Package ‘tmbstan’

May 8, 2026

Type Package

Title MCMC Sampling from 'TMB' Model Object using 'Stan'

Version 1.1.0

Date 2026-03-20

Maintainer Kasper Kristensen <kaskr@dtu.dk>

Description Enables all 'rstan' functionality for a 'TMB' model object, in particular MCMC sampling and chain visualization. Sampling can be performed with or without Laplace approximation for the random effects. This is demonstrated in Monnahan & Kristensen (2018) <DOI:10.1371/journal.pone.0197954>.

Imports methods, Rcpp, TMB

Depends R (>= 4.5.0), rstan

LinkingTo StanHeaders, rstan, BH, Rcpp, RcppEigen, TMB, RcppParallel

Suggests RTMB

License GPL (>= 3)

BugReports https://github.com/kaskr/tmbstan/issues

NeedsCompilation yes

SystemRequirements GNU make

RoxygenNote 7.3.3

Author Kasper Kristensen [aut, cre]

Repository CRAN

Date/Publication 2026-03-20 14:40:02 UTC

Contents

tmbstan *Draw MCMC samples from a TMB model object using Stan*

Description

Draw MCMC samples from a TMB model object using Stan

Usage

```
tmbstan(
  obj,
  ...,
  lower = numeric(0),
  upper = numeric(0),
  laplace = FALSE,
  silent = TRUE,
  debug = FALSE
)
```

Arguments

obj	TMB model object.
...	Passed to rstan::sampling with some modifications - see details.
lower	Vector of lower parameter bounds.
upper	Vector of upper parameter bounds.
laplace	Apply the Laplace approximation to random subset of parameters ? The default disables the Laplace approximation.
silent	Be silent during samling ?
debug	Should not be used.

Details

tmbstan works for models with or without random effects.

By default a full Bayesian analysis is carried out, i.e. both parameters and random effects are sampled using MCMC. Models with random effects will thus have the Laplace approximation disabled. It is possible to mix the Laplace approximation with MCMC by setting laplace=TRUE. All methods provided by the rstan package can be applied to a fitted object. Get a complete list using methods(class="stanfit").

Lower and upper bounds can be set using lower and upper. The bounds can be specified in one of two ways. Either in short format, i.e. have the same length as obj$par. Remaining parameters (the random effects) are set as unbounded in this case. Otherwise the bounds must be in long format, i.e. have the same length as the full parameter vector objenvpar including the random effects. In both cases -Inf and Inf are valid components of lower and upper respectively. Note that initial values must be within the specified bounds.

The function arguments ... are passed to rstan's fitting function, see ?rstan::sampling. A few notable arguments are:

- `chains` The number of chains.
- `iter` The number of iterations.
- `init` Initial values for the sampler. Behaves like `rstan` with some additions:
  - Default is `"random"` - see `?stan`.
  - Special values `0` and `"0"` are allowed - see `?stan`.
  - Additional special characters `"par"` and `"last.par.best"` are allowed and will be looked up in the TMB model object. The value `"par"` signifies to start from the defaults of the model object. If an optimization has been carried out, the intial value `"last.par.best"` will start from the MLE.
  - We also allow to pass a single numeric vector, or a list of numeric vectors. List length must match the number of chains. Vector lengths must match the number of sampled parameters. Names are currently ignored.
  - Parameters that do not follow the previous scheme (e.g. characters) are passed on to `rstan` unchanged. If in doubt, use `rstan::get_inits` to inspect the applied initial values.
- `seed` Random seed.

Value

Object of class `stanfit`

Examples

```
if (requireNamespace("RTMB")) {
   func <- function(parms) {
      -sum(RTMB::dnorm(rivers, parms$mu, exp(parms$logsd), log=TRUE))
   }
   obj <- RTMB::MakeADFun(func, parameters=list(mu=0, logsd=0))
   fit <- tmbstan(obj, chains=1)
   class(fit)  ## "stanfit"
   ## The available methods are
   methods(class="stanfit")
   ## Trace plot
   traceplot(fit, pars=names(obj$par), inc_warmup=TRUE)
}
## Not run:
## Pairs plot
pairs(fit, pars=names(obj$par))

## End(Not run)
```

Index