Package 'rstap'

February 6, 2019

Type Package

Version 1.0.3

Title Spatial Temporal Aggregated Predictor Models via 'stan'

Description Estimates previously compiled stap regression models using the 'rstan' package. Users specify models via a custom R syntax with a formula and data.frame plus additional arguments for priors.

License GPL (>= 3)

URL https://biostatistics4socialimpact.github.io/rstap

BugReports https://github.com/biostatistics4socialimpact/rstap/issues

Encoding UTF-8

LazyData true

ByteCompile true

Depends R (>= 3.4.0), Rcpp (>= 0.12.16), methods

Imports abind (>= 1.4-5), bayesplot (>= 1.5.0), ggplot2 (>= 2.2.1), lme4 (>= 1.1-8), loo (>= 2.0.0), Matrix (>= 1.2-13), nlme (>= 3.1-124), rstan (>= 2.18.1), rstantools (>= 1.4.0), stats, utils, pracma (>= 2.1.4), dplyr (>= 0.7.6)

LinkingTo StanHeaders (>= 2.18.0), rstan (>= 2.18.1), BH (>= 1.66.0-1), Rcpp (>= 0.12.16), RcppEigen(>= 0.3.3.3.0)

SystemRequirements GNU make

NeedsCompilation yes

RoxygenNote 6.1.0

Suggests testthat, digest, knitr, rmarkdown, pkgdown, fields

VignetteBuilder knitr

Author Adam Peterson [aut, cre]

Maintainer Adam Peterson <atpvyc@umich.edu>

Repository CRAN

Date/Publication 2019-02-06 20:30:03 UTC

R topics documented:

`rstap-package` *The 'rstap' package.*

Description

rstap is a package that implements spatial-temporal aggregated predictor functions in R. This allows for the modeling of features impact on measured subjects that can be related either through space or time.

References

Stan Development Team (2018). RStan: the R interface to Stan. R package version 2.17.3. http://mc-stan.org Adam Peterson: "rstap: An R Package for Spatial Temporal Aggregated Predictor Models", 2018; [http://arxiv.org/abs/1812.10208 arXiv:1812.10208].

`adapt_delta` *Target average acceptance probability*

Description

Details about the `adapt_delta` argument to **rstap**'s modeling functions - also found in the **rstanarm** documentation.

Details

For the No-U-Turn Sampler (NUTS), the variant of Hamiltonian Monte Carlo used used by **rstap**, `adapt_delta` is the target average proposal acceptance probability for adaptation.

The default value of `adapt_delta` is 0.95

In general you should not need to change `adapt_delta` unless you see a warning message about divergent transitions, in which case you can increase `adapt_delta` from the default to a value *closer* to 1 (e.g. from 0.95 to 0.99, or from 0.99 to 0.999, etc). The step size used by the numerical integrator is a function of `adapt_delta` in that increasing `adapt_delta` will result in a smaller step size and fewer divergences. Increasing `adapt_delta` will typically result in a slower sampler, but it will always lead to a more robust sampler.

References

Stan Development Team. (2017). *Stan Modeling Language Users Guide and Reference Manual.* http://mc-stan.org/documentation/

`as.matrix.stapreg` *Extract the posterior sample via matrix*

Description

The posterior sample —the post-warmup draws from the posterior distribution— can be extracted from a fitted model object as a matrix, data frame, or array. The `as.matrix` and `as.data.frame` methods merge all chains together, whereas the `as.array` method keeps the chains separate.

Usage

```
## S3 method for class 'stapreg'
as.matrix(x, ..., pars = NULL, regex_pars = NULL)

## S3 method for class 'stapreg'
as.array(x, ..., pars = NULL, regex_pars = NULL)

## S3 method for class 'stapreg'
as.data.frame(x, ..., pars = NULL, regex_pars = NULL)
```

Arguments

x	A fitted model object returned by one of the **rstap** modeling functions. See stapreg-objects.
...	Ignored.
pars	An optional character vector of parameter names.
regex_pars	An optional character vector of regular expressions to use for parameter selection. regex_pars can be used in place of pars or in addition to pars. Currently, all functions that accept a regex_pars argument ignore it for models fit using optimization.

Value

A matrix, data.frame, or array, the dimensions of which depend on pars and regex_pars, as well as the model and estimation algorithm (see the Description section above).

See Also

stapreg-methods

Examples

```
if (!exists("example_model")) example(example_model)
# Extract posterior sample after MCMC
draws <- as.matrix(example_model)
print(dim(draws))

# For example, we can see that the median of the draws for the intercept
# is the same as the point estimate rstanarm uses
print(median(draws[, "(Intercept)"]))
print(example_model$coefficients[["(Intercept)"]])

# The as.array method keeps the chains separate
draws_array <- as.array(example_model)
print(dim(draws_array)) # iterations x chains x parameters
```

check_constant_vars *Check if any variables in a model frame are constants (the exception is that a constant variable of all 1's is allowed)*

Description

Check if any variables in a model frame are constants (the exception is that a constant variable of all 1's is allowed)

Usage

```
check_constant_vars(mf)
```

Arguments

mf	A model frame or model matrix

Value

If no constant variables are found mf is returned, otherwise an error is thrown.

example_model *Example model*

Description

A model for use in **rstap** examples.

Format

Calling `example("example_model")` will run the model in the Examples section, below, and the resulting stapreg object will then be available in the global environment. The `chains` and `iter` arguments are specified to make this example small in size. In practice, we recommend that they be left unspecified in order to use the default values (4 and 2000 respectively) or increased if there are convergence problems. The `cores` argument is optional and on a multicore system, the user may well want to set that equal to the number of chains being executed.

See Also

The Longituinal Vignette for `stap_glmer`.

Examples

```
 ## following lines make example run faster
distdata <- subset(homog_longitudinal_bef_data[,c("subj_ID","measure_ID","class","dist")],
                   subj_ID<=10)
timedata <- subset(homog_longitudinal_bef_data[,c("subj_ID","measure_ID","class","time")],
                   subj_ID<=10)
timedata$time <- as.numeric(timedata$time)
subjdata <- subset(homog_longitudinal_subject_data,subj_ID<=10)
example_model <-
  stap_glmer(y_bern ~ centered_income +  sex + centered_age + stap(Coffee_Shop) + (1|subj_ID),
             family = gaussian(),
             subject_data = subjdata,
             distance_data = distdata,
             time_data = timedata,
             subject_ID = 'subj_ID',
             group_ID = 'measure_ID',
             prior_intercept = normal(location = 25, scale = 4, autoscale = FALSE),
             prior = normal(location = 0, scale = 4, autoscale = FALSE),
             prior_stap = normal(location = 0, scale = 4),
             prior_theta = list(Coffee_Shop = list(spatial = log_normal(location = 1,
                                                                        scale = 1),
                                                   temporal = log_normal(location = 1,
                                                                         scale = 1))),
             max_distance = 3, max_time = 50,
             # chains, cores, and iter set to make the example small and fast
             chains = 1, iter = 25, cores = 1)
```

`get_stapless_formula` *get_stapless_formula*

Description

Get formula for typical covariates

Usage

```
get_stapless_formula(f)
```

Arguments

`f`	formula from stap_glm

Value

formula without ~ stap() components

log_lik.stapreg *Pointwise log-likelihood matrix*

Description

For models fit using MCMC, the log_lik method returns the S by N pointwise log-likelihood matrix, where S is the size of the posterior sample and N is the number of data points.

Usage

```
## S3 method for class 'stapreg'
log_lik(object, newsubjdata = NULL,
  newdistdata = NULL, newtimedata = NULL, offset = NULL, ...)
```

Arguments

object	A fitted model object returned by one of the **rstap** modeling functions. See stapreg-objects.
newsubjdata	Optionally, a data frame of the subject-specific data in which to look for variables with which to predict. If omitted, the original datasets are used. If newsubjdata is provided and any variables were transformed (e.g. rescaled) in the data used to fit the model, then these variables must also be transformed in newsubjdata. Also see the Note section below for a note about using the newsubjdata argument with with binomial models.
newdistdata	If newsubjdata is provided a data frame of the subject-distance must also be given for models with a spatial component - can be the same as original distance_dataframe
newtimedata	If newsubjdata is provided, a data frame of the subject-time data must also be given for models with a temporal component
offset	A vector of offsets. Only required if newsubjdata is specified and an offset was specified when fitting the model.
...	Currently ignored.

Value

A S by N matrix, where S is the size of the posterior sample and N is the number of data points.

pairs.stapreg *Pairs method for stapreg objects*

Description

Interface to **bayesplot**'s mcmc_pairs function for use with **rstap** models. Be careful not to specify too many parameters to include or the plot will be both hard to read and slow to render.

Usage

```
## S3 method for class 'stapreg'
pairs(x, pars = NULL, regex_pars = NULL,
  condition = pairs_condition(nuts = "accept_stat__"), ...)
```

Arguments

x	A fitted model object returned by one of the **rstap** modeling functions. See stapreg-objects.
pars	An optional character vetor of parameter names. All parameters are included by default, but for models with more than just a few parameters it may be far too many to visualize on a small computer screen and also may require substantial computing time.
regex_pars	An optional character vector of regular expressions to use for parameter selection. regex_pars can be used in place of pars or in addition to pars. Currently, all functions that accept a regex_pars argument ignore it for models fit using optimization.
condition	Same as the condition argument to mcmc_pairs except the *default is different* for **rstap** models. By default, the mcmc_pairs function in the **bayesplot** package plots some of the Markov chains (half, in the case of an even number of chains) in the panels above the diagonal and the other half in the panels below the diagonal. However since we know that **rstap** models were fit using Stan (which **bayesplot** doesn't assume) we can make the default more useful by splitting the draws according to the accept_stat__ diagnostic. The plots below the diagonal will contain realizations that are below the median accept_stat__ and the plots above the diagonal will contain realizations that are above the median accept_stat__. To change this behavior see the documentation of the condition argument at mcmc_pairs.
...	Optional arguments passed to mcmc_pairs. The np, lp, and max_treedepth arguments to mcmc_pairs are handled automatically by **rstap** and do not need to be specified by the user in .... The arguments that can be specified in ... include transformations, diag_fun, off_diag_fun, diag_args, off_diag_args, and np_style. These arguments are documented thoroughly on the help page for mcmc_pairs.

plot.stapreg *Plot method for stapreg objects*

Description

The plot method for stapreg-objects provides a convenient interface to the MCMC module in the **bayesplot** package for plotting MCMC draws and diagnostics. It is also straightforward to use the functions from the **bayesplot** package directly rather than via the plot method. Examples of both methods of plotting are given below.

Usage

```
## S3 method for class 'stapreg'
plot(x, plotfun = "intervals", pars = NULL,
  regex_pars = NULL, ...)
```

Arguments

x	A fitted model object returned by one of the **rstap** modeling functions. See stapreg-objects.
plotfun	A character string naming the **bayesplot** MCMC function to use. The default is to call mcmc_intervals. plotfun can be specified either as the full name of a **bayesplot** plotting function (e.g. "mcmc_hist") or can be abbreviated to the part of the name following the "mcmc_" prefix (e.g. "hist"). To get the names of all available MCMC functions see available_mcmc.
pars	An optional character vector of parameter names.
regex_pars	An optional character vector of regular expressions to use for parameter selection. regex_pars can be used in place of pars or in addition to pars. Currently, all functions that accept a regex_pars argument ignore it for models fit using optimization.
...	Additional arguments to pass to plotfun for customizing the plot. These are described on the help pages for the individual plotting functions. For example, the arguments accepted for the default plotfun="intervals" can be found at mcmc_intervals.

Value

Either a ggplot object that can be further customized using the **ggplot2** package, or an object created from multiple ggplot objects (e.g. a gtable object created by arrangeGrob).

References

Gabry, J., Simpson, D., Vehtari, A., Betancourt, M., and Gelman, A. (2018). Visualization in Bayesian workflow. *Journal of the Royal Statistical Society Series A*, accepted for publication. arXiv preprint: http://arxiv.org/abs/1709.01449.

See Also

- The vignettes in the **bayesplot** package for many examples.
- MCMC-overview (**bayesplot**) for links to the documentation for all the available plotting functions.
- color_scheme_set (**bayesplot**) to change the color scheme used for plotting.
- pp_check for graphical posterior predictive checks.

Examples

```
## Not run:
# Not run for CRAN check speed
fit_glm <- stap_glm(formula = y ~ sex + sap(Fast_Food),
                    subject_data = homog_subject_data,
                      distance_data = homog_distance_data,
                      family = gaussian(link = 'identity'),
                      subject_ID = 'subj_id',
                      prior = normal(location = 0, scale = 5, autoscale = F),
                      prior_intercept = normal(location = 25, scale = 5, autoscale = F),
                      prior_stap = normal(location = 0, scale = 3, autoscale = F),
                      prior_theta = log_normal(location = 1, scale = 1),
                      prior_aux = cauchy(location = 0,scale = 5),
                      max_distance = max(homog_distance_data$Distance),
                      chains = CHAINS, iter = ITER,
                      refresh = -1,verbose = F)

plot(fit_glm, plotfun = 'mcmc_hist', pars = "Fast_Food")

## End(Not run)
```

---

posterior_interval.stapreg

Posterior uncertainty intervals

Description

The posterior_interval function computes Bayesian posterior uncertainty intervals. These intervals are also often referred to as *credible* intervals.

Usage

```
## S3 method for class 'stapreg'
posterior_interval(object, prob = 0.9,
  type = "central", pars = NULL, regex_pars = NULL, ...)
```

Arguments

`object`	A fitted model object returned by one of the **rstap** modeling functions. See `stapreg-objects`.
`prob`	A number $p \in (0, 1)$ indicating the desired probability mass to include in the intervals. The default is to report 90% intervals (`prob=0.9`) rather than the traditionally used 95% (see Details).
`type`	The type of interval to compute. Currently the only option is `"central"` (see Details). A central $100p\%$ interval is defined by the $\alpha/2$ and $1-\alpha/2$ quantiles, where $\alpha = 1 - p$.
`pars`	An optional character vector of parameter names.
`regex_pars`	An optional character vector of regular expressions to use for parameter selection. `regex_pars` can be used in place of `pars` or in addition to `pars`. Currently, all functions that accept a `regex_pars` argument ignore it for models fit using optimization.
`...`	Currently ignored.

Details

Interpretation: Unlike for a frenquentist confidence interval, it is valid to say that, conditional on the data and model, we believe that with probability p the value of a parameter is in its $100p\%$ posterior interval. This intuitive interpretation of Bayesian intervals is often erroneously applied to frequentist confidence intervals. See Morey et al. (2015) for more details on this issue and the advantages of using Bayesian posterior uncertainty intervals (also known as credible intervals).

Default 90% intervals: We default to reporting 90% intervals rather than 95% intervals for several reasons:

- Computational stability: 90% intervals are more stable than 95% intervals (for which each end relies on only 2.5% of the posterior draws).
- Relation to Type-S errors (Gelman and Carlin, 2014): 95% of the mass in a 90% central interval is above the lower value (and 95% is below the upper value). For a parameter θ, it is therefore easy to see if the posterior probability that $\theta > 0$ (or $\theta < 0$) is larger or smaller than 95%.

Of course, if 95% intervals are desired they can be computed by specifying `prob=0.95`.

Types of intervals: Currently `posterior_interval` only computes central intervals because other types of intervals are rarely useful for the models that **rstap** can estimate. Additional possibilities may be provided in future releases as more models become available.

Value

A matrix with two columns and as many rows as model parameters (or the subset of parameters specified by `pars` and/or `regex_pars`). 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 will be `"5%"` and `"95%"`, respectively.

References

Gelman, A. and Carlin, J. (2014). Beyond power calculations: assessing Type S (sign) and Type M (magnitude) errors. *Perspectives on Psychological Science*. 9(6), 641–51.

Morey, R. D., Hoekstra, R., Rouder, J., Lee, M. D., and Wagenmakers, E. (2016). The fallacy of placing confidence in confidence intervals. *Psychonomic Bulletin & Review*. 23(1), 103–123.

See Also

predictive_interval for predictive intervals.

Examples

```
if (!exists("example_model")) example(example_model)
posterior_interval(example_model)
posterior_interval(example_model, regex_pars = "Coffee_Shop")
```

---

posterior_predict.stapreg

Draw from posterior predictive distribution

Description

The posterior predictive distribution is the distribution of the outcome implied by the model after using the observed data to update our beliefs about the unknown parameters in the model. Simulating data from the posterior predictive distribution using the observed predictors is useful for checking the fit of the model. Drawing from the posterior predictive distribution at interesting values of the predictors also lets us visualize how a manipulation of a predictor affects (a function of) the outcome(s). With new observations of predictor variables we can use the posterior predictive distribution to generate predicted outcomes.

Usage

```
## S3 method for class 'stapreg'
posterior_predict(object, newsubjdata = NULL,
  newdistdata = NULL, newtimedata = NULL, draws = NULL,
  subject_ID = NULL, group_ID = NULL, re.form = NULL, fun = NULL,
  seed = NULL, offset = NULL, ...)
```

Arguments

object	A fitted model object returned by one of the **rstap** modeling functions. See stapreg-objects.
newsubjdata	Optionally, a data frame of the subject-specific data in which to look for variables with which to predict. If omitted, the original datasets are used. If newsubjdata is provided and any variables were transformed (e.g. rescaled) in the data used to fit the model, then these variables must also be transformed in newsubjdata. This only applies if variables were transformed before passing the data to one

	of the modeling functions and *not* if transformations were specified inside the model formula. Also see the Note section below for a note about using the `newdata` argument with with binomial models.
`newdistdata`	If newsubjdata is provided a data frame of the subject-distance must also be given for models with a spatial component
`newtimedata`	If newsubjdata is provided, a data frame of the subject-time data must also be given for models with a temporal component
`draws`	An integer indicating the number of draws to return. The default and maximum number of draws is the size of the posterior sample.
`subject_ID`	name of column to join on between subject_data and bef_data
`group_ID`	name of column to join on between `subject_data` and bef_data that uniquely identifies the correlated groups (e.g. visits,schools). Currently only one group (e.g. a measurement ID) can be accounted for in a spatial temporal setting.
`re.form`	If `object` contains `group-level` parameters, a formula indicating which group-level parameters to condition on when making predictions. `re.form` is specified in the same form as for `predict.merMod`. The default, `NULL`, indicates that all estimated group-level parameters are conditioned on. To refrain from conditioning on any group-level parameters, specify `NA` or `~0`. The `newdata` argument may include new *levels* of the grouping factors that were specified when the model was estimated, in which case the resulting posterior predictions marginalize over the relevant variables.
`fun`	An optional function to apply to the results. `fun` is found by a call to `match.fun` and so can be specified as a function object, a string naming a function, etc.
`seed`	An optional `seed` to use.
`offset`	A vector of offsets. Only required if `newsubjdata` is specified and an `offset` argument was specified when fitting the model.
`...`	optional arguments to pass to pp_args

Value

A `draws` by `nrow(newdata)` matrix of simulations from the posterior predictive distribution. Each row of the matrix is a vector of predictions generated using a single draw of the model parameters from the posterior distribution. The returned matrix will also have class `"ppd"` to indicate it contains draws from the posterior predictive distribution.

Note

For binomial models with a number of trials greater than one (i.e., not Bernoulli models), if `newsubjdata` is specified then it must include all variables needed for computing the number of binomial trials to use for the predictions. For example if the left-hand side of the model formula is `cbind(successes, failures)` then both `successes` and `failures` must be in `newdata`. The particular values of `successes` and `failures` in `newdata` do not matter so long as their sum is the desired number of trials. If the left-hand side of the model formula were `cbind(successes, trials - successes)` then both `trials` and `successes` would need to be in `newsubjdata`, probably with `successes` set to `0` and `trials` specifying the number of trials.

See Also

Examples of posterior predictive checking can also be found in the **rstanarm** vignettes and demos.

predictive_error and predictive_interval.

Examples

```
if (!exists("example_model")) example(example_model)
yrep <- posterior_predict(example_model)
table(yrep)


# If using new data the all pertinent data must be submitted to the function including subject_ID
# The same distance and time datasets below are used in the original function
# Which will associate the same spatio-temporal exposure to this subject's new fixed covariates.
newdata <- data.frame(subj_ID = 1, measure_ID = 1, centered_income = 0, sex = 0, centered_age = 0)
pps <- posterior_predict(example_model, newsubjdata = newdata,
                         newdistdata= subset(distdata,subj_ID == 1, measure_ID == 1),
                         newtimedata = subset(timedata, subj_ID == 1, measure_ID == 1),
                         subject_ID = "subj_ID", group_ID = "measure_ID" )
```

---

predictive_error *In-sample or out-of-sample predictive errors*

Description

This is a convenience function for computing $y - y^{rep}$ (in-sample, for observed y) or $y - \tilde{y}$ (out-of-sample, for new or held-out y). The method for stapreg objects calls posterior_predict internally, whereas the method for objects with class "ppd" accepts the matrix returned by posterior_predict as input and can be used to avoid multiple calls to posterior_predict.

The **rstap** model-fitting functions return an object of class 'stapreg', which is a list containing at a minimum the components listed below. Each stapreg object will also have additional classes (e.g. 'glm') and several additional components depending on the model and estimation algorithm.

Usage

```
## S3 method for class 'stapreg'
predictive_error(object, newsubjdata = NULL,
  newdistdata = NULL, newtimedata = NULL, draws = NULL,
  re.form = NULL, seed = NULL, offset = NULL, ...)
```

Arguments

`object` Either a fitted model object returned by one of the **rstap** modeling functions (a stapreg object) or, for the `"ppd"` method, a matrix of draws from the posterior predictive distribution returned by `posterior_predict`.

`newsubjdata, newdistdata, newtimedata, draws, seed, offset, re.form` Optional arguments passed to `posterior_predict`. For binomial models, please see the **Note** section below if `newsubjdata` will be specified.

`...` Currently ignored.

Value

A `draws` by `nrow(newsubjdata)` matrix. If `newsubjdata` is not specified then it will be `draws` by `nobs(object)`.

Elements for `stapreg` objects

`coefficients` Point estimates, as described in `print.stapreg`.

`ses` Standard errors based on `mad`, as described in `print.stapreg`.

`residuals` Residuals of type `'response'`.

`fitted.values` Fitted mean values. For GLMs the linear predictors are transformed by the inverse link function.

`linear.predictors` Linear fit on the link scale. For linear models this is the same as `fitted.values`.

`covmat` Variance-covariance matrix for the coefficients based on draws from the posterior distribution, the variational approximation, or the asymptotic sampling distribution, depending on the estimation algorithm.

`model,x,y,z` If requested, the the model frame, model matrix and response variable used, respectively. Note that z corresponds to the fixed covariates, z to the spatial aggregated covariates, and y the response.

`family` The `family` object used.

`call` The matched call.

`formula` The model `formula`.

`data,offset,weights` The `data`, `offset`, and `weights` arguments.

`prior.info` A list with information about the prior distributions used.

`stapfit,stan_summary` The object of `stanfit-class` returned by RStan and a matrix of various summary statistics from the stapfit object.

`rstan_version` The version of the **rstan** package that was used to fit the model.

Note

The **Note** section in `posterior_predict` about `nnewsubjdata` for binomial models also applies for `predictive_error`, with one important difference. For `posterior_predict` if the left-hand side of the model formula is `cbind(successes, failures)` then the particular values of `successes` and `failures` in `newsubjdata` don't matter, only that they add to the desired number of trials. **This is not the case for** `predictive_error`. For `predictive_error` the particular value of `successes` matters because it is used as y when computing the error.

See Also

posterior_predict to draw from the posterior predictive distribution without computing predictive errors.

---

predictive_interval.stapreg

Predictive intervals

Description

The predictive_interval function computes Bayesian predictive intervals. The method for stapreg objects calls posterior_predict internally, whereas the method for objects of class "ppd" accepts the matrix returned by posterior_predict as input and can be used to avoid multiple calls to posterior_predict.

Usage

```
## S3 method for class 'stapreg'
predictive_interval(object, prob = 0.9,
  newsubjdata = NULL, newdistdata = NULL, newtimedata = NULL,
  draws = NULL, subject_ID = NULL, group_ID = NULL, re.form = NULL,
  fun = NULL, seed = NULL, offset = NULL, ...)

## S3 method for class 'ppd'
predictive_interval(object, prob = 0.9, ...)
```

Arguments

object	Either a fitted model object returned by one of the **rstap** modeling functions (a stapreg object) or, for the "ppd" method, a matrix of draws from the posterior predictive distribution returned by posterior_predict.
prob	A number $p \in (0, 1)$ indicating the desired probability mass to include in the intervals. The default is to report 90% intervals (prob=0.9) rather than the traditionally used 95% .
newsubjdata	Optionally, a data frame of the subject-specific data in which to look for variables with which to predict. If omitted, the original datasets are used. If newsubjdata is provided and any variables were transformed (e.g. rescaled) in the data used to fit the model, then these variables must also be transformed in newsubjdata. This only applies if variables were transformed before passing the data to one of the modeling functions and *not* if transformations were specified inside the model formula. Also see the Note section below for a note about using the newsubjdata argument with with binomial models.
newdistdata	If newsubjdata is provided a data frame of the subject-distance must also be given for models with a spatial component
newtimedata	If newsubjdata is provided, a data frame of the subject-time data

draws, fun, offset, re.form, seed
: Passed to posterior_predict.

subject_ID
: same as stap_glm

group_ID
: same as stap_glmer

...
: Currently ignored.

Value

A matrix with two columns and as many rows as are in newsubjdata. If newsubjdata is not provided then the matrix will have as many rows as the data used to fit the model. For a given value of prob, p, the columns correspond to the lower and upper $100p\%$ central 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 will be "5%" and "95%", respectively.

See Also

predictive_error, posterior_predict, posterior_interval

Examples

```
if (!exists("example_model")) example(example_model)

predictive_interval(example_model)
newdata <- data.frame(subj_ID = c(1,1), measure_ID = c(1,2),
                      centered_income = c(-1,-.7), sex = c(0,0),
                      centered_age = c(-1,-.7))
# newdata
predictive_interval(example_model, newsubjdata = newdata,
                    newdistdata = distdata,
                    newtimedata = timedata,
                    subject_ID = "subj_ID",
                    group_ID = "measure_ID")
```

---

print.stapreg *Print method for stapreg objects*

Description

The print method for stapreg objects displays a compact summary of the fitted model. See the **Details** section below for descriptions of the different components of the printed output. For additional summary statistics and diagnostics use the summary method.

Usage

```
## S3 method for class 'stapreg'
print(x, digits = 1, ...)
```

Arguments

x	A fitted model object returned by one of the **rstap** modeling functions. See stapreg-objects.
digits	Number of digits to use for formatting numbers.
...	Ignored.

Details

Point estimates: Point estimates are medians computed from simulations. For models fit using MCMC ("sampling") the posterior sample is used. The point estimates reported are the same as the values returned by coef.

Uncertainty estimates (MAD_SD): The standard deviations reported (labeled MAD_SD in the print output) are computed from the same set of draws described above and are proportional to the median absolute deviation (mad) from the median. Compared to the raw posterior standard deviation, the MAD_SD will be more robust for long-tailed distributions. These are the same as the values returned by se.

Additional output:

- The median and MAD_SD are also reported for mean_PPD, the sample average posterior predictive distribution of the outcome. This is useful as a quick diagnostic. A useful heuristic is to check if mean_PPD is plausible when compared to mean(y). If it is plausible then this does *not* mean that the model is good in general (only that it can reproduce the sample mean), however if mean_PPD is implausible then it is a sign that something is wrong (severe model misspecification, problems with the data, computational issues, etc.).
- For GLMs with group-specific terms (see stap_glmer) the printed output also shows point estimates of the standard deviations of the group effects (and correlations if there are both intercept and slopes that vary by group).

Value

Returns x, invisibly.

See Also

summary.stapreg, stapreg-methods

---

priors *Prior distributions and options*

Description

The functions described on this page are used to specify the prior-related #' arguments of the various modeling functions in the **rstap** package (to view the priors used for an existing model see `prior_summary`). The default priors used in the various **rstap** modeling functions are intended to be *weakly informative* in that they provide moderate regularlization and help stabilize computation. For many applications the defaults will perform well, but prudent use of more informative priors is encouraged. All of the priors here are informed by the priors in **rstanarm**, though it should be noted that the heirarchical shape priors are not included.

Usage

```
normal(location = 0, scale = NULL, autoscale = TRUE)

student_t(df = 1, location = 0, scale = NULL, autoscale = TRUE)

cauchy(location = 0, scale = NULL, autoscale = TRUE)

laplace(location = 0, scale = NULL, autoscale = TRUE)

lasso(df = 1, location = 0, scale = NULL, autoscale = TRUE)

product_normal(df = 2, location = 0, scale = 1)

exponential(rate = 1, autoscale = TRUE)

log_normal(location = 0, scale = 1)

decov(regularization = 1, concentration = 1, shape = 1, scale = 1)

lkj(regularization = 1, scale = 10, df = 1, autoscale = TRUE)
```

Arguments

`location`	Prior location. In most cases, this is the prior mean, but for `cauchy` (which is equivalent to `student_t` with `df=1`), the mean does not exist and `location` is the prior median. The default value is 0.
`scale`	Prior scale. The default depends on the family (see **Details**).
`autoscale`	A logical scalar, defaulting to `TRUE`. If `TRUE` then the scales of the priors on the intercept and regression coefficients may be additionally modified internally by **rstanarm** in the following cases. First, for Gaussian models only, the prior scales for the intercept, coefficients, and the auxiliary parameter `sigma` (error standard deviation) are multiplied by `sd(y)`. Additionally — not only for Gaussian models — if the `QR` argument to the model fitting function (e.g. `stap_glm`) is `FALSE` then: for a predictor with only one value nothing is changed; for a predictor `x` with exactly two unique values, we take the user-specified (or default) scale(s) for the selected priors and divide by the range of `x`; for a predictor `x` with more than two unique values, we divide the prior scale(s) by `sd(x)`.

df	Prior degrees of freedom. The default is 1 for student_t, in which case it is equivalent to cauchy. For the product_normal prior, the degrees of freedom parameter must be an integer (vector) that is at least 2 (the default).
rate	Prior rate for the exponential distribution. Defaults to 1. For the exponential distribution the rate parameter is the *reciprocal* of the mean.
regularization	Exponent for an LKJ prior on the correlation matrix in the decov or lkj prior. The default is 1, implying a joint uniform prior.
concentration	Concentration parameter for a symmetric Dirichlet distribution. The default is 1, implying a joint uniform prior.
shape	Shape parameter for a gamma prior on the scale parameter in the decov prior. If shape and scale are both 1 (the default) then the gamma prior simplifies to the unit-exponential distribution.

Details

The details depend on the family of the prior being used:

Student t family: Family members:

- normal(location, scale)
- student_t(df, location, scale)
- cauchy(location, scale)

Each of these functions also takes an argument autoscale which is relevant if used for any of the non-stap related parameters. It is not used otherwise.

For the prior distribution for the intercept, location, scale, and df should be scalars. As the degrees of freedom approaches infinity, the Student t distribution approaches the normal distribution and if the degrees of freedom are one, then the Student t distribution is the Cauchy distribution.

If scale is not specified it will default to 10 for the intercept and 2.5 for the other coefficients, unless the probit link function is used, in which case these defaults are scaled by a factor of dnorm(0)/dlogis(0), which is roughly 1.6.

If the autoscale argument is TRUE (the default), then the scales will be further adjusted as described above in the documentation of the autoscale argument in the **Arguments** section.

Laplace family: Family members:

- laplace(location, scale)
- lasso(df, location, scale)

Each of these functions also takes an argument autoscale.

The Laplace distribution is also known as the double-exponential distribution. It is a symmetric distribution with a sharp peak at its mean / median / mode and fairly long tails. This distribution can be motivated as a scale mixture of normal distributions and the remarks above about the normal distribution apply here as well.

The lasso approach to supervised learning can be expressed as finding the posterior mode when the likelihood is Gaussian and the priors on the coefficients have independent Laplace distributions. It is commonplace in supervised learning to choose the tuning parameter by cross-validation, whereas a more Bayesian approach would be to place a prior on "it", or rather its reciprocal in our case (i.e. *smaller* values correspond to more shrinkage toward the prior location vector). We

use a chi-square prior with degrees of freedom equal to that specified in the call to `lasso` or, by default, 1. The expectation of a chi-square random variable is equal to this degrees of freedom and the mode is equal to the degrees of freedom minus 2, if this difference is positive.

It is also common in supervised learning to standardize the predictors before training the model. We do not recommend doing so. Instead, it is better to specify `autoscale = TRUE` (the default value), which will adjust the scales of the priors according to the dispersion in the variables. See the documentation of the `autoscale` argument above and also the `prior_summary` page for more information.

Product-normal family: Family members:

- `product_normal(df, location, scale)`

The product-normal distribution is the product of at least two independent normal variates each with mean zero, shifted by the `location` parameter. It can be shown that the density of a product-normal variate is symmetric and infinite at `location`, so this prior resembles a "spike-and-slab" prior for sufficiently large values of the `scale` parameter. For better or for worse, this prior may be appropriate when it is strongly believed (by someone) that a regression coefficient "is" equal to the `location`, parameter even though no true Bayesian would specify such a prior.

Each element of `df` must be an integer of at least 2 because these "degrees of freedom" are interpreted as the number of normal variates being multiplied and then shifted by `location` to yield the regression coefficient. Higher degrees of freedom produce a sharper spike at `location`.

Each element of `scale` must be a non-negative real number that is interpreted as the standard deviation of the normal variates being multiplied and then shifted by `location` to yield the regression coefficient. In other words, the elements of `scale` may differ, but the k-th standard deviation is presumed to hold for all the normal deviates that are multiplied together and shifted by the k-th element of `location` to yield the k-th regression coefficient. The elements of `scale` are not the prior standard deviations of the regression coefficients. The prior variance of the regression coefficients is equal to the scale raised to the power of 2 times the corresponding element of `df`. Thus, larger values of `scale` put more prior volume on values of the regression coefficient that are far from zero.

Value

A named list to be used internally by the **rstap** model fitting functions.

References

Gelman, A., Jakulin, A., Pittau, M. G., and Su, Y. (2008). A weakly informative default prior distribution for logistic and other regression models. *Annals of Applied Statistics*. 2(4), 1360–1383.

Can assign priors to names N05 <- normal(0, 5)

See Also

The various vignettes for the **rstanarm** and **rstap** packages also discuss and demonstrate the use of some of the supported prior distributions.

#'

`prior_summary.stapreg` *Summarize the priors used for an rstap model*

Description

The `prior_summary` method provides a summary of the prior distributions used for the parameters in a given model. In some cases the user-specified prior does not correspond exactly to the prior used internally by **rstap** (see the sections below). Especially in these cases, but also in general, it can be much more useful to visualize the priors.

Usage

```
## S3 method for class 'stapreg'
prior_summary(object, digits = 2, ...)
```

Arguments

`object`	A fitted model object returned by one of the **rstap** modeling functions. See `stapreg-objects`.
`digits`	Number of digits to use for rounding.
`...`	Currently ignored by the method for stapreg objects.

Value

A list of class "prior_summary.stapreg", which has its own print method.

Intercept (after predictors centered)

For **rstap** modeling functions that accept a `prior_intercept` argument, the specified prior for the intercept term applies to the intercept after **rstap** internally centers the predictors so they each have mean zero. The estimate of the intercept returned to the user correspond to the intercept with the predictors as specified by the user (unmodified by **rstap**), but when *specifying* the prior the intercept can be interpreted as the expected outcome when the predictors are set to their means.

Adjusted scales

For some models you may see "`adjusted scale`" in the printed output and adjusted scales included in the object returned by `prior_summary`. These adjusted scale values are the prior scales actually used by **rstap** and are computed by adjusting the prior scales specified by the user to account for the scales of the predictors (as described in the documentation for the `autoscale` argument). To disable internal prior scale adjustments set the `autoscale` argument to `FALSE` when setting a prior using one of the distributions that accepts an `autoscale` argument. For example, `normal(0, 5, autoscale=FALSE)` instead of just `normal(0, 5)`. Note that for prior_stap all priors are set on the scaled covariates this is done so that multiple priors placed on differing staps can be roughly comparable.

Spatial - Temporal Scales

If only one prior was specified this will be returned in a section entitled "STAP scales". Otherwise no priors will be printed out. A more structured system for STAP prior printing is planned for the next release.

See Also

The priors help page and the *Prior Distributions* vignette from the **rstanarm** package.

`rstap-datasets` *Datasets for rstap examples*

Description

Small datasets for use in **rstap** examples and vignettes.

Format

`homog_longitudinal_bef_data` Simulated data for the longitudinal simulation

- `subj_ID`: The subject unique identifier
- `measure_ID`: The measurement unique identifier
- `bef_ID` The Built Environment Unique identifier
- `measure_date` The date at which the subject was measured
- `date_open`: The date at which the business opened
- `date_close`: The date at which the business may have closed; NA if the business is still open
- `date`: The date at which the subject first moved to the location associated with the distance and time with the built environment feature
- `class`: The kind of built environment feature. Only one is in the simulated dataset - "Coffee Shop"
- `dist`: The distance between the subject and BEF at the date to be associated with the measure ID
- `time`: The time for which the subject was "exposed" to the BEF at corresponding distance

Source: Longitudinal Vignette

`homog_longitudinal_subject_data`

- `subj_ID`: The subject unique identifier
- `Income`: Simulated continuous covariate
- `measure_date`: The simulated date the subject was measured
- `ran_int`: Random intercept generated for the longitudinal I simulation
- `y`: Continuous outcome simulated for longitudinal I simulation - meant to be akin to BMI
- `y_bern`: Bernoulli outcome simulated
- `sex`: Discrete 1-0 covariate simulated to be akin to sex
- `Coffee_Shop`: The "true" Coffee Shop Exposure covariate
- `centered_income`: scaled and centered version of Income covariate

- `centered_age`: scaled and centered version of Age covariate

Source: \hrefhttps://biostatistics4socialimpact.github.io/rstap/articles/longitudinal-I.htmlLongitudinal Vignette

`homog_subject_data`
- subj_idThe subject unique identifier
- yContinuous simulated outcome, meant to be BMI
- sexdiscrete factor coded "M" for male, "F" for females

Source: Introduction Vignette

`homog_distance_data`
- `subj_id`: The subject unique identifier
- `BEF` Built Environment Feature class identifier - only one included in this dataset "Fast_Food"
- Distance: The euclidean distance between the row's subject and Fast Food restaurant locations'

Source: Introduction Vignette

`stapreg` *Create a stapreg object*

Description

Create a stapreg object

Usage

```
stapreg(object)
```

Arguments

`object`	A list provided by one of the `stap_*` modeling functions.

Value

A stanreg object

`stapreg-methods` *Methods for stapreg objects*

Description

The methods documented on this page are actually some of the least important methods defined for stapreg objects. The most important methods are documented separately, each with its own page. Links to those pages are provided in the **See Also** section, below.

Usage

```
## S3 method for class 'stapreg'
coef(object, ...)

## S3 method for class 'stapreg'
confint(object, ...)

## S3 method for class 'stapreg'
fitted(object, ...)

## S3 method for class 'stapreg'
nobs(object, ...)

## S3 method for class 'stapreg'
nstap(object)

## S3 method for class 'stapreg'
ntap(object)

## S3 method for class 'stapreg'
nsap(object)

## S3 method for class 'stapreg'
nfix(object, ...)

## S3 method for class 'stapreg'
residuals(object, ...)

## S3 method for class 'stapreg'
se(object, ...)

## S3 method for class 'stapreg'
vcov(object, correlation = FALSE, ...)

## S3 method for class 'stapreg'
fixef(object, ...)

## S3 method for class 'stapreg'
ngrps(object, ...)

## S3 method for class 'stapreg'
ranef(object, ...)

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

## S3 method for class 'stapreg'
VarCorr(x, sigma = 1, ...)
```

Arguments

object, x	A fitted model object returned by one of the **rstap** modeling functions. See stapreg-objects.
...	Ignored
correlation	For vcov, if FALSE (the default) the covariance matrix is returned. If TRUE, the correlation matrix is returned instead.
sigma	Ignored (included for compatibility with VarCorr).

Details

The methods documented on this page are similar to the methods defined for objects of class 'lm', 'glm', 'glmer', etc. However there are a few key differences:

residuals Residuals are *always* of type "response" (not "deviance" residuals or any other type).

coef Medians are used for point estimates. See the *Point estimates* section in print.stapreg for more details.

se The se function returns standard errors based on mad. See the *Uncertainty estimates* section in print.stapreg for more details.

confint confint will throw an error because the posterior_interval function should be used to compute Bayesian uncertainty intervals.

See Also

- The print, summary, and prior_summary methods for stapreg objects for information on the fitted model.
- The plot method to plot estimates and diagnostics.
- The posterior_predict and predictive_error methods for predictions and predictive errors - can be used for posterior predictive checks.
- The posterior_interval and predictive_interval methods for uncertainty intervals for model parameters and predictions.
- log_lik method for computing the log-likelihood of (possibly new) data.
- The as.matrix, as.data.frame, and as.array methods to access posterior draws.

---

stapreg-objects *Fitted model objects*

Description

The **rstap** model-fitting functions return an object of class 'stapreg', which is a list containing at a minimum the components listed below. Each stapreg object will also have additional classes (e.g. 'glm')

Elements for `stapreg` objects

`coefficients` Point estimates, as described in `print.stapreg`.

`ses` Standard errors based on `mad`, as described in `print.stapreg`.

`residuals` Residuals of type `'response'`.

`fitted.values` Fitted mean values. For GLMs the linear predictors are transformed by the inverse link function.

`linear.predictors` Linear fit on the link scale. For linear models this is the same as `fitted.values`.

`covmat` Variance-covariance matrix for the coefficients based on draws from the posterior distribution.

`model,x,y,z` If requested, the latent samples, model frame, model matrix and response variable used, respectively.

`family` The `family` object used.

`call` The matched call.

`formula` The model `formula`.

`data,offset,weights` The `data`, `offset`, and `weights` arguments.

`prior.info` A list with information about the prior distributions used.

`stapfit,stan_summary` The object of `stanfit-class` returned by RStan and a matrix of various summary statistics from the stanfit object.

`rstan_version` The version of the **rstan** package that was used to fit the model.

See Also

`stapreg-methods`

`stap_data` *Create a stap_data object*

Description

Create a stap_data object

Usage

```
stap_data(object)
```

Arguments

`object`	a named list of objects containing information about the staps for a given model

Value

an object of class "stap_data"

stap_glm *Bayesian generalized spatial-temporal aggregated predictor(STAP) models via Stan*

Description

Generalized linear modeling with spatial temporal aggregated predictors using prior distributions for the coefficients, intercept, spatial-temporal scales, and auxiliary parameters.

Usage

```
stap_glm(formula, family = gaussian(), subject_data = NULL,
  distance_data = NULL, time_data = NULL, subject_ID = NULL,
  max_distance = NULL, max_time = NULL, weights, offset = NULL,
  model = TRUE, y = TRUE, contrasts = NULL, ..., prior = normal(),
  prior_intercept = normal(), prior_stap = normal(),
  prior_theta = log_normal(location = 1L, scale = 1L),
  prior_aux = exponential(), adapt_delta = NULL)

stap_lm(formula, family = gaussian(), subject_data = NULL,
  distance_data = NULL, time_data = NULL, subject_ID = NULL,
  max_distance = NULL, max_time = NULL, weights, offset = NULL,
  model = TRUE, y = TRUE, contrasts = NULL, ..., prior = normal(),
  prior_intercept = normal(), prior_stap = normal(),
  prior_theta = log_normal(location = 1L, scale = 1L),
  prior_aux = exponential(), adapt_delta = NULL)
```

Arguments

formula	Same as for glm. Note that in-formula transformations will not be passed ot the final design matrix. Covariates that have "scale" in their name are not advised as this text is parsed for in the final model fit.
family	Same as glm for gaussian, binomial, and poisson families.
subject_data	a data.frame that contains data specific to the subject or subjects on whom the outcome is measured. Must contain one column that has the subject_ID on which to join the distance and time_data
distance_data	a (minimum) three column data.frame that contains (1) an id_key (2) The sap/tap/stap features and (3) the distances between subject with a given id and the built environment feature in column (2), the distance column must be the only column of type "double" and the sap/tap/stap features must be specified in the dataframe exactly as they are in the formula.
time_data	same as distance_data except with time that the subject has been exposed to the built environment feature, instead of distance
subject_ID	name of column(s) to join on between subject_data and bef_data
max_distance	the inclusion distance; upper bound for all elements of dists_crs

`max_time` inclusion time; upper bound for all elements of times_crs

`offset, weights`

Same as `glm`.

`model` logical denoting whether or not to return the fixed covariates model frame object in the fitted object

`y` In `stap_glm`, logical scalar indicating whether to return the response vector. In `stan_glm.fit`, a response vector.

`contrasts` Same as `glm`, but rarely specified.

`...` Further arguments passed to the function in the **rstap** to specify `iter`, `chains`, `cores`, `refresh`, etc.

`prior` The prior distribution for the regression coefficients. `prior` should be a call to one of the various functions provided by **rstap** for specifying priors. The subset of these functions that can be used for the prior on the coefficients can be grouped into several "families":

Family	**Functions**
Student t family	`normal`, `student_t`, `cauchy`
Hierarchical shrinkage family	`hs`, `hs_plus`
Laplace family	`laplace`, `lasso`
Product normal family	`product_normal`

See the priors help page for details on the families and how to specify the arguments for all of the functions in the table above. To omit a prior —i.e., to use a flat (improper) uniform prior— `prior` can be set to `NULL`, although this is rarely a good idea.

Note: If `prior` is from the Student t family or Laplace family, and if the `autoscale` argument to the function used to specify the prior (e.g. `normal`) is left at its default and recommended value of `TRUE`, then the default or user-specified prior scale(s) may be adjusted internally based on the scales of the predictors. See the priors help page and the *Prior Distributions* vignette for details on the rescaling and the `prior_summary` function for a summary of the priors used for a particular model.

`prior_intercept`

The prior distribution for the intercept. `prior_intercept` can be a call to `normal`, `student_t` or `cauchy`. See the priors help page for details on these functions. To omit a prior on the intercept —i.e., to use a flat (improper) uniform prior— `prior_intercept` can be set to `NULL`.

Note: The prior distribution for the intercept is set so it applies to the value *when all predictors are centered*. If you prefer to specify a prior on the intercept without the predictors being auto-centered, then you have to omit the intercept from the `formula` and include a column of ones as a predictor, in which case some element of `prior` specifies the prior on it, rather than `prior_intercept`. Regardless of how `prior_intercept` is specified, the reported *estimates* of the intercept always correspond to a parameterization without centered predictors (i.e., same as in `glm`).

prior_stap	prior for spatial-temporal aggregated predictors. Note that prior is set on the standardized latent covariates.
prior_theta	prior for the spatial-temporal aggregated predictors' scale. Can either be a single prior or a prior nested within a list of lists.
prior_aux	The prior distribution for the "auxiliary" parameter (if applicable). The "auxiliary" parameter refers to a different parameter depending on the family. For Gaussian models prior_aux controls "sigma", the error standard deviation. For negative binomial models prior_aux controls "reciprocal_dispersion", which is similar to the "size" parameter of rnbinom: smaller values of "reciprocal_dispersion" correspond to greater dispersion. For gamma models prior_aux sets the prior on to the "shape" parameter (see e.g., rgamma), and for inverse-Gaussian models it is the so-called "lambda" parameter (which is essentially the reciprocal of a scale parameter). Binomial and Poisson models do not have auxiliary parameters. prior_aux can be a call to exponential to use an exponential distribution, or normal, student_t or cauchy, which results in a half-normal, half-t, or half-Cauchy prior. See priors for details on these functions. To omit a prior —i.e., to use a flat (improper) uniform prior— set prior_aux to NULL.
adapt_delta	See the adapt_delta help page for details.

Details

The stap_glm function is similar in syntax to stan_glm except instead of performing full bayesian inference for a generalized linear model stap_glm incorporates spatial-temporal covariates

Value

A stapreg object is returned for stap_glm.

A stapfit object (or a slightly modified stapfit object) is returned if stan_glm.fit is called directly.

References

Gelman, A. and Hill, J. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press, Cambridge, UK.

Muth, C., Oravecz, Z., and Gabry, J. (2018) User-friendly Bayesian regression modeling: A tutorial with rstanarm and shinystan. *The Quantitative Methods for Psychology*. 14(2), 99–119. https://www.tqmp.org/RegularArticles/vol14-2/p099/p099.pdf

See Also

stapreg-methods and glm.

The various vignettes for stap_glm at https://biostatistics4socialimpact.github.io/rstap/articles and the preprint article.

Examples

```
fit_glm <- stap_glm(formula = y ~ sex + sap(Fast_Food),
                  subject_data = homog_subject_data[1:100,], # for speed of example only
                    distance_data = homog_distance_data,
                    family = gaussian(link = 'identity'),
                    subject_ID = 'subj_id',
                    prior = normal(location = 0, scale = 5, autoscale = FALSE),
                  prior_intercept = normal(location = 25, scale = 5, autoscale = FALSE),
                    prior_stap = normal(location = 0, scale = 3, autoscale = FALSE),
                    prior_theta = log_normal(location = 1, scale = 1),
                    prior_aux = cauchy(location = 0,scale = 5),
                    max_distance = max(homog_distance_data$Distance),
                    chains = 1, iter = 300, # for speed of example only
                    refresh = -1, verbose = FALSE)
```

`stap_glm.fit` *Fitting Generalized Linear STAP models*

Description

Fitting Generalized Linear STAP models

Usage

```
stap_glm.fit(y, z, dists_crs, u_s, times_crs, u_t, weight_functions,
  stap_data, max_distance = max(dists_crs), max_time = max(times_crs),
  weights = rep(1, NROW(y)), offset = rep(0, NROW(y)),
  family = stats::gaussian(), ..., prior = normal(),
  prior_intercept = normal(), prior_stap = normal(), group = list(),
  prior_theta = list(theta_one = normal()), prior_aux = cauchy(location
  = 0L, scale = 5L), adapt_delta = NULL)
```

Arguments

`y`	n length vector or n x 2 matrix of outcomes
`z`	n x p design matrix of subject specific covariates
`dists_crs`	(q_s+q_st) x M matrix of distances between outcome observations and built environment features with a hypothesized spatial scale
`u_s`	n x (q *2) matrix of compressed row storage array indices for dists_crs
`times_crs`	(q_t+q_st) x M matrix of times where the outcome observations were exposed to the built environment features with a hypothesized temporal scale
`u_t`	n x (q*2) matrix of compressed row storage array indices for times_crs
`weight_functions`	a Q x 2 matrix with integers coding the appropriate weight function for each STAP

stap_data	object of class "stap_data" that contains information on all the spatial-temporal predictors in the model
max_distance	the upper bound on any and all distances included in the model
max_time	the upper bound on any and all times included in the model
weights	weights to be added to the likelihood observation for a given subject
offset	offset term to be added to the outcome for a given subject
family	distributional family - only binomial gaussian or poisson currently allowed
...	optional arguments passed to the sampler - e.g. iter,warmup, etc.
prior, prior_intercept, prior_stap, prior_theta, prior_aux	see stap_glm for more information
group	list of of group terms from lme4::glmod
adapt_delta	See the adapt_delta help page for details.

---

stap_glmer *Bayesian spatial-temporal generalized linear models with group-specific terms via Stan*

Description

Bayesian inference for stap-glms with group-specific coefficients that have unknown covariance matrices with flexible priors.

Usage

```
stap_glmer(formula, family = gaussian(), subject_data = NULL,
  distance_data = NULL, time_data = NULL, subject_ID = NULL,
  group_ID = NULL, max_distance = NULL, max_time = NULL, weights,
  offset, contrasts = NULL, ..., prior = normal(),
  prior_intercept = normal(), prior_stap = normal(),
  prior_theta = log_normal(location = 1L, scale = 1L),
  prior_aux = exponential(), prior_covariance = decov(),
  adapt_delta = NULL)

stap_lmer(formula, subject_data = NULL, distance_data = NULL,
  time_data = NULL, subject_ID = NULL, group_ID = NULL,
  max_distance = NULL, max_time = NULL, weights, offset,
  contrasts = NULL, ..., prior = normal(),
  prior_intercept = normal(), prior_stap = normal(),
  prior_theta = log_normal(location = 1L, scale = 1L),
  prior_aux = exponential(), prior_covariance = decov(),
  adapt_delta = NULL)
```

Arguments

formula	Same as for glmer. Note that in-formula transformations will not be passed to the final design matrix.Covariates that have "scale" in their name are not advised as this text is parsed for in the final model fit.
family	Same as for glmer except limited to gaussian, binomial and poisson
subject_data	a data.frame that contains data specific to the subject or subjects on whom the outcome is measured. Must contain one column that has the subject_ID on which to join the distance and time_data
distance_data	a (minimum) three column data.frame that contains (1) an id_key (2) The sap/tap/stap features and (3) the distances between subject with a given id and the built environment feature in column (2), the distance column must be the only column of type "double" and the sap/tap/stap features must be specified in the dataframe exactly as they are in the formula.
time_data	same as distance_data except with time that the subject has been exposed to the built environment feature, instead of distance
subject_ID	name of column to join on between subject_data and bef_data
group_ID	name of column to join on between subject_data and bef_data that uniquely identifies the groups
max_distance	the upper bound on any and all distances included in the model
max_time	the upper bound on any and all times included in the model
weights, offset	Same as glm.
contrasts	Same as glm, but rarely specified.
...	For stap_glmer, further arguments passed to sampling (e.g. iter, chains, cores, etc.). For stap_lmer ... should also contain all relevant arguments to pass to stap_glmer (except family).
prior	The prior distribution for the regression coefficients. prior should be a call to one of the various functions provided by **rstap** for specifying priors. The subset of these functions that can be used for the prior on the coefficients can be grouped into several "families":

Family	Functions
Student t family	normal, student_t, cauchy
Hierarchical shrinkage family	hs, hs_plus
Laplace family	laplace, lasso
Product normal family	product_normal

See the priors help page for details on the families and how to specify the arguments for all of the functions in the table above. To omit a prior —i.e., to use a flat (improper) uniform prior— prior can be set to NULL, although this is rarely a good idea.

Note: If prior is from the Student t family or Laplace family, and if the autoscale argument to the function used to specify the prior (e.g. normal) is left at its de-

fault and recommended value of TRUE, then the default or user-specified prior scale(s) may be adjusted internally based on the scales of the predictors. See the priors help page and the *Prior Distributions* vignette for details on the rescaling and the prior_summary function for a summary of the priors used for a particular model.

prior_intercept

The prior distribution for the intercept. prior_intercept can be a call to normal, student_t or cauchy. See the priors help page for details on these functions. To omit a prior on the intercept —i.e., to use a flat (improper) uniform prior— prior_intercept can be set to NULL.

Note: The prior distribution for the intercept is set so it applies to the value *when all predictors are centered*. If you prefer to specify a prior on the intercept without the predictors being auto-centered, then you have to omit the intercept from the formula and include a column of ones as a predictor, in which case some element of prior specifies the prior on it, rather than prior_intercept. Regardless of how prior_intercept is specified, the reported *estimates* of the intercept always correspond to a parameterization without centered predictors (i.e., same as in glm).

prior_theta, prior_stap

priors for the spatial scale and spatial effect parameters, respectively

prior_aux

The prior distribution for the "auxiliary" parameter (if applicable). The "auxiliary" parameter refers to a different parameter depending on the family. For Gaussian models prior_aux controls "sigma", the error standard deviation. For negative binomial models prior_aux controls "reciprocal_dispersion", which is similar to the "size" parameter of rnbinom: smaller values of "reciprocal_dispersion" correspond to greater dispersion. For gamma models prior_aux sets the prior on to the "shape" parameter (see e.g., rgamma), and for inverse-Gaussian models it is the so-called "lambda" parameter (which is essentially the reciprocal of a scale parameter). Binomial and Poisson models do not have auxiliary parameters.

prior_aux can be a call to exponential to use an exponential distribution, or normal, student_t or cauchy, which results in a half-normal, half-t, or half-Cauchy prior. See priors for details on these functions. To omit a prior —i.e., to use a flat (improper) uniform prior— set prior_aux to NULL.

prior_covariance

Cannot be NULL; see decov for more information about the default arguments.

adapt_delta

See the adapt_delta help page for details.

Details

The stap_glmer function is similar in syntax to glmer but rather than performing (restricted) maximum likelihood estimation of generalized linear models, Bayesian estimation is performed via MCMC. The Bayesian model adds priors on the regression coefficients (in the same way as stap_glm) and priors on the terms of a decomposition of the covariance matrices of the group-specific parameters. See priors for more information about the priors.

The stap_lmer function is equivalent to stap_glmer with family = gaussian(link = "identity").

Value

A stapreg object is returned for `stap_glmer, stap_lmer`.

References

Gelman, A. and Hill, J. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models.* Cambridge University Press, Cambridge, UK.

Muth, C., Oravecz, Z., and Gabry, J. (2018) User-friendly Bayesian regression modeling: A tutorial with rstanarm and shinystan. *The Quantitative Methods for Psychology*. 14(2), 99–119. https://www.tqmp.org/RegularArticles/vol14-2/p099/p099.pdf

See Also

stapreg-methods and glmer.

The Longituinal Vignette for `stap_glmer` and the preprint article available through arXiv.

Examples

```
## Not run:
## subset to only include id, class name and distance variables
distdata <- homog_longitudinal_bef_data[,c("subj_ID","measure_ID","class","dist")]
timedata <- homog_longitudinal_bef_data[,c("subj_ID","measure_ID","class","time")]
## distance or time column must be numeric
timedata$time <- as.numeric(timedata$time)
fit <- stap_glmer(y_bern ~ centered_income +  sex + centered_age + stap(Coffee_Shop) + (1|subj_ID),
                  family = binomial(link='logit'),
                  subject_data = homog_longitudinal_subject_data,
                  distance_data = distdata,
                  time_data = timedata,
                  subject_ID = 'subj_ID',
                  group_ID = 'measure_ID',
                  prior_intercept = normal(location = 25, scale = 4, autoscale = F),
                  prior = normal(location = 0, scale = 4, autoscale=F),
                  prior_stap = normal(location = 0, scale = 4),
                 prior_theta = list(Coffee_Shop = list(spatial = log_normal(location = 1,
                                                                            scale = 1),
                                                       temporal = log_normal(location = 1,
                                                                             scale = 1))),
                  max_distance = 3, max_time = 50,
                  chains = 4, refresh = -1, verbose = FALSE,
                  iter = 1E3, cores = 1)

## End(Not run)
```

`stap_termination` *Spatial-Temporal Exposure Termination-Maximization Estimates*

Description

Spatial-Temporal Exposure Termination-Maximization Estimates

Usage

```
stap_termination(object, prob = 0.9, exposure_limit = 0.05,
  pars = NULL, regex_pars = NULL, max_value = NULL, ...)

## S3 method for class 'stapreg'
stap_termination(object, prob = 0.9,
  exposure_limit = 0.05, pars = NULL, regex_pars = NULL,
  max_value = NULL, ...)
```

Arguments

`object`	A fitted model object returned by one of the **rstap** modeling functions. See `stapreg-objects`.
`prob`	A number $p \in (0,1)$ indicating the desired probability mass to include in the intervals. The default is to report 90% intervals (`prob=0.9`) rather than the traditionally used 95% (see Details).
`exposure_limit`	A number indicating the desired amount of exposure for which the function will return an estimate of distance/time. Note that the exposure_limit corresponds to spatial exposure and 1-temporal exposure.
`pars`	An optional character vector of parameter names.
`regex_pars`	An optional character vector of regular expressions to use for parameter selection. `regex_pars` can be used in place of `pars` or in addition to `pars`. Currently, all functions that accept a `regex_pars` argument ignore it for models fit using optimization.
`max_value`	by defuault the max_distance and/or time from the model's original input will be used to calculate the upper bound of possible terminating distances/times - the max_value can be used to specify a new value for this value.
`...`	Currently ignored.

Value

A matrix with two columns and as many rows as model parameters (or the subset of parameters specified by `pars` and/or `regex_pars`). 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 will be `"5%"` and `"95%"`, respectively.

Examples

```
## Not run:
fit_glm <- stap_glm(formula = y ~ sex + sap(Fast_Food),
                    subject_data = homog_subject_data,
                      distance_data = homog_distance_data,
                      family = gaussian(link = 'identity'),
                      subject_ID = 'subj_id',
                      prior = normal(location = 0, scale = 5, autoscale = F),
                      prior_intercept = normal(location = 25, scale = 5, autoscale = F),
                      prior_stap = normal(location = 0, scale = 3, autoscale = F),
                      prior_theta = log_normal(location = 1, scale = 1),
                      prior_aux = cauchy(location = 0,scale = 5),
                      max_distance = max(homog_distance_data$Distance),
                      chains = CHAINS, iter = ITER,
                      refresh = -1,verbose = F)
terminal_points <- stap_termination(fit_glm, prob = .9, exposure_limit = 0.01)

## End(Not run)
## Not run:
fit_glm <- stap_glm(formula = y ~ sex + sap(Fast_Food),
                    subject_data = homog_subject_data,
                      distance_data = homog_distance_data,
                      family = gaussian(link = 'identity'),
                      subject_ID = 'subj_id',
                      prior = normal(location = 0, scale = 5, autoscale = F),
                      prior_intercept = normal(location = 25, scale = 5, autoscale = F),
                      prior_stap = normal(location = 0, scale = 3, autoscale = F),
                      prior_theta = log_normal(location = 1, scale = 1),
                      prior_aux = cauchy(location = 0,scale = 5),
                      max_distance = max(homog_distance_data$Distance),
                      chains = CHAINS, iter = ITER,
                      refresh = -1,verbose = F)
terminal_vals <- stap_termination(fit_glm, prob = .9, exposure_limit = 0.01)

## End(Not run)
```

`summary.stapreg` *Summary method for stapreg objects*

Description

Summaries of parameter estimates and MCMC convergence diagnostics (Monte Carlo error, effective sample size, Rhat).

Usage

```
## S3 method for class 'stapreg'
summary(object, pars = NULL, regex_pars = NULL,
```

```
  probs = NULL, waic = F, ..., digits = 1)

## S3 method for class 'summary.stapreg'
print(x, digits = max(1, attr(x,
  "print.digits")), ...)

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

Arguments

object	A fitted model object returned by one of the **rstap** modeling functions. See stapreg-objects.
pars	An optional character vector specifying a subset of parameters to display. Parameters can be specified by name or several shortcuts can be used. Using pars="beta" will restrict the displayed parameters to only the regression coefficients (without the intercept). "alpha" can also be used as a shortcut for "(Intercept)". If the model has varying intercepts and/or slopes they can be selected using pars = "varying". In addition, for stapmvreg objects there are some additional shortcuts available. Using pars = "long" will display the parameter estimates for the longitudinal submodels only (excluding group-specific pparameters, but including auxiliary parameters). Using pars = "event" will display the parameter estimates for the event submodel only, including any association parameters. Using pars = "assoc" will display only the association parameters. Using pars = "fixef" will display all fixed effects, but not the random effects or the auxiliary parameters. pars and regex_pars are set to NULL then all fixed effect regression coefficients are selected, as well as any auxiliary parameters and the log posterior. If pars is NULL all parameters are selected for a stapreg object.
regex_pars	An optional character vector of regular expressions to use for parameter selection. regex_pars can be used in place of pars or in addition to pars. Currently, all functions that accept a regex_pars argument ignore it for models fit using optimization.
probs	For models fit using MCMC, an optional numeric vector of probabilities passed to quantile.
waic	logical to determine whether waic should be calculated and printed with the summary object
...	Currently ignored.
digits	Number of digits to use for formatting numbers when printing. When calling summary, the value of digits is stored as the "print.digits" attribute of the returned object.
x	An object of class "summary.stapreg".

Value

The summary method returns an object of class "summary.stapreg", inheriting "summary.stapreg"), which is a matrix of summary statistics and diagnostics, with attributes storing information for

use by the print method. The print method for summary.stapreg or summary.stapmvreg objects is called for its side effect and just returns its input. The as.data.frame method for summary.stapreg objects converts the matrix to a data.frame, preserving row and column names but dropping the print-related attributes.

See Also

prior_summary to extract or print a summary of the priors used for a particular model.

---

validate_distancedata *Validate distance_data*

Description

Make sure that data is a data frame.

Usage

```
validate_distancedata(distance_data, max_distance)
```

Arguments

distance_data	User's distance_data argument
max_distance	upper bound on all possible distances

Value

If no error is thrown, the column index for the distance data is returned. If no distance_data is supplied NULL type returned.

---

validate_family *Check family argument*

Description

Check family argument

Usage

```
validate_family(f)
```

Arguments

f	the family argument specified by user (or default)

Value

If no error is thrown than either f itself is returned (if already a family) or the family object created from f is returned if f a string or function. Code adapted from **rstanarm**.

`validate_newdata` *Validate newsubjdata argument for posterior_predict, log_lik, etc.*

Description

Doesn't check if the correct variables are included (that's done in pp_data), just that newdata is either NULL or a data frame with no missing values. Also drops any unused dimensions in variables (e.g. a one column matrix inside a data frame is converted to a vector).

Usage

```
validate_newdata(x)
```

Arguments

`x`	User's 'newdata' argument

Value

Either NULL or a data frame

`validate_timedata` *Validate time_data*

Description

Make sure that time_data is a data frame, return time column index.

Usage

```
validate_timedata(time_data)
```

Arguments

`time_data`	User's time_data argument

Value

If no error is thrown, the index corresponding to the column holding the time data is returned. If no time_data is supplied NULL type returned.

validate_weights *Check weights argument*

Description

Check weights argument

Usage

```
validate_weights(w)
```

Arguments

w	The weights argument specified by user or the result of calling model.weights on a model frame.

Value

If no error is thrown then w is returned.

waic.stapreg *WAIC*

Description

WAIC

Usage

```
## S3 method for class 'stapreg'
waic(x)
```

Arguments

x	a stapreg object

Index