Package 'Rlgt2'

December 1, 2017

Type Package

Title LGT package

Version 0.0-1

URL TODO

Date

Author Slawek Smyl, Christoph Bergmeir

Maintainer Christoph Bergmeir <christoph.bergmeir@gmail.com>

Description This package is based on code from Slawek Smyl to implement LGT, an exponential smoothing forecasting method using Rstan for model fitting.

License GPL-3

Depends R (>= 3.0.2), Rcpp (>= 0.12.8), methods

Imports rstan (>= 2.13.2), sn, forecast, rstantools

LinkingTo StanHeaders (>= 2.13.1), rstan (>= 2.13.2), BH (>= 1.62.0.1), Rcpp (>= 0.12.8), RcppEigen (>= 0.3.2.9.0)

RoxygenNote 6.0.1

NeedsCompilation yes

R topics documented:

Rlgt2-package *Getting started with the Rlgt2 package*

Description

An implementation of LGT and SGT models as described in ...

Author(s)

Slawek Smyl <slaweks@hotmail.co.uk>

Christoph Bergmeir <christoph.bergmeir@gmail.com>

Examples

```
x <- 1
```

fit.lgt *Runs the model fitting*

Description

Runs the model fitting

Usage

```
fit.lgt(y, model = c("LGT", "SGT", "LGTe", "SGTe", "Trend"),
  control = lgt.control(), nChains = 2, nCores = 2, addJitter = TRUE,
  verbose = FALSE)
```

Arguments

y	the time series
model	a stan model
control	control arguments list
nChains	number of MCMC chains . Must >=1. Perhaps optimal number is 4.
nCores	number of cores to be used. For performance reasons it should be equal to nChains, but nChains should be smaller or equal to the number of cores on the computer.
addJitter	adding a bit of jitter is helping Stan in case of some flat series
verbose	print verbose information yes/no

Value

lgtModel

`forecast.lgt` *produce forecasts*

Description

This function produces forecasts from a model

Usage

```
## S3 method for class 'lgt'
forecast(object, h = ifelse(frequency(object$x) > 1, 2 *
  frequency(object$x), 10), level = c(80, 95), NUM_OF_TRIALS = 2000,
  MIN_VAL = 0.001, MAX_VAL = 1e+38, ...)
```

Arguments

`object`	lgt object
`h`	Forecasting horizon (10 for annual and 2*periods otherwise)
`level`	Confidence levels for prediction intervals a.k.a. coverage percentiles. Beween 0 and 100.
`NUM_OF_TRIALS`	Number of simulations to run. Suggested rannge (1000,5000), but it may have to be higher for good coverage of very high levels, e.g. 99.8.
`MIN_VAL`	Minimum value the forecast can take. Must be positive.
`MAX_VAL`	Maximum value the forecast can take.
`...`	description

Value

returns a forecast object compatible with the forecast package

Author(s)

bergmeir

`initModel` *Initialize a non-seasonal LGT stan model*

Description

Initialize a stan model that uses the (non-seasonal) LGT

Usage

```
initModel(modelType = NULL)
```

Arguments

modelType	type of the forecasting model selected

Value

SkeletonModel

---

lgt *lgt class*

Description

a constructor function for the "lgt" class

Usage

```
lgt(y, lgtmodel, params, paramMean, seasonality, samples)
```

Arguments

y	the time series data
lgtmodel	type of lgtmodel selected
params	list of parameters
paramMean	mean of each parameter
seasonality	number of seasons, 1 for annual
samples	stanfit object representing the MCMC samples

Value

lgt instance

---

lgt.control *Sets and initializes the main parameters of the algorithm*

Description

This is a function that initializes and sets the parameters of the algorithm. It generates a list of parameters, to be used with the fit.lgt function.

Usage

```
lgt.control(MAX_RHAT_ALLOWED = 1.005, NUM_OF_ITER = 2500,
  MAX_NUM_OF_REPEATS = 3, CAUCHY_SD_DIV = 200, MIN_SIGMA = 0.001,
  MIN_NU = 2, MAX_NU = 20, MIN_POW_TREND = -0.5, MAX_POW_TREND = 1,
  POW_TREND_ALPHA = 1, POW_TREND_BETA = 1, POW_SIGMA_ALPHA = 1,
  POW_SIGMA_BETA = 1, ADAPT_DELTA = 0.9, MAX_TREE_DEPTH = 11,
  SEASONALITY = 1, SKEW = 0)
```

Arguments

`MAX_RHAT_ALLOWED`
: Maximum average Rhat that suggests a good fit, see Stan's manual. Suggested range(1.005,1.02), see also MAX_NUM_OF_REPEATS description below.

`NUM_OF_ITER`
: Number of iterations for each chain. Suggested range(1000,5000). Generally, the longer the series, the smaller the vallue will do. See also MAX_NUM_OF_REPEATS description below.

`MAX_NUM_OF_REPEATS`
: Maximum number of the sampling procedure repeats if the fit is unsatisfactorily (avgRHat>MAX_RHAT_ALLOWED). Each round doubles the number of iterations. Suggested range(2,4)

`CAUCHY_SD_DIV`
: For parameters with non-obvious range Cauchy distribution is used. The error size of this distribution is calculated by dividing max value of the time series by this constant. Suggested range(100,300)

`MIN_SIGMA`
: Minimum size of the fitted sigma, applied for numerical stability. Must bve positive.

`MIN_NU`
: Minimum degrees of freedom of the Student's distribution, that is used in most models. Suggested range(1.2, 5)

`MAX_NU`
: Maximum degrees of freedom of the Student's distribution. Suggested range(15,30)

`MIN_POW_TREND`
: Minimum value of power of trend coefficient. Suggested range(-1,0)

`MAX_POW_TREND`
: Maximum value of power of trend coefficient. It should stay 1 to allow the model to approach exponential growth when needed.

`POW_TREND_ALPHA`
: Alpha parameter of Beta distribution that is the prior of the power coefficient in the formula of trend parameter. To make the forecast more curved, make it larger. Suggested range(1,6)

`POW_TREND_BETA`
: Beta parameter of Beta distribution that is the prior of the power of trend parameter. 1 by default, see also above.

`POW_SIGMA_ALPHA`
: Alpha parameter of Beta distribution that is the prior of the power coefficient in the formula of the error size. 1 by default, see also below.

`POW_SIGMA_BETA`
: Beta parameter of Beta distribution that is the prior of the power coefficient in the formula of the error size. If the powSigma fitted is considered too often too high (i.e.> 0.7) you can attempt to tame it down by increasing POW_SIGMA_BETA. Suggested range(1,4). ADAPT_DELTA Target Metropolis acceptance rate. See Stan manual. Suggested range (0.8-0.97). MAX_TREE_DEPTH NUTS maximum tree depth. See Stan manual. Suggested range (10-12).

ADAPT_DELTA	Description Setting it negative makes negative innovations having smaller impact on the fitting than the positive ones, which would have the effect of making a model "more optimistic". Suggested range (-0.5, 0.5).
MAX_TREE_DEPTH	Description
SEASONALITY	E.g. 12 for monthly seasonality. 1 for non-seasonal models
SKEW	Skew of error distribution used by manually-skewed models. 0 be default.

`posterior_interval.lgt`

Posterior uncertainty intervals

Description

For models fit using MCMC (`algorithm="sampling"`) or one of the variational approximations (`"meanfield"` or `"fullrank"`), the `posterior_interval` function computes Bayesian posterior uncertainty intervals. These intervals are often referred to as *credible* intervals, but we use the term *uncertainty* intervals to highlight the fact that wider intervals correspond to greater uncertainty.

Usage

```
## S3 method for class 'lgt'
posterior_interval(object, prob = 0.9, type = "central", ...)
```

Arguments

`object`	an lgt object
`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 $\alpha = 1 - p$.
`...`	additional params (currently not being used)

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 **rstanarm** 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.

`print.lgt` *Generic print function for lgt models*

Description

Print out some characteristics of a `lgt` model.

Usage

```
## S3 method for class 'lgt'
print(x, ...)
```

Arguments

`x`	the `lgt` model
`...`	additional function parameters (currently not used)

Index