If person_labels and group_labels are set to FALSE, controls the size of the points plotted.
hjust_length	horizontal adjustment of the legislator labels
person_labels	if TRUE, use the person_id column to plot labels for the person (legislator) ideal points
group_labels	if TRUE, use the group column to plot text markers for the group (parties) from the person/legislator data
person_ci_alpha	The transparency level of the dot plot and confidence bars for the person ideal points
group_color	If TRUE, give each group/bloc a different color
...	Other options passed on to plotting function, currently ignored

Details

This function will plot the person/legislator over-time variances as a vertical dot plot with associated high-density posterior interval (can be changed with high_limit and low_limit options).

Examples

```
# To demonstrate, we load the 114th Senate data and fit a time-varying model

data('senate114_fit')

## Not run:
senate_data <- id_make(senate114,outcome = 'cast_code',
person_id = 'bioname',
item_id = 'rollnumber',
group_id= 'party_code',
time_id='date',
miss_val='Absent')

 senate114_time_fit <- id_estimate(senate_data,
```

```
 model_type = 2,
 use_vb = T,
 fixtype='vb_partial',
 vary_ideal_pts='random_walk',
 restrict_ind_high = "WARREN, Elizabeth",
 restrict_ind_low="BARRASSO, John A.",
 seed=84520)
# We plot the variances for all the Senators

id_plot_legis_var(senate114_fit,item_plot=5)

## End(Not run)
```

`id_plot_ppc` *Plot Posterior Predictive Distribution for* `idealstan` *Objects*

Description

This function is the generic method for generating posterior distributions from a fitted `idealstan` model. Functions are documented in the actual method.

Usage

```
id_plot_ppc(object, ...)
```

Arguments

`object`	A fitted `idealstan` object
`...`	Other arguments passed on to `ppc_bars`

`id_plot_ppc,idealstan-method`
Plot Posterior Predictive Distribution for `idealstan` *Objects*

Description

This function is the actual method for generating posterior distributions from a fitted `idealstan` model.

Usage

```
## S4 method for signature 'idealstan'
id_plot_ppc(object, ppc_pred = NULL,
  group = NULL, item = NULL, ...)
```

Arguments

object	A fitted idealstan object
ppc_pred	The output of the id_post_pred function on a fitted idealstan object
group	A character vector of the person or group IDs over which to subset the predictive distribution
item	A character vector of item IDs to subset the posterior distribution
...	Other arguments passed on to ppc_bars

---

id_plot_rhats *Plotting Function to Display Rhat Distribution*

Description

This plotting function displays a histogram of the Rhat values of all parameters in an idealstan model.

Usage

```
id_plot_rhats(obj)
```

Arguments

obj	A fitted idealstan object.

---

id_plot_sims *This function plots the results from a simulation generated by id_sim_gen.*

Description

This function plots the results from a simulation generated by id_sim_gen.

Usage

```
id_plot_sims(sims, type = "RMSE")
```

Arguments

sims	A fitted idealstan object that has true data generated by id_sim_gen
type	Type of analysis of true versus fitted values, can be 'RMSE', 'Residuals' or 'Coverage'

id_post_pred *Generic Method for Obtaining Posterior Predictive Distribution from Stan Objects*

Description

This function is a generic that is used to match the functions used with ppc_bars to calculate the posterior predictive distribution of the data given the model.

Usage

```
id_post_pred(object, ...)
```

Arguments

object	A fitted idealstan object
...	All other parameters passed on to the underlying function.

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.

id_post_pred,idealstan-method *Posterior Prediction for* idealstan *objects*

Description

This function will draw from the posterior distribution, whether in terms of the outcome (prediction) or to produce the log-likelihood values.

This function can also produce either distribution of the outcomes (i.e., predictions) or the log-likelihood values of the posterior (set option type to 'log_lik'. For more information, see the package vignette How to Evaluate Models.

You can then use functions such as id_plot_ppc to see how well the model does returning the correct number of categories in the score/vote matrix. Also see help("posterior_predict", package = "rstanarm")

Usage

```
## S4 method for signature 'idealstan'
id_post_pred(object, draws = 100,
  output = "observed", type = "predict", sample_scores = NULL, ...)
```

Arguments

object	A fitted idealstan object
draws	The number of draws to use from the total number of posterior draws (default is 100).
output	If the model has an unbounded outcome (Poisson, continuous, etc.), then specify whether to show the 'observed' data (the default) or the binary output 'missing' showing whether an observation was predicted as missing or not
type	Whether to produce posterior predictive values ('predict', the default), or log-likelihood values ('log_lik'). See the How to Evaluate Models vignette for more info.
sample_scores	In addition to reducing the number of posterior draws used to calculate the posterior predictive distribution, which will reduce computational overhead. Only available for calculating predictive distributions, not log-likelihood values.
...	Any other arguments passed on to posterior_predict (currently none available)

---

id_sim_coverage *Function that computes how often the true value of the parameter is included within the 95/5 high posterior density interval*

Description

Function that computes how often the true value of the parameter is included within the 95/5 high posterior density interval

Usage

```
id_sim_coverage(obj, rep = 1, quantiles = c(0.95, 0.05))
```

Arguments

obj	A fitted idealstan object with true data generated by id_sim_gen
rep	How many times the models were fitted on new data, currently can only be 1
quantiles	What the quantile coverage of the high posterior density interval should be

`id_sim_gen` *Simulate IRT ideal point data*

Description

A function designed to simulate IRT ideal point data.

Usage

```
id_sim_gen(num_person = 20, num_bills = 50, model_type = "binary",
  latent_space = FALSE, absence_discrim_sd = 2,
  absence_diff_mean = 0.5, reg_discrim_sd = 2, diff_sd = 0.25,
  time_points = 1, time_process = "random", time_sd = 0.1,
  ideal_pts_sd = 1, prior_type = "gaussian", ordinal_outcomes = 3,
  inflate = FALSE, sigma_sd = 1)
```

Arguments

`num_person`	The number of persons/persons
`num_bills`	The number of items/bills
`model_type`	One of `'binary'`, `'ordinal_rating'`, `'ordinal_grm'`, `'poisson'` `'normal'`, or `'lognormal'`
`latent_space`	Whether to use the latent space formulation of the ideal point model `FALSE` by default. NOTE: currently, the package only has estimation for a binary response with the latent space formulation.
`absence_discrim_sd`	The SD of the discrimination parameters for the inflated model
`absence_diff_mean`	The mean intercept for the inflated model; increasing it will lower the total number of missing data
`reg_discrim_sd`	The SD of the discrimination parameters for the non-inflated model
`diff_sd`	The SD of the difficulty parameters (bill/item intercepts)
`time_points`	The number of time points for time-varying legislator/person parameters
`time_process`	The process used to generate the ideal points: currently either `'random'` for a random walk or `'AR'` for an AR1 process
`time_sd`	The standard deviation of the change in ideal points over time (should be low relative to `ideal_pts_sd`)
`ideal_pts_sd`	The SD for the person/person ideal points
`prior_type`	The statistical distribution that generates the data. Currently only 'gaussian' is supported.
`ordinal_outcomes`	If `model` is `'ordinal'`, an integer giving the total number of categories
`inflate`	If `TRUE`, an missing-data-inflated dataset is produced.
`sigma_sd`	If a normal or log-normal distribution is being fitted, this parameter gives the standard deviation of the outcome (i.e. the square root of the variance).

Details

This function produces simulated data that matches (as closely as possible) the models used in the underlying Stan code. Currently the simulation can produce inflated and non-inflated models with binary, ordinal (GRM and rating-scale), Poisson, Normal and Log-Normal responses.

Value

The results is a `idealdata` object that can be used in the `id_estimate` function to run a model. It can also be used in the simulation plotting functions.

See Also

`id_plot_sims` for plotting fitted models versus true values.

`id_sim_resid` *Residual function for checking estimated samples compared to true simulation scores Returns a data frame with residuals plus quantiles.*

Description

Residual function for checking estimated samples compared to true simulation scores Returns a data frame with residuals plus quantiles.

Usage

```
id_sim_resid(obj, rep = 1)
```

Arguments

`obj`	A fitted `idealstan` object with true data from `id_sim_gen`
`rep`	Over how many replicates to calculate residuals? Currently can only be 1

`id_sim_rmse` *RMSE function for calculating individual RMSE values compared to true simulation scores Returns a data frame with RMSE plus quantiles.*

Description

RMSE function for calculating individual RMSE values compared to true simulation scores Returns a data frame with RMSE plus quantiles.

Usage

```
id_sim_rmse(obj, rep = 1)
```

Arguments

obj	A fitted idealstan object with true data from id_sim_gen
rep	Over how many replicates to calculate RMSE? Currently can only be 1

---

launch_shinystan *Generic Method to Use* shinystan *with* idealstan

---

Description

A generic function for launching launch_shinystan.

Usage

```
launch_shinystan(object, ...)
```

Arguments

object	A fitted idealstan object.
...	Other arguments passed on to underlying function

---

launch_shinystan,idealstan-method *Function to Launch Shinystan with an* idealstan *Object*

Description

This wrapper will pull the rstan samples out of a fitted idealstan model and then launch launch_shinystan. This function is useful for examining convergence statistics of the underlying MCMC sampling.

Usage

```
## S4 method for signature 'idealstan'
launch_shinystan(object, pars = c("L_free",
  "sigma_reg_free", "sigma_abs_free", "restrict_high", "restrict_low",
  "restrict_ord", "steps_votes", "steps_votes_grm"), ...)
```

Arguments

object	A fitted idealstan object
pars	A character vector of parameters to select from the underlying rstan model object
...	Other parameters passed on to shinystan

See Also

shinystan

release_questions *Function that provides additional check questions for package release*

Description

Function that provides additional check questions for package release

Usage

```
release_questions()
```

senate114 *Rollcall vote data for 114th Senate*

Description

This rollcall vote object (see `rollcall`) contains voting records for the 114th Senate in the US Congress. Not all rollcalls are included, only those that had a 70-30 or closer split in the vote. The data can be pre-processed via the `id_make` function for estimation. See package vignette for details.

Usage

```
senate114
```

Format

A long data frame with one row for every vote cast by a Senator.

Source

http://www.voteview.com/

stan_trace *Plot the MCMC posterior draws by chain*

Description

This function allows you to produce trace plots for assessing the quality and convergence of MCMC chains.

Usage

```
stan_trace(object, ...)
```

Arguments

object	A fitted idealstan model
...	Other options passed on to stan_trace

Details

To use this function, you must pass a fitted idealstan object along with the name of a parameter in the model. To determine these parameter names, use the summary function or obtain the data from a plot by passing the return_data=TRUE option to id_plog_legis or id_plot_legis_dyn to find the name of the parameter in the Stan model.

This function is a simple wrapper around stan_trace. Please refer to that function's documentation for further options.

stan_trace,idealstan-method *Plot the MCMC posterior draws by chain*

Description

This function allows you to produce trace plots for assessing the quality and convergence of MCMC chains.

Usage

```
## S4 method for signature 'idealstan'
stan_trace(object, par = "L_full[1]")
```

Arguments

object	A fitted idealstan model
par	The character string name of a parameter in the model
...	Other options passed on to stan_trace

Details

To use this function, you must pass a fitted idealstan object along with the name of a parameter in the model. To determine these parameter names, use the summary function or obtain the data from a plot by passing the return_data=TRUE option to id_plog_legis or id_plot_legis_dyn to find the name of the parameter in the Stan model.

This function is a simple wrapper around stan_trace. Please refer to that function's documentation for further options.

---

summary,idealstan-method

Posterior Summaries for fitted idealstan *object*

Description

This function produces quantiles and standard deviations for the posterior samples of idealstan objects.

Usage

```
## S4 method for signature 'idealstan'
summary(object, pars = "ideal_pts",
  high_limit = 0.95, low_limit = 0.05, aggregate = TRUE)
```

Arguments

object	An idealstan object fitted by id_estimate
pars	Either 'ideal_pts' for person ideal points, 'items' for items/bills difficulty and discrimination parameters, and 'all' for all parameters in the model, including incidental parameters.
high_limit	A number between 0 and 1 reflecting the upper limit of the uncertainty interval (defaults to 0.95).
low_limit	A number between 0 and 1 reflecting the lower limit of the uncertainty interval (defaults to 0.05).
aggregate	Whether to return summaries of the posterior values or the full posterior samples. Defaults to TRUE.

Value

A tibble data frame with parameters as rows and descriptive statistics as columns

Index

∗Topic **datasets**
senate114, 30

derive_chain, 2

extract, *11*

ggplot, *15*

id_estimate, *4*, 4, *11*, *13–15*, *28*, *32*
id_extract, 10
id_extract,idealstan-method, 11
id_make, *3–9*, *11*, 11, *30*
id_plot, 14
id_plot,idealstan-method, 14
id_plot_all_hist, *15*, 15
id_plot_compare, *15*, 16
id_plot_cov, 16
id_plot_legis, *9*, *15*, 17
id_plot_legis_dyn, 19
id_plot_legis_var, 21
id_plot_ppc, 23, *25*
id_plot_ppc,idealstan-method, 23
id_plot_rhats, *15*, 24
id_plot_sims, *12*, *15*, 24, *28*
id_post_pred, *3*, *24*, 25
id_post_pred,idealstan-method, 25
id_sim_coverage, 26
id_sim_gen, *12*, *20*, *24*, *26*, 27, *28, 29*
id_sim_resid, 28
id_sim_rmse, 28
idealdata-class, 3
idealstan, 3, *8*
idealstan-class, 4
idealstan-package *(*idealstan*)*, 3

launch_shinystan, *29*, 29
launch_shinystan,idealstan-method, 29

mcmc_intervals, *17*

posterior_predict, *9*
ppc_bars, *23–25*

release_questions, 30
rollcall, *30*
rstan, *11*

senate114, 30
shinystan, *29*
stan, *4–6*, *11*
stan_trace, *31*, 31, *32*
stan_trace,idealstan-method, 31
summary, *9*
summary,idealstan-method, 32

tibble, *32*

vb, *5*, *8*