

# Package: bayesforecast (via r-universe)

October 24, 2024

**Title** Bayesian Time Series Modeling with Stan

**Version** 1.0.1

**Description** Fit Bayesian time series models using 'Stan' for full Bayesian inference. A wide range of distributions and models are supported, allowing users to fit Seasonal ARIMA, ARIMAX, Dynamic Harmonic Regression, GARCH, t-student innovation GARCH models, asymmetric GARCH, Random Walks, stochastic volatility models for univariate time series. Prior specifications are flexible and explicitly encourage users to apply prior distributions that actually reflect their beliefs. Model fit can easily be assessed and compared with typical visualization methods, information criteria such as loglik, AIC, BIC WAIC, Bayes factor and leave-one-out cross-validation methods. References: Hyndman (2017) <[doi:10.18637/jss.v027.i03](https://doi.org/10.18637/jss.v027.i03)>; Carpenter et al. (2017) <[doi:10.18637/jss.v076.i01](https://doi.org/10.18637/jss.v076.i01)>.

**License** GPL-2

**Encoding** UTF-8

**LazyData** true

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.1.1

**Biarch** true

**Depends** R (>= 4.0.0)

**Imports** bayesplot (>= 1.5.0), methods, gridExtra, ggplot2, forecast, loo (>= 2.2.0), Rcpp (>= 0.12.0), rstan (>= 2.26.0), rstantools (>= 2.0.0), RcppParallel (>= 5.0.1), bridgesampling (>= 0.3-0), MASS, StanHeaders (>= 2.26.0), astsa, lubridate, prophet, zoo

**Suggests** knitr, rmarkdown, ggfortify

**LinkingTo** BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), RcppEigen (>= 0.3.3.3.0), rstan (>= 2.26.0), StanHeaders (>= 2.26.0)

**SystemRequirements** GNU make

**Collate** 'autoplot.R' 'auto\_sarima.R' 'bayes\_factor.R'  
 'bayesforecast-package.R' 'Birth.R' 'Fit.R' 'forecast.R'  
 'garch.R' 'get\_params.R' 'log\_lik.R' 'Misc.R' 'model.R'  
 'naive.R' 'posterior\_intervals.R' 'posterior\_predict.R' 'ets.R'  
 'predictive\_error.R' 'print.R' 'prior.R' 'report.R' 'Sarima.R'  
 'summary.R' 'ssm.R' 'stanmodels.R' 'SVM.R' 'varstan.R'  
 'stan\_models.R'

**NeedsCompilation** yes

**Author** Asael Alonzo Matamoros [aut, cre], Cristian Cruz Torres [aut],  
 Andres Dala [ctb], Rob Hyndman [ctb], Mitchell O'Hara-Wild  
 [ctb]

**Maintainer** Asael Alonzo Matamoros <asael.alonzo@gmail.com>

**VignetteBuilder** knitr

**Repository** <https://asael697.r-universe.dev>

**RemoteUrl** <https://github.com/asael697/bayesforecast>

**RemoteRef** HEAD

**RemoteSha** 9909f78d7f676d936636356c43f819cedfbfff0a

## Contents

bayesforecast-package . . . . .	4
aic . . . . .	5
AICc . . . . .	5
air . . . . .	6
as.stan . . . . .	7
aust . . . . .	8
auto.sarima . . . . .	8
autoplot.ts . . . . .	11
autoplot.varstan . . . . .	12
bayes_factor.varstan . . . . .	13
beta . . . . .	14
bic . . . . .	15
birth . . . . .	16
bridge_sampler.varstan . . . . .	16
cauchy . . . . .	17
check_residuals . . . . .	18
chisq . . . . .	19
demgbp . . . . .	19
exponential . . . . .	20
extract_stan . . . . .	20
fitted.varstan . . . . .	21
forecast.varstan . . . . .	22
fourier . . . . .	24
gamma . . . . .	25
garch . . . . .	26

get_parameters . . . . .	28
get_prior . . . . .	28
ggacf . . . . .	29
gghist . . . . .	30
ggnorm . . . . .	31
ggpacf . . . . .	31
Holt . . . . .	32
Hw . . . . .	33
inverse.chisq . . . . .	35
inverse.gamma . . . . .	36
ipc . . . . .	36
jeffrey . . . . .	37
laplace . . . . .	37
LKJ . . . . .	38
LocalLevel . . . . .	38
loglik . . . . .	40
log_lik.varstan . . . . .	41
loo.varstan . . . . .	42
mcmc_plot.varstan . . . . .	43
model . . . . .	44
naive . . . . .	45
normal . . . . .	46
oildata . . . . .	47
plot.varstan . . . . .	47
posterior_epred.varstan . . . . .	48
posterior_interval . . . . .	49
posterior_predict.varstan . . . . .	50
predictive_error.varstan . . . . .	51
print.garch . . . . .	52
print.Holt . . . . .	53
print.Hw . . . . .	53
print.LocalLevel . . . . .	54
print.naive . . . . .	54
print.Sarima . . . . .	55
print.ssm . . . . .	55
print.SVM . . . . .	56
print.varstan . . . . .	56
prior_summary.varstan . . . . .	57
report . . . . .	58
residuals.varstan . . . . .	59
Sarima . . . . .	59
set_prior . . . . .	61
ssm . . . . .	62
stan_garch . . . . .	64
stan_Holt . . . . .	67
stan_Hw . . . . .	70
stan_LocalLevel . . . . .	73
stan_naive . . . . .	75

stan_sarima . . . . .	78
stan_ssm . . . . .	81
stan_SVM . . . . .	84
student . . . . .	86
summary.varstan . . . . .	87
SVM . . . . .	87
uniform . . . . .	89
varstan . . . . .	89
waic.varstan . . . . .	91

<b>Index</b>	<b>93</b>
--------------	-----------

---

bayesforecast-package *Bayesian Time Series Modeling with Stan.*

---

## Description

Fit univariate time series models using 'Stan' for full Bayesian inference. A wide range of distributions and models are supported, allowing users to fit Seasonal ARIMA, ARIMAX, Dynamic Harmonic Regression, GARCH, t-student innovation GARCH models, asymmetric GARCH, Random Walks, and stochastic volatility models. Prior specifications are flexible and explicitly encourage users to apply prior distributions that actually reflect their beliefs. Model fit can easily be assessed and compared with typical visualization methods, information criteria such as loglik, AIC, BIC WAIC, Bayes factor and leave-one-out cross-validation methods.

## References

- Carpenter, B. and Gelman, A. and Hoffman, D. and Lee, D. and Goodrich, B. and Betancourt, M. and Brubaker, and Guo, L. and Riddell. 2017. Stan: A probabilistic programming language. *Journal of Statistical Software* 76(1). doi: 10.18637/jss.v076.i01.
- Stan Development Team. (2018). Stan Modeling Language Users Guide and Reference Manual, Version 2.18.0. url: <https://mc-stan.org>.
- Hyndman, R. & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*. 26(3), 1-22. doi: 10.18637/jss.v027.i03.
- Tsay, R (2010). Analysis of Financial Time Series. *Wiley-Interscience*. 978-0470414354, second edition.
- Shumway, R.H. and Stoffer, D.S. (2010). Time Series Analysis and Its Applications: With R Examples. *Springer Texts in Statistics*. isbn: 9781441978646. First edition.

---

aic	<i>Computes posterior sample of the pointwise AIC method from a varstan object</i>
-----	--

---

**Description**

Convenience function for computing the pointwise Akaike Information Criteria method from a varstan object.

**Usage**

```
aic(x)
```

**Arguments**

x                    A varstan object of the time series fitted model.

**Value**

A numeric array of size R, containing the posterior samples of the AICc for a varstan object, where R is the number of iterations. If multiple chains are fitted, then the array is of length M\*R, where M is the number of chains

**Author(s)**

Asael Alonzo Matamoros

**Examples**

```
library(astsa)
model = Sarima(birth,order = c(0,1,2),seasonal = c(1,1,1))
fit1 = varstan(model,iter = 500,chains = 1)

aic1 = aic(fit1)
mean(aic1)
```

---

AICc	<i>Computes posterior sample of the pointwise corrected AIC method from a varstan object</i>
------	--

---

**Description**

Convenience function for computing the pointwise corrected Akaike Information Criteria method from a varstan object.

**Usage**`AICc(x)`**Arguments**

`x` A varstan object of the time series fitted model.

**Value**

A numeric array of size R, containing the posterior samples of the AICc for a varstan object, where R is the number of iterations. If multiple chains are fitted, then the array is of length M\*R, where m is the number of chains

**Author(s)**

Asael Alonzo Matamoros

**Examples**

```
library(astsa)
model = Sarima(birth,order = c(0,1,2),seasonal = c(1,1,1))
fit1 = varstan(model,iter = 500,chains = 1)

aic1 = AICc(fit1)
mean(aic1)
```

---

air

*Air Transport Passengers Australia*

---

**Description**

Total annual air passengers (in millions) including domestic and international aircraft passengers of air carriers registered in Australia. 1970-2016.

**Usage**`air`**Format**

the format is: Annual Time-Series (1:27) from 1990 to 2016:

**Source**`fpp2`

## References

Hyndman, R. & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*. 26(3), 1-22.doi: 10.18637/jss.v027.i03

---

as.stan	<i>Convert to a stanfit object.</i>
---------	-------------------------------------

---

## Description

Convert a varstan object to a stanfit object of the **rstan** package.

## Usage

```
as.stan(object)
```

## Arguments

object            a varstan object.

## Value

a stanfit object.

## Author(s)

Asael Alonzo Matamoros

## Examples

```
# Fitting a GARCH(1,1) model
dat = garch(ipc,order = c(1,1,0))
fit1 = varstan(dat,iter = 500,chains = 1)

# Converting to a Stanfit object
stanfit1 = as.stan(fit1)
```

---

 aust

---

*International Tourists to Australia: Total visitor nights.*


---

### Description

Quarterly visitor nights (in millions) spent by international tourists to Australia. 1999-2015

### Usage

```
aust
```

### Format

the format is: Quarterly Time-Series (1:44) from 1999 to 2015:

### Source

**fpp2**

### References

Hyndman, R. & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*. 26(3), 1-22.doi: 10.18637/jss.v027.i03

---

 auto.sarima

---

*Automatic estimate of a Seasonal ARIMA model*


---

### Description

Returns the best seasonal ARIMA model using a bic value, this function the `auto.arima` function of the **forecast** package to select the seasonal ARIMA model and estimates the model using a HMC sampler.

### Usage

```
auto.sarima(ts,seasonal = TRUE,xreg = NULL,chains=4,iter=4000,warmup=floor(iter/2),
  adapt.delta = 0.9,tree.depth =10,stepwise = TRUE, series.name = NULL,
  prior_mu0 = NULL,prior_sigma0 = NULL,prior_ar = NULL, prior_ma = NULL,
  prior_sar = NULL,prior_sma = NULL, prior_breg = NULL,...)
```



**Arguments**

<code>ts</code>	a numeric or <code>ts</code> object with the univariate time series.
<code>seasonal</code>	optionally, a logical value for seasonal ARIMA models. By default <code>seasonal = TRUE</code> .
<code>xreg</code>	Optionally, a numerical matrix of external regressors, which must have the same number of rows as <code>ts</code> . It should not be a data frame.
<code>chains</code>	An integer of the number of Markov Chains chains to be run, by default 4 chains are run.
<code>iter</code>	An integer of total iterations per chain including the warm-up, by default the number of iterations are 2000.
<code>warmup</code>	A positive integer specifying number of warm-up (aka burn-in) iterations. This also specifies the number of iterations used for step-size adaptation, so warm-up samples should not be used for inference. The number of warmup should not be larger than <code>iter</code> and the default is <code>iter/2</code> .
<code>adapt.delta</code>	An optional real value between 0 and 1, the thin of the jumps in a HMC method. By default is 0.9.
<code>tree.depth</code>	An integer of the maximum depth of the trees evaluated during each iteration. By default is 10.
<code>stepwise</code>	If <code>TRUE</code> , will do stepwise selection (faster). Otherwise, it searches over all models. Non-stepwise selection can be very slow, especially for seasonal models.
<code>series.name</code>	an optional string vector with the series names.
<code>prior_mu0</code>	The prior distribution for the location parameter in an ARIMA model. By default the value is set <code>NULL</code> , then the default <code>student(7,0,1)</code> prior is used.
<code>prior_sigma0</code>	The prior distribution for the scale parameter in an ARIMA model. By default the value is set <code>NULL</code> , then the default <code>student(7,0,1)</code> prior is used.
<code>prior_ar</code>	The prior distribution for the auto-regressive parameters in an ARIMA model. By default the value is set <code>NULL</code> , then the default <code>normal(0,0.5)</code> priors are used.
<code>prior_ma</code>	The prior distribution for the moving average parameters in an ARIMA model. By default the value is set <code>NULL</code> , then the default <code>normal(0,0.5)</code> priors are used.
<code>prior_sar</code>	The prior distribution for the seasonal auto-regressive parameters in a SARIMA model. By default the value is set <code>NULL</code> , then the default <code>normal(0,0.5)</code> priors are used.
<code>prior_sma</code>	The prior distribution for the seasonal moving average parameters in a SARIMA model. By default the value is set <code>NULL</code> , then the default <code>normal(0,0.5)</code> priors are used.
<code>prior_breg</code>	The prior distribution for the regression coefficient parameters in a ARIMAX model. By default the value is set <code>NULL</code> , then the default <code>student(7,0,1)</code> priors are used.
<code>...</code>	Further arguments passed to <code>auto.arima</code> function.

## Details

Automatic ARIMA model fitting implemented by Rob Hyndman, this function finds the best Seasonal ARIMA model using `bic`, and then proceeds to fit the model using `varstan` function and the default priors of a `Sarima` model constructor.

This function provides an initial model fit for beginning the Bayesian analysis of the univariate time series. For better fit and model selection try different models and other model selection criteria such as `loo` or `bayes_factor`.

The default arguments are designed for rapid estimation of models for many time series. If you are analyzing just one time series, and can afford to take some more time, it is recommended that you set `stepwise=FALSE` and reduce the number of iterations per chain (`iter`).

For more information look at `auto.arima()` function of `forecast` package.

## Value

A `varstan` object with the "best" fitted ARIMA model to the data

## Author(s)

Asael Alonzo Matamoros

## References

Hyndman, R. & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*. 26(3), 1-22. doi: 10.18637/jss.v027.i03.

Box, G. E. P. and Jenkins, G.M. (1978). Time series analysis: Forecasting and control. San Francisco: Holden-Day. *Biometrika*, 60(2), 297-303. doi:10.1093/biomet/65.2.297.

Kennedy, P. (1992). Forecasting with dynamic regression models: Alan Pankratz, 1991. *International Journal of Forecasting*. 8(4), 647-648. url: <https://EconPapers.repec.org/RePEc:eee:intfor:v:8:y:1992:i:4>

## See Also

`Sarima` `varstan`.

## Examples

```
# Automatic Sarima model for the birth data
auto.sarima(birth,iter = 500,chains = 1)

# Dynamic Harmonic regression
auto.sarima(birth,xreg = fourier(birth,K= 6),iter = 500,chains = 1)
```

---

`autoplot.ts`*Automatically create a ggplot for time series objects.*

---

## Description

`autoplot` takes an object of type `ts` or `mts` and creates a `ggplot` object suitable for usage with `stat_forecast`.

## Usage

```
## S3 method for class 'ts'
autoplot(
  object,
  series = NULL,
  xlab = "Time",
  ylab = deparse(substitute(object)),
  main = NULL,
  facets = FALSE,
  colour = TRUE,
  ...
)

## S3 method for class 'ts'
fortify(model, data, ...)
```

## Arguments

<code>object</code>	Object of class “ <code>ts</code> ” or “ <code>mts</code> ”.
<code>series</code>	Identifies the time series with a colour, which integrates well with the functionality of <code>geom_forecast</code> .
<code>xlab</code>	a string with the plot’s x axis label. By default a NULL value.
<code>ylab</code>	a string with the plot’s y axis label. By default a counts" value.
<code>main</code>	a string with the plot’s title.
<code>facets</code>	If TRUE, multiple time series will be faceted (and unless specified, colour is set to FALSE). If FALSE, each series will be assigned a colour.
<code>colour</code>	If TRUE, the time series will be assigned a colour aesthetic
<code>...</code>	Other plotting parameters to affect the plot.
<code>model</code>	Object of class “ <code>ts</code> ” to be converted to “ <code>data.frame</code> ”.
<code>data</code>	Not used (required for <code>fortify</code> method)

## Details

`fortify.ts` takes a `ts` object and converts it into a data frame (for usage with `ggplot2`).

**Value**

None. Function produces a ggplot2 graph.

**Author(s)**

Mitchell O'Hara-Wild.

**See Also**

[plot.ts](#), [fortify](#)

**Examples**

```
library(ggplot2)
autoplot(USAccDeaths)

lungDeaths <- cbind(mdeaths, fdeaths)
autoplot(lungDeaths)
autoplot(lungDeaths, facets=TRUE)
```

---

autoplot.varstan

*autoplot methods for varstan models.*

---

**Description**

Preliminary autoplot methods for varstan models only valid for univariate time series models. The function prints the fitted values time series, the trace and density plots for the sampled model parameters, or the residuals' posterior mean time series.

**Usage**

```
## S3 method for class 'varstan'
autoplot(object, prob = 0.95, ...)
```

**Arguments**

object	An object of class varstan.
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%.
...	Further arguments passed to mcmc_combo.

**Value**

None. Function produces a ggplot2 graph.

## Examples

```
library(astsa)
sf1 = auto.sarima(ts = birth,iter = 500,chains = 1)
# fitted model
autoplot(sf1)
```

---

bayes\_factor.varstan *Bayes Factors from Marginal Likelihoods.*

---

## Description

Compute Bayes factors from marginal likelihoods.

## Usage

```
## S3 method for class 'varstan'
bayes_factor(x1, x2, log = FALSE, ...)
```

## Arguments

x1	A varstan object
x2	Another varstan object based on the same data.
log	A boolean parameter for report the Bayes_factor in log scale. The default value is FALSE.
...	Additional arguments passed to bayes_factor.

## Details

The computation of marginal likelihoods based on bridge sampling requires a lot more posterior samples than usual. A good conservative rule of thumb is perhaps 10-fold more samples (read: the default of 4000 samples may not be enough in many cases). If not enough posterior samples are provided, the bridge sampling algorithm tends to be unstable leading to considerably different results each time it is run. We thus recommend running `bridge_sampler` multiple times to check the stability of the results.

For more details check the **bridgesampling** package.

## Value

The bayes factors of two models.

## Examples

```
library(astsa)
# Fitting a seasonal arima model
mod1 = Sarima(birth,order = c(0,1,2),seasonal = c(1,1,1))
fit1 = varstan(mod1,iter = 500,chains = 1)

# Fitting a Dynamic harmonic regression
mod2 = Sarima(birth,order = c(0,1,2),xreg = fourier(birth,K=6))
fit2 = varstan(mod2,iter = 500,chains = 1)

# compute the Bayes factor
bayes_factor(fit1, fit2)
```

---

beta

*Define a beta prior distribution*

---

## Description

beta(shape1,shape2)

## Usage

beta(shape1 = 2, shape2 = 2)

## Arguments

shape1            the first form parameter  
shape2            the second form parameter

## Details

Define a beta prior distribution using the hyper parameters shape1 and shape2, by default a beta(2,2) distribution is return.

## Value

a numerical vector interpreted as a prior in Stan

---

bic	<i>Computes posterior sample of the pointwise BIC method from a varstan object</i>
-----	--

---

### Description

Convenience function for computing the pointwise Bayesian Information Criteria method from a varstan object.

### Usage

```
bic(x)
```

### Arguments

x                    A varstan object of the time series fitted model.

### Value

A numeric array of size R, containing the posterior samples of the bic for a varstan object, where R is the number of iterations. If multiple chains are fitted, then the array is of length M\*R, where M is the number of chains

### Author(s)

Asael Alonzo Matamoros

### Examples

```
library(astsa)
model = Sarima(birth,order = c(0,1,2),seasonal = c(1,1,1))
fit1 = varstan(model,iter = 500,chains = 1)

bic1 = bic(fit1)
mean(bic1)
```

---

birth	<i>U.S. Monthly Live Births.</i>
-------	----------------------------------

---

**Description**

Monthly live births (adjusted) in thousands for the United States, 1948-1979.

**Usage**

birth

**Format**

the format is: Time-Series (1:373) from 1948 to 1979:

**Source**

**astsa**

**References**

<http://www.stat.pitt.edu/stoffer/tsa4/> <http://www.stat.pitt.edu/stoffer/tsda/>

---

bridge_sampler.varstan
------------------------

*Log Marginal Likelihood via Bridge Sampling.*

---

**Description**

Computes log marginal likelihood via bridge sampling, which can be used in the computation of Bayes factors and posterior model probabilities.

**Usage**

```
## S3 method for class 'varstan'  
bridge_sampler(samples, ...)
```

**Arguments**

samples	A varstan object.
...	Additional arguments passed to <code>bridge_sampler.stanfit</code> .



**Details**

The `varstan` class is just a thin wrapper that contains the `stanfit` objects.

Computing the marginal likelihood via the `bridgesampler` package for `stanfit` objects.

The computation of marginal likelihoods based on bridge sampling requires a lot more posterior samples than usual. A good conservative rule of thumb is perhaps 10-fold more samples (read: the default of 4000 samples may not be enough in many cases). If not enough posterior samples are provided, the bridge sampling algorithm tends to be unstable leading to considerably different results each time it is run. We thus recommend running `bridge_sampler` multiple times to check the stability of the results.

For more details check the **bridgesampling** package.

**Value**

the model's marginal likelihood from the `bridge_sampler` package.

**Examples**

```
library(astsa)
# Fitting a seasonal ARIMA model
mod1 = Sarima(birth,order = c(0,1,2),seasonal = c(1,1,1))
fit1 = varstan(mod1,iter = 500,chains = 1)

fit1
bridge_sampler(fit1)

# Fitting a Dynamic harmonic regression
mod2 = Sarima(birth,order = c(0,1,2),xreg = fourier(birth,K=6))
fit2 = varstan(mod2,iter = 500,chains = 1)

fit2
bridge_sampler(fit2)
```

---

cauchy

*Define a Cauchy prior distribution*

---

**Description**

`cauchy(mu,sd)`

**Usage**

`cauchy(mu = 0, sd = 1)`

**Arguments**

mu                    the location parameter mu  
sd                    the standard deviation parameter sigma

**Details**

Define a Cauchy prior distribution using the hyper parameters mu and sigma, by default a standard Cauchy(0,1) distribution is return.

**Value**

a numerical vector interpreted as a prior in Stan

---

check_residuals	<i>Visual check of residuals in a varstan object.</i>
-----------------	---

---

**Description**

Performs a visual check of residuals in time series models, this method is inspired in the `check.residuals` function provided by the `forecast` package.

**Usage**

```
check_residuals(object, ...)
```

**Arguments**

object                a varstan object.  
...                    Other plotting parameters to affect the plot.

**Value**

None. Function produces a ggplot2 graph.

**Author(s)**

Asael Alonzo Matamoros.

**Examples**

```
library(astsa)
sf1 = auto.sarima(ts = birth, iter = 500, chains = 1)
# fitted model
check_residuals(sf1)
```

---

chisq	<i>Define a chi square prior distribution</i>
-------	---

---

**Description**

chisq(df)

**Usage**

chisq(df = 7)

**Arguments**

df                    the degree freedom parameter df

**Details**

Define a gamma prior distribution using the degree freedom df hyper parameter, by default an chisq(7) distribution is return.

**Value**

a numerical vector interpreted as a prior in Stan

---

demgbp	<i>DEM/GBP exchange rate log-returns</i>
--------	--

---

**Description**

The vector dem2gbp contains daily observations of the Deutschmark vs British Pound foreign exchange rate log-returns. This data set has been promoted as an informal benchmark for GARCH time-series software validation. See McCullough and Renfro (1999), and Brooks, Burke, and Persaud (2001) for details. The nominal returns are expressed in percent as in Bollerslev and Ghysels (1996). The sample period is from January 3, 1984, to December 31, 1991, for a total of 1974 observations.

**Usage**

demgbp

**Format**

the format is: Time-Series (1:350) from 1984 to 1985:

**Source**

**bayesGARCH**

**References**

- Engle, R. (1982). Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. *Econometrica*, 50(4), 987-1007. url: <http://www.jstor.org/stable/1912773>.
- Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*. 31(3), 307-327. doi: [https://doi.org/10.1016/0304-4076\(86\)90063-1](https://doi.org/10.1016/0304-4076(86)90063-1).
- Ardia, D. and Hoogerheide, L. (2010). Bayesian Estimation of the GARCH(1,1) Model with Student-t Innovations. *The R Journal*. 2(7), 41-47. doi: 10.32614/RJ-2010-014.

---

 exponential

*Define an exponential prior distribution*


---

**Description**

exponential(rate)

**Usage**

exponential(rate = 1)

**Arguments**

rate                    the rate parameter lambda in exponential distribution

**Details**

Define a gamma prior distribution using the rate hyper parameter, by default an exponential(1) distribution is return.

**Value**

a numerical vector interpreted as a prior in Stan

---

 extract\_stan

*Extract chains of an stanfit object implemented in rstan package*


---

**Description**

Extract chains of an stanfit object implemented in rstan package

**Usage**

extract\_stan(object, pars, permuted = TRUE, inc\_warmup = FALSE, include = TRUE, ...)

**Arguments**

object	a varstan object
pars	n optional character vector providing the parameter names (or other quantity names) of interest. If not specified, all parameters and other quantities are used. The log-posterior with name lp__ is also included by default.
permuted	A logical scalar indicating whether the draws after the warmup period in each chain should be permuted and merged. If FALSE, the original order is kept. For each stanfit object, the permutation is fixed (i.e., extracting samples a second time will give the same sequence of iterations).
inc_warmup	A logical scalar indicating whether to include the warmup draws. This argument is only relevant if permuted is FALSE.
include	A logical scalar indicating whether the parameters named in pars should be included (TRUE) or excluded (FALSE).
...	Further arguments passed to extract function.

**Value**

a list with the posterior samples of the provided parameters.

**Author(s)**

Asael Alonzo Matamoros

**Examples**

```
library(astsa)
# Fitting a GARCH(1,1) model
dat = garch(ipc,order = c(1,1,0))
fit2 = varstan(dat,iter = 500,chains = 1)

# Extracting the mean parameter
mu0 = extract_stan(fit2,pars = "mu0")
```

---

fitted.varstan

*Expected Values of the Posterior Predictive Distribution*


---

**Description**

The function returns the posterior estimate of the fitted values of a varstan model, similar to the fit\_values functions of other packages.

**Usage**

```
## S3 method for class 'varstan'
fitted(object, robust = FALSE, ...)
```

**Arguments**

object	A varstan object, <a href="#">varstan</a>
robust	A boolean value, if its TRUE it returns the median of the posterior distribution, and if its FALSE it returns the mean, by default is the FALSE value
...	Further arguments passed to posterior_predict.

**Details**

This function returns a time series of the predicted *mean* response values.

**Value**

A time series (ts) of predicted *mean* response values.

**Author(s)**

Asael Alonzo Matamoros

**See Also**

[posterior\\_predict.varstan](#)

---

forecast.varstan      *Forecasting varstan objects*

---

**Description**

forecast is a generic function for forecasting from time series or varstan models. The function invokes particular *methods* which depend on the class of the first argument.

**Usage**

```
## S3 method for class 'varstan'
forecast(
  object,
  h = 10,
  probs = c(0.8, 0.9),
  xreg = NULL,
  robust = FALSE,
  draws = 1000,
  seed = NULL,
  ...
)
```

**Arguments**

object	a time series or varstan model for which forecasts are required.
h	Number of periods for forecasting.
probs	A numerical vector $p \in (0, 1)$ indicating the desired probability mass to include in the intervals. The default is to report 90% and 80% intervals ( <code>level=c(0.8, 0.9)</code> ).
xreg	Optionally, a numerical matrix of external regressors, which must have the same number of rows as <code>ts</code> . It should not be a data frame.
robust	A boolean for obtain the robust estimation. The default
draws	An integer indicating the number of draws to return. The default number of draws is 1000.
seed	An optional <a href="#">seed</a> to use.
...	Further arguments passed to <code>posterior_predict</code> .

**Details**

If `model=NULL`, the function `forecast.ts` makes forecasts using [ets](#) models (if the data are non-seasonal or the seasonal period is 12 or less) or [stlf](#) (if the seasonal period is 13 or more).

If `model` is not `NULL`, `forecast.ts` will apply the `model` to the object time series, and then generate forecasts accordingly.

**Value**

An object of class "forecast".

The function `summary` is used to obtain and print a summary of the results, while the function `plot` produces a plot of the forecasts and prediction intervals.

The generic accessors functions `fitted.values` and `residuals` extract various useful features of the value returned by `forecast$model`.

An object of class "forecast" is a list usually containing at least the following elements:

model	A list containing information about the fitted model
method	The name of the forecasting method as a character string
mean	Point forecasts as a time series
lower	Lower limits for prediction intervals
upper	Upper limits for prediction intervals
level	The confidence values associated with the prediction intervals
x	The original time series (either object itself or the time series used to create the model stored as object).
residuals	Residuals from the fitted model. For models with additive errors, the residuals will be <code>x</code> minus the fitted values.
fitted	Fitted values (one-step forecasts)

**Author(s)**

Asael Alonzo Matamoros.

**See Also**

The "forecast" methods of the forecast package.

**Examples**

```
fit = auto.sarima(ts = birth,iter = 500,chains = 1)
fc = forecast(fit,h = 12)
```

---

fourier

*Fourier terms for modeling seasonality.*


---

**Description**

fourier returns a matrix containing terms from a Fourier series, up to order K, suitable for use in [Sarima](#) or [auto.sarima](#).

**Usage**

```
fourier(x, K, h = NULL)
```

**Arguments**

x	Seasonal time series: a ts or a msts object
K	Maximum order(s) of Fourier terms
h	Number of periods ahead to forecast (optional)

**Details**

The period of the Fourier terms is determined from the time series characteristics of x. When h is missing, the length of x also determines the number of rows for the matrix returned by fourier. Otherwise, the value of h determines the number of rows for the matrix returned by fourier, typically used for forecasting. The values within x are not used.

Typical use would omit h when generating Fourier terms fitting a model and include h when generating Fourier terms for forecasting.

When x is a ts object, the value of K should be an integer and specifies the number of sine and cosine terms to return. Thus, the matrix returned has 2\*K columns.

When x is a msts object, then K should be a vector of integers specifying the number of sine and cosine terms for each of the seasonal periods. Then the matrix returned will have 2\*sum(K) columns.

**Value**

Numerical matrix.



**Author(s)**

Rob J Hyndman

**See Also**

[seasonaldummy](#)

**Examples**

```
library(astsa)
# Dynamic Harmonic regression
sf1 = auto.sarima(birth,xreg = fourier(birth,K= 6),iter = 500,chains = 1)
```

---

gamma

*Define a gamma prior distribution*

---

**Description**

gamma(shape,rate)

**Usage**

```
gamma(shape = 2, rate = 1)
```

**Arguments**

shape	the form parameter alpha in gamma distribution
rate	the rate parameter beta in gamma distribution

**Details**

Define a gamma prior distribution using the hyper parameters shape and rate, by default an gamma(2,1) distribution is return.

**Value**

a numerical vector interpreted as a prior in Stan

---

garch *A constructor for a GARCH(s,k,h) model.*

---

### Description

Constructor of the GARCH(s, k, h) object for Bayesian estimation in **Stan**.

### Usage

```
garch(ts, order = c(1, 1, 0), arma = c(0, 0), xreg = NULL,
      genT = FALSE, asym = "none", series.name = NULL)
```

### Arguments

ts	a numeric or ts object with the univariate time series.
order	A specification of the garch model: the three components (s, k, h) are the arch order, the garch order, and the mgarch order.
arma	A specification of the ARMA model, same as order parameter: the two components (p, q) are the AR order, and the MA order.
xreg	Optionally, a numerical matrix of external regressors, which must have the same number of rows as ts. It should not be a data frame.
genT	a boolean value to specify for a generalized t-student garch model.
asym	a string value for the asymmetric function for an asymmetric GARCH process. By default the value "none" for standard GARCH process. If "logit" a logistic function is used for asymmetry, and if "exp" an exponential function is used.
series.name	an optional string vector with the time series names.

### Details

The function returns a list with the data for running `stan()` function of **rstan** package.

By default the `garch()` function generates a GARCH(1,1) model, when `genT` option is TRUE a t-student innovations GARCH model (see Ardia (2010)) is generated, and for Asymmetric GARCH models use the option `asym` for specify the asymmetric function, see Fonseca, et. al (2019) for more details.

The default priors used in a GARCH(s,k,h) model are:

- `ar ~ normal(0,0.5)`
- `ma ~ normal(0,0.5)`
- `mu0 ~ t-student(0,2.5,6)`
- `sigma0 ~ t-student(0,1,7)`
- `arch ~ normal(0,0.5)`
- `garch ~ normal(0,0.5)`
- `mgarch ~ normal(0,0.5)`

- $dfv \sim \text{gamma}(2,0.1)$
- $breg \sim \text{t-student}(0,2.5,6)$

For changing the default prior use the function `set_prior()`.

### Value

The function returns a list with the data for running `stan()` function of **rstan** package.

### Author(s)

Asael Alonzo Matamoros.

### References

- Engle, R. (1982). Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. *Econometrica*, 50(4), 987-1007. url: <http://www.jstor.org/stable/1912773>.
- Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*. 31(3), 307-327. doi: [https://doi.org/10.1016/0304-4076\(86\)90063-1](https://doi.org/10.1016/0304-4076(86)90063-1).
- Fonseca, T. and Cequeira, V. and Migon, H. and Torres, C. (2019). The effects of degrees of freedom estimation in the Asymmetric GARCH model with Student-t Innovations. *arXiv* doi: arXiv: 1910.01398.
- Ardia, D. and Hoogerheide, L. (2010). Bayesian Estimation of the GARCH(1,1) Model with Student-t Innovations. *The R Journal*. 2(7), 41-47. doi: 10.32614/RJ-2010-014.

### See Also

[Sarima auto.arima set\\_prior](#)

### Examples

```
# Declaring a garch(1,1) model for the ipc data.
dat = garch(ipc,order = c(1,1,0))
dat

# Declaring a t-student M-GARCH(2,3,1)-ARMA(1,1) process for the ipc data.
dat = garch(ipc,order = c(2,3,1),arma = c(1,1),genT = TRUE)
dat

# Declaring a logistic Asymmetric GARCH(1,1) process.
dat = garch(ipc,order = c(1,1,0),asym = "logit")
dat
```

---

get\_parameters      *Get parameters of a varstan object*

---

**Description**

Get the sampled parameters of a varstan object.

**Usage**

```
get_parameters(object)
```

**Arguments**

object      a varstan object

**Value**

a vector with the sampled parameters

**Author(s)**

Asael Alonzo Matamoros

**Examples**

```
library(astsa)
sf1 = auto.sarima(birth,iter = 500,chains = 1)
get_parameters(sf1)
```

---

get\_prior      *Get the prior distribution of a model parameter*

---

**Description**

The functions gets the defined distribution of a defined model parameter

**Usage**

```
get_prior(model,par,lag = 0)
```

**Arguments**

model	a time series model class specified in varstan.
par	a string value with the desired parameter which a prior is defined could be: "mu0", "sigma0", "ar", "ma", "arch", "garch", "mgarch", "dfv", "df", "LKJ" or "breg".
lag	an optional integer value, indicates the desired lag of the parameter which the prior is defined if lag = 0, then the prior distribution will be applied for all lags

**Value**

None. Prints the prior distribution of a desired parameter.

**Author(s)**

Asael Alonzo Matamoros

**Examples**

```
library(astsa)
# get all the ar parameters
dat = Sarima(birth,order = c(2,1,2))
get_prior(model = dat,par = "ar")

# change the mean constant parameter
dat = set_prior(model = dat,par = "mu0",dist = student(0,2.5,7))
get_prior(dat,par = "mu0")

# change and print only the second ma parameter
dat = set_prior(model = dat,par = "ma",dist = beta(2,2),lag = 2)
get_prior(dat,par = "ma")
```

---

ggacf

*acf plot*


---

**Description**

Plot of the auto-correlation function for a univariate time series.

**Usage**

```
ggacf(y, title = NULL)
```

**Arguments**

y	a numeric vector or an object of the ts class containing a stationary time series.
title	a string with the plot's title.

**Value**

None. Function produces a ggplot2 graph.

**Author(s)**

Asael Alonzo Matamoros

**Examples**

```
x = rnorm(100)
ggacf(x)
```

---

gghist

*Histogram with optional normal density functions*

---

**Description**

Plots a histogram and density estimates using ggplot.

**Usage**

```
gghist(y, title = NULL, xlab = NULL, ylab = "counts", bins, add.normal = TRUE)
```

**Arguments**

y	a numeric vector or an object of the ts class containing a stationary time series.
title	a string with the plot's title.
xlab	a string with the plot's x axis label. By default a NULL value
ylab	a string with the plot's y axis label. By default a "counts" value
bins	The number of bins to use for the histogram. Selected by default using the Friedman-Diaconis rule.
add.normal	A boolean value. Add a normal density function for comparison, by default add.normal = TRUE.

**Value**

None. Function produces a ggplot2 graph.

**Author(s)**

Rob J Hyndman

**Examples**

```
x = rnorm(100)
gghist(x, add.normal = TRUE)
```

---

ggnorm	qqplot with normal qqline
--------	---------------------------

---

**Description**

Plot the quantile-quantile plot and quantile-quantile line using ggplot.

**Usage**

```
ggnorm(y, title = NULL, add.normal = TRUE)
```

**Arguments**

<code>y</code>	a numeric vector or an object of the <code>ts</code> class containing a stationary time series.
<code>title</code>	a string with the plot's title.
<code>add.normal</code>	Add a normal density function for comparison.

**Value**

None. Function produces a ggplot2 graph.

**Author(s)**

Asael Alonzo Matamoros

**Examples**

```
x = rnorm(100)
ggnorm(x)
```

---

ggpacf	pacf plot.
--------	------------

---

**Description**

Plot of the partial autocorrelation function for a univariate time series.

**Usage**

```
ggpacf(y, title = NULL)
```

**Arguments**

<code>y</code>	a numeric vector or an object of the <code>ts</code> class containing a stationary time series.
<code>title</code>	a string with the plot's title.

**Value**

None.

**Author(s)**

Mitchell O'Hara-Wild and Asael Alonzo Matamoros

**Examples**

```
x = rnorm(100)
ggpacf(x)
```

---

Holt

*A constructor for a Holt trend state-space model.*

---

**Description**

Constructor of the `ets("A", "A", "Z")` object for Bayesian estimation in **Stan**.

**Usage**

```
Holt(ts,damped = FALSE,xreg = NULL,genT = FALSE,series.name = NULL)
```

**Arguments**

<code>ts</code>	a numeric or <code>ts</code> object with the univariate time series.
<code>damped</code>	a boolean value to specify a damped trend local level model. By default is <code>FALSE</code> . If trend option is <code>FALSE</code> then <code>damped</code> is set to <code>FALSE</code> automatically.
<code>xreg</code>	Optionally, a numerical matrix of external regressors, which must have the same number of rows as <code>ts</code> . It should not be a data frame.
<code>genT</code>	a boolean value to specify for a generalized t-student SSM model.
<code>series.name</code>	an optional string vector with the time series names.

**Details**

The function returns a list with the data for running `stan()` function of **rstan** package.

When `genT` option is `TRUE` a t-student innovations ssm model (see Ardia (2010)) is generated see Fonseca, et. al (2019) for more details.

The default priors used in a `ssm()` model are:

- `level` ~ `normal(0,0.5)`
- `trend` ~ `normal(0,0.5)`
- `damped` ~ `normal(0,0.5)`
- `sigma0` ~ `t-student(0,1,7)`



- level1 ~ normal(0,1)
- trend1 ~ normal(0,1)
- dfv ~ gamma(2,0.1)
- breg ~ t-student(0,2.5,6)

For changing the default prior use the function `set_prior()`.

### Value

The function returns a list with the data for running `stan()` function of **rstan** package.

### Author(s)

Asael Alonzo Matamoros.

### References

Fonseca, T. and Cequeira, V. and Migon, H. and Torres, C. (2019). The effects of degrees of freedom estimation in the Asymmetric GARCH model with Student-t Innovations. *arXiv* doi: arXiv: 1910.01398.

### See Also

[Sarima](#) [auto.arima](#) [set\\_prior](#) [garch](#)

### Examples

```
mod1 = Holt(ipc)

# Declaring a Holt damped trend model for the ipc data.
mod2 = Holt(ipc,damped = TRUE)
```

---

Hw

*A constructor for a Holt-Winters state-space model.*

---

### Description

Constructor of the `ets("A", "A", "A")` object for Bayesian estimation in **Stan**.

### Usage

```
Hw(ts,damped = FALSE,xreg = NULL,period = 0,genT = FALSE,series.name = NULL)
```

**Arguments**

<code>ts</code>	a numeric or ts object with the univariate time series.
<code>damped</code>	a boolean value to specify a damped trend local level model. By default is FALSE. If trend option is FALSE then damped is set to FALSE automatically.
<code>xreg</code>	Optionally, a numerical matrix of external regressors, which must have the same number of rows as <code>ts</code> . It should not be a data frame.
<code>period</code>	an integer specifying the periodicity of the time series by default the value <code>frequency(ts)</code> is used.
<code>genT</code>	a boolean value to specify for a generalized t-student SSM model.
<code>series.name</code>	an optional string vector with the time series names.

**Details**

The function returns a list with the data for running `stan()` function of **rstan** package.

When `genT` option is TRUE a t-student innovations ssm model (see Ardia (2010)) is generated see Fonseca, et. al (2019) for more details.

The default priors used in a `ssm()` model are:

- `level` ~ `normal(0,0.5)`
- `Trend` ~ `normal(0,0.5)`
- `damped` ~ `normal(0,0.5)`
- `Seasonal` ~ `normal(0,0.5)`
- `sigma0` ~ `t-student(0,1,7)`
- `level1` ~ `normal(0,1)`
- `trend1` ~ `normal(0,1)`
- `seasonal1` ~ `normal(0,1)`
- `dfv` ~ `gamma(2,0.1)`
- `breg` ~ `t-student(0,2.5,6)`

For changing the default prior use the function `set_prior()`.

**Value**

The function returns a list with the data for running `stan()` function of **rstan** package.

**Author(s)**

Asael Alonzo Matamoros.

**References**

Fonseca, T. and Cequeira, V. and Migon, H. and Torres, C. (2019). The effects of degrees of freedom estimation in the Asymmetric GARCH model with Student-t Innovations. *arXiv* doi: arXiv: 1910.01398.

**See Also**

[Sarima auto. arima set\\_prior garch](#)

**Examples**

```
mod1 = Hw(ipc)

# Declaring a Holt Winters damped trend model for the ipc data.
mod2 = Hw(ipc,damped = TRUE)

# Declaring an additive Holt-Winters model for the birth data
mod3 = Hw(birth,damped = FALSE)
```

---

inverse.chisq

*Define an inverse gamma prior distribution*

---

**Description**

inverse.chisq(df)

**Usage**

```
inverse.chisq(df = 7)
```

**Arguments**

df                    the degree freedom parameter df

**Details**

Define a inverse chi square prior distribution using the hyper parameter df, by default an inverse.chisq(df = 2) distribution is return.

If sigma has a chi square distribution then 1/sigma has n inverse chi square distribution.

**Value**

a numerical vector interpreted as a prior in Stan

---

inverse.gamma	<i>Define an inverse gamma prior distribution</i>
---------------	---

---

**Description**

inverse.gamma(shape,rate)

**Usage**

inverse.gamma(shape = 2, rate = 1)

**Arguments**

shape	the form parameter alpha in gamma distribution
rate	the rate parameter beta in gamma distribution

**Details**

Define a inverse.gamma prior distribution using the hyper parameters shape and rate, by default an inverse.gamma(2,1) distribution is return.

If sigma has a gamma distribution then 1/sigma has n inverse gamma distribution. The rate parameter is the inverse of an scale parameter.

**Value**

a numerical vector interpreted as a prior in Stan

---

ipc	<i>Monthly inflation coefficients from 1980-2018.</i>
-----	---

---

**Description**

Monthly return coefficients for the inflation. An economic indicator of a country's economy.

**Usage**

ipc

**Format**

A time series of monthly data from 1980 to 2018.

**Source**

[https://www.bch.hn/series\\_estadisticas.php](https://www.bch.hn/series_estadisticas.php)

---

jeffrey	<i>Define a non informative Jeffrey's prior for the degree freedom hyper parameter</i>
---------	--

---

**Description**

jeffrey.df()

**Usage**

jeffrey()

**Details**

Define a non informative Jeffrey's prior distribution, by default an jeffrey.df() distribution is return. This prior can only be used in garch models with t-student innovations, or Bekk models with generalized t-student distribution.

**Value**

a numerical vector interpreted as a prior in Stan

---

laplace	<i>Define a Laplace prior distribution</i>
---------	--

---

**Description**

laplace(mu,sd)

**Usage**

laplace(mu = 0, sd = 1)

**Arguments**

mu	the location parameter mu
sd	the standard deviation parameter sigma

**Details**

Define a Laplace prior distribution using the hyper parameters mu and sigma, by default a standard Laplace distribution is return.

The laplace distribution is exactly the same as the double exponential distribution

**Value**

a numerical vector interpreted as a prior in Stan

---

LKJ *Define a LKJ matrix prior distribution*

---

**Description**

LKJ(df)

**Usage**

LKJ(df = 2)

**Arguments**

df                    the degree freedom parameter df

**Details**

Define a Lewandowski Kurowicka and Joe (LKJ) matrix correlation prior distribution using the degree freedom df hyper parameter, by default a LKJ(2) distribution is return.

**Value**

a numerical vector interpreted as a prior in Stan

**References**

Lewandowski D, Kurowicka D, Joe H (2009). "Generating random correlation matrices based on vines and extended onion method." Journal of Multivariate Analysis, 100(9), 1989 2001. ISSN 0047-259X. doi:<https://doi.org/10.1016/j.jmva.2009.04.008>. URL: <http://www.sciencedirect.com/science/article/pii/S0047259X0900008>

---

LocalLevel *A constructor for local level state-space model.*

---

**Description**

Constructor of the ets("A", "N", "N") object for Bayesian estimation in **Stan**.

**Usage**

LocalLevel(ts, xreg = NULL, genT = FALSE, series.name = NULL)

**Arguments**

ts                    a numeric or ts object with the univariate time series.  
 xreg                  Optionally, a numerical matrix of external regressors, which must have the same number of rows as ts. It should not be a data frame.  
 genT                  a boolean value to specify for a generalized t-student SSM model.  
 series.name         an optional string vector with the time series names.

## Details

The function returns a list with the data for running `stan()` function of **rstan** package.

By default the `ssm()` function generates a local level model (or a `ets("A","N","N")` or exponential smoothing model from the **forecast** package). If `trend` is set `TRUE`, then a local trend `ssm` model is defined (a equivalent `ets("A","A","N")` or Holt model from the **forecast** package). For damped trend models set `damped` to `TRUE`. If `seasonal` is set to `TRUE` a seasonal local level model is defined (a equivalent `ets("A","N","A")` model from the **forecast** package). For a Holt-Winters method (`ets("A","A","A")`) set `Trend` and `seasonal` to `TRUE`.

When `genT` option is `TRUE` a t-student innovations `ssm` model (see Ardia (2010)) is generated see Fonseca, et. al (2019) for more details.

The default priors used in a `ssm()` model are:

- `level` ~ `normal(0,0.5)`
- `sigma0` ~ `t-student(0,1,7)`
- `level1` ~ `normal(0,1)`
- `dfv` ~ `gamma(2,0.1)`
- `breg` ~ `t-student(0,2.5,6)`

For changing the default prior use the function `set_prior()`.

## Value

The function returns a list with the data for running `stan()` function of **rstan** package.

## Author(s)

Asael Alonzo Matamoros.

## References

Fonseca, T. and Cequeira, V. and Migon, H. and Torres, C. (2019). The effects of degrees of freedom estimation in the Asymmetric GARCH model with Student-t Innovations. *arXiv* doi: arXiv: 1910.01398.

## See Also

[Sarima](#) [auto.arima](#) [set\\_prior](#) [garch](#)

## Examples

```
mod1 = LocalLevel(ipc)
```

---

loglik	<i>Extract posterior sample of the accumulated log-likelihood from a varstan object</i>
--------	---

---

### Description

Convenience function for extracting the posterior sample of the accumulated log-likelihood array from a fitted varstan object.

### Usage

```
loglik(object, permuted = TRUE)
```

### Arguments

object	A varstan object of the time series fitted model.
permuted	A logical scalar indicating whether the draws after the warmup period in each chain should be permuted and merged. If FALSE, the original order is kept. For each stanfit object, the permutation is fixed (i.e., extracting samples a second time will give the same sequence of iterations).

### Value

A real value with the accumulated log likelihood.

### References

Vehtari, A., Gelman, A., & Gabry J. (2016). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *In Statistics and Computing*, doi:10.1007/s11222-016-9696-4.

Gelman, A., Hwang, J., & Vehtari, A. (2014). Understanding predictive information criteria for Bayesian models. *Statistics and Computing*, 24, 997-1016.

Watanabe, S. (2010). Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. *The Journal of Machine Learning Research*, 11, 3571-3594.

### Examples

```
library(astsa)
model = Sarima(birth,order = c(0,1,2),seasonal = c(1,1,1))
fit1 = varstan(model,iter = 500,chains = 1)

log1 = loglik(fit1)
log1
```



---

log_lik.varstan	<i>Extract posterior sample of the pointwise log-likelihood from a varstan object.</i>
-----------------	--

---

### Description

Convenience function for extracting the pointwise log-likelihood matrix or array from a fitted Stan model.

### Usage

```
## S3 method for class 'varstan'  
log_lik(object, permuted = TRUE, ...)
```

### Arguments

object	A varstan object of the time series fitted model.
permuted	A logical scalar indicating whether the draws after the warmup period in each chain should be permuted and merged. If FALSE, the original order is kept. For each stanfit object, the permutation is fixed (i.e., extracting samples a second time will give the same sequence of iterations).
...	additional values need in log_lik methods

### Value

Usually, an  $S \times N$  matrix containing the pointwise log-likelihood samples, where  $S$  is the number of samples and  $N$  is the number of observations in the data. If `permuted` is FALSE, an  $S \times N \times R$  array is returned, where  $R$  is the number of fitted chains.

### References

Vehtari, A., Gelman, A., & Gabry J. (2016). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *In Statistics and Computing*, doi:10.1007/s11222-016-9696-4.

Gelman, A., Hwang, J., & Vehtari, A. (2014). Understanding predictive information criteria for Bayesian models. *Statistics and Computing*. 24, 997-1016.

Watanabe, S. (2010). Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. *The Journal of Machine Learning Research*. 11, 3571-3594.

### Examples

```
library(astsa)  
model = Sarima(birth,order = c(0,1,2),seasonal = c(1,1,1))  
fit1 = varstan(model,iter = 500,chains = 1)  
  
log1 = log_lik(fit1)
```

```
log1
```

---

```
loo.varstan          Leave-one-out cross-validation
```

---

## Description

The `loo` method for `varstan` objects. Computes approximate leave-one-out cross-validation using Pareto smoothed importance sampling (PSIS-LOO CV).

## Usage

```
## S3 method for class 'varstan'
loo(x, ...)
```

## Arguments

```
x          A varstan object
...        additional values need in loo methods
```

## Value

an object from the `loo` class with the results of the Pareto-Smooth Importance Sampling, leave one out cross validation for model selection.

## References

Vehtari, A., Gelman, A., & Gabry J. (2016). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *In Statistics and Computing*, doi:10.1007/s11222-016-9696-4.

Gelman, A., Hwang, J., & Vehtari, A. (2014). Understanding predictive information criteria for Bayesian models. *Statistics and Computing*, 24, 997-1016.

Watanabe, S. (2010). Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. *The Journal of Machine Learning Research*, 11, 3571-3594.

## See Also

- The `loo` package [vignettes](#) for demonstrations.
- [psis\(\)](#) for the underlying Pareto Smoothed Importance Sampling (PSIS) procedure used in the LOO-CV approximation.
- [pareto-k-diagnostic](#) for convenience functions for looking at diagnostics.
- [loo\\_compare\(\)](#) for model comparison.

**Examples**

```
library(astsa)
model = Sarima(birth,order = c(0,1,2),seasonal = c(1,1,1))
fit1 = varstan(model,iter = 500,chains = 1)

loo1 = loo(fit1)
loo1
```

---

mcmc\_plot.varstan      *MCMC Plots Implemented in **bayesplot***

---

**Description**

Convenient way to call MCMC plotting functions implemented in the **bayesplot** package.

**Usage**

```
## S3 method for class 'varstan'
mcmc_plot(
  object,
  pars = NULL,
  combo = c("dens", "trace"),
  fixed = FALSE,
  exact_match = FALSE,
  ...
)

mcmc_plot(object, ...)
```

**Arguments**

object	An varstan object.
pars	Names of parameters to be plotted, as given by a character vector or regular expressions. By default, all parameters except for group-level and smooth effects are plotted. May be ignored for some plots.
combo	An array that contains the types of plot. By default <code>combo = c("dens","trace")</code> . Supported types are (as names) <code>hist</code> , <code>dens</code> , <code>hist_by_chain</code> , <code>dens_overlay</code> , <code>violin</code> , <code>intervals</code> , <code>areas</code> , <code>acf</code> , <code>acf_bar</code> , <code>trace</code> , <code>trace_highlight</code> , <code>scatter</code> , <code>rhat</code> , <code>rhat_hist</code> , <code>neff</code> , <code>neff_hist</code> , <code>nuts_acceptance</code> , <code>nuts_divergence</code> , <code>nuts_stepsize</code> , <code>nuts_treedepth</code> , and <code>nuts_energy</code> . For an overview on the various plot types see <a href="#">MCMC-overview</a> .
fixed	Indicates whether parameter names should be matched exactly (TRUE) or treated as regular expressions (FALSE). Default is FALSE.
exact_match	Deprecated alias of argument <code>fixed</code> .
...	Additional arguments passed to the plotting functions. See <a href="#">MCMC-overview</a> for more details.

**Value**

A `ggplot` object that can be further customized using the **ggplot2** package.

**Examples**

```
## Not run:
sf1 = stan_ssm(ipc, iter = 500, chains = 1)

# plot posterior intervals
mcmc_plot(sf1)

# only show population-level effects in the plots
mcmc_plot(sf1, pars = "level")

## End(Not run)
```

---

`model`*Print the defined model of a varstan object.*

---

**Description**

The function returns a string with the users defined model for the given time series data.

**Usage**

```
model(object, ...)
```

**Arguments**

<code>object</code>	a varstan object or one of the defined current defined models in varstan package.
<code>...</code>	additional values need in print methods

**Details**

if `object` is a varstan object the function will print the information of the defined model inside of the object. If `object` is one of the model classes (like `Sarima`, `garch`, `SVM` or `varma`), then it will print the model information as well.

For full information of the model with the used priors use the function `report` or just print the object.

**Value**

a string with the defined time series model.

**Author(s)**

Asael Alonzo Matamoros.

**See Also**

report print

**Examples**

```
library(astsa)
model1 = Sarima(birth,order = c(0,1,2),seasonal = c(1,1,1))
model(model1)
```

---

naive	<i>Naive and Random Walk models.</i>
-------	--------------------------------------

---

**Description**

naive is the model constructor for a random walk model applied to  $y$ . This is equivalent to an ARIMA(0,1,0) model. naive() is simply a wrapper to maintain forecast package similitude. seasonal returns the model constructor for a seasonal random walk equivalent to an ARIMA(0,0,0)(0,1,0) $m$  model where  $m$  is the seasonal period.

**Usage**

```
naive(ts,seasonal = FALSE,m = 0)
```

**Arguments**

ts	a numeric or ts object with the univariate time series.
seasonal	a Boolean value for select a seasonal random walk instead.
m	an optional integer value for the seasonal period.

**Details**

The random walk with drift model is

$$Y_t = mu_0 + Y_{t-1} + \epsilon_t$$

where  $\epsilon_t$  is a normal iid error.

The seasonal naive model is

$$Y_t = mu_0 + Y_{t-m} + \epsilon_t$$

where  $\epsilon_t$  is a normal iid error.

**Value**

The function returns a list with the data for running stan() function of rstan package.

**Author(s)**

Asael Alonzo Matamoros

## References

Hyndman, R. & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*. 26(3), 1-22. doi: 10.18637/jss.v027.i03.

Box, G. E. P. and Jenkins, G.M. (1978). Time series analysis: Forecasting and control. San Francisco: Holden-Day. *Biometrika*, 60(2), 297-303. doi:10.1093/biomet/65.2.297.

Kennedy, P. (1992). Forecasting with dynamic regression models: Alan Pankratz, 1991. *International Journal of Forecasting*. 8(4), 647-648. url: <https://EconPapers.repec.org/RePEc:eee:intfor:v:8:y:1992:i:>

## See Also

[Sarima](#)

## Examples

```
library(astsa)
# A seasonal Random-walk model.
model = naive(birth,seasonal = TRUE)
model
```

---

normal

*Define a normal prior distribution*

---

## Description

normal(mu,sd)

## Usage

```
normal(mu = 0, sd = 1)
```

## Arguments

mu	the location parameter mu
sd	the standard deviation parameter sigma

## Details

Define a normal prior distribution using the hyper parameters mu and sigma, by default a standard normal distribution is return.

## Value

a numerical vector interpreted as a prior in Stan

---

oildata	<i>Annual oil production in Saudi Arabia</i>
---------	--

---

**Description**

Annual oil production (millions of tonnes), Saudi Arabia, 1965-2013.

**Usage**

oildata

**Format**

the format is: Annual Time-Series (1:18) from 1996 to 2013:

**Source**

**fpp2**

**References**

Hyndman, R. & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*. 26(3), 1-22. doi: 10.18637/jss.v027.i03

---

plot.varstan	<i>plot methods for varstan models.</i>
--------------	---

---

**Description**

Preliminary plot methods for varstan models only valid for univariate time series models. The function prints the fitted values time series, the trace and density plots for the sampled model parameters, or the residuals' posterior mean time series.

**Usage**

```
## S3 method for class 'varstan'
plot(x, prob = 0.95, ...)
```

**Arguments**

x	An object of class varstan.
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%.
...	Further arguments passed to mcmc_combo.

**Value**

None. Function produces a ggplot2 graph.

**Examples**

```
library(astsa)
sf1 = auto.sarima(ts = birth, iter = 500, chains = 1)
# fitted model
plot(sf1)
```

---

posterior\_epred.varstan

*Expected Values of the Posterior Predictive Distribution*

---

**Description**

Compute posterior samples of the expected value/mean of the posterior predictive distribution. Can be performed for the data used to fit the model (posterior predictive checks) or for new data. By definition, these predictions have smaller variance than the posterior predictions performed by the [posterior\\_predict.varstan](#) method. This is because only the uncertainty in the mean is incorporated in the samples computed by `posterior_epred` while any residual error is ignored. However, the estimated means of both methods averaged across samples should be very similar.

**Usage**

```
## S3 method for class 'varstan'
posterior_epred(
  object,
  h = 0,
  xreg = NULL,
  robust = FALSE,
  draws = 1000,
  seed = NULL,
  ...
)
```

**Arguments**

object	a varstan object
h	An integer indicating the number of predictions. The default number of predictions is 12.
xreg	Optionally, a numerical matrix of external regressors, which must have the same number of rows as ts. It should not be a data frame.
robust	A boolean for obtain the robust estimation. The default



draws	An integer indicating the number of draws to return. The default number of draws is 1000
seed	An optional <a href="#">seed</a> to use.
...	Further arguments passed to posterior_predict.

**Value**

An array of predicted *mean* response values. For categorical and ordinal models, the output is an  $S \times N \times C$  array. Otherwise, the output is an  $S \times N$  matrix, where  $S$  is the number of posterior samples,  $N$  is the number of observations, and  $C$  is the number of categories. In multivariate models, an additional dimension is added to the output which indexes along the different response variables.

---

posterior\_interval      *Posterior uncertainty intervals*

---

**Description**

The posterior\_interval function computes Bayesian posterior uncertainty intervals. These intervals are often referred to as *credible* intervals, for more details see **rstanarm**

**Usage**

```
posterior_interval(mat, prob = 0.9, ...)
```

**Arguments**

mat	a matrix containing the posterior samples of a fitted parameter
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%.
...	Further arguments passed to posterior_intervals.

**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  $100 \times p\%$  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.

**Author(s)**

Asael Alonzo Matamoros

---

```
posterior_predict.varstan
```

*Draw from posterior predictive h steps ahead 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 'varstan'
posterior_predict(
  object,
  h = 0,
  xreg = NULL,
  robust = FALSE,
  draws = 1000,
  seed = NULL,
  ...
)
```

## Arguments

object	a varstan object
h	An integer indicating the number of predictions. The default number of predictions is 12.
xreg	Optionally, a numerical matrix of external regressors, which must have the same number of rows as ts. It should not be a data frame.
robust	A boolean for obtain the robust estimation. The default
draws	An integer indicating the number of draws to return. The default number of draws is 1000
seed	An optional <a href="#">seed</a> to use.
...	Further arguments passed to posterior_predict.

## Value

A draws by h data.frame of simulations from the posterior predictive distribution. Each row of the data.frame is a vector of predictions generated using a single draw of the model parameters from the posterior distribution.

**Author(s)**

Asael Alonzo Matamoros

---

`predictive_error.varstan`*Out-of-sample predictive errors*

---

**Description**

This is a convenience function for computing  $y - y_h$ . The method for stanreg objects calls `posterior_predict` internally, where as the method accepts the data.frame returned by `posterior_predict` as input and can be used to avoid multiple calls to `posterior_predict`.

**Usage**

```
## S3 method for class 'varstan'
predictive_error(
  object,
  newdata = NULL,
  xreg = NULL,
  draws = 1000,
  seed = NULL,
  ...
)
```

**Arguments**

<code>object</code>	Either a fitted model object returned by one of the <b>rstanarm</b> modeling functions (a stanreg object) or, for the "ppd" method, a matrix of draws from the posterior predictive distribution returned by <code>posterior_predict</code> .
<code>newdata</code>	An array with the newdata vector.
<code>xreg</code>	Optional, a numerical matrix of external regressors, which must have the same number of rows as <code>ts</code> . It should not be a data frame.
<code>draws, seed</code>	Optional arguments passed to <code>posterior_predict</code> . Please see the <b>Note</b> section below if <code>newdata</code> will be specified.
<code>...</code>	Further arguments passed to <code>predictive_error</code> .

**Value**

A draws by `nrow(newdata)` data.frame.

**Note**

If object is a **varstan** object of a varma model then newdata has to be a matrix with number of **cols** as the dimension of the time series and number of **rows** as the number new elements.

If object is a posterior\_predict data.frame, then the length of newdata has to be equal to the ncol of object.

If object is a posterior\_predict data.frame, for a **varma** model, then the dimension product of newdata matrix has to be equal to the ncol of object.

**See Also**

posterior\_predict function from rstanarm package, to draw from the posterior predictive distribution without computing predictive errors.

---

`print.garch`*Print a garch model*

---

**Description**

Print a garch model

**Usage**

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

**Arguments**

x                    a garch model from the varstan package  
...                   additional values need in print methods

**Value**

None. prints the object

---

print.Holt	<i>Print a Holt model</i>
------------	---------------------------

---

**Description**

Print a Holt model

**Usage**

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

**Arguments**

x	a Holt model from the varstan package
...	additional values need in print methods

**Value**

None. prints the object.

---

print.Hw	<i>Print a Holt-Winter model</i>
----------	----------------------------------

---

**Description**

Print a Holt-Winter model

**Usage**

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

**Arguments**

x	a Hw from the varstan package
...	additional values need in print methods

**Value**

None. prints the object.

---

print.LocalLevel	<i>Print a Local Level model</i>
------------------	----------------------------------

---

**Description**

Print a Local Level model

**Usage**

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

**Arguments**

x	a LocalLevel model from the varstan package
...	additional values need in print methods

**Value**

None. prints the object.

---

print.naive	<i>Print a naive model</i>
-------------	----------------------------

---

**Description**

Print a naive model

**Usage**

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

**Arguments**

x	a naive model from the varstan package
...	additional values need in print methods

**Value**

None. prints the object

---

print.Sarima	<i>Print a Sarima model</i>
--------------	-----------------------------

---

**Description**

Print a Sarima model

**Usage**

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

**Arguments**

x	a Sarima model from the varstan package
...	additional values need in print methods

**Value**

None. prints the object

---

print.ssm	<i>Print a state-space model</i>
-----------	----------------------------------

---

**Description**

Print a state-space model

**Usage**

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

**Arguments**

x	a ssm model from the varstan package
...	additional values need in print methods

**Value**

None. prints the object.

print.SVM                    *Print a Stochastic Volatility model*

---

**Description**

Print a Stochastic Volatility model

**Usage**

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

**Arguments**

x                    a SVM model from the varstan package  
...                  additional values need in print methods

**Value**

None. prints the object

---

print.varstan                *Print a varstan object*

---

**Description**

Print a varstan object

**Usage**

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

**Arguments**

x                    a varstan object  
...                  additional values need in print methods

**Value**

None. prints the object.



---

prior\_summary.varstan *Generic function for extracting information about prior distributions*

---

### Description

The function returns a report with the users defined model for the given time series data and all the current defined priors of the model.

### Usage

```
## S3 method for class 'varstan'  
prior_summary(object, ...)
```

### Arguments

object	a varstan object or one of the defined current defined reports in varstan package
...	additional values need in print methods

### Details

if object is a varstan object the function will print the information of the defined model inside of the object. If object is one of the model classes (like Sarima or garch) then it will print the report information as well.

### Value

none. prints a string with the defined time series model report

### Author(s)

Asael Alonzo Matamoros

### Examples

```
library(astsa)  
dat2 = garch(birth, order = c(1,1,0))  
prior_summary(dat2)
```

---

report	<i>Print a full report of the time series model in a varstan object.</i>
--------	--

---

### Description

The function returns a report with the users defined model for the given time series data and all the current defined priors of the model.

### Usage

```
report(object, ...)
```

### Arguments

object	an object varstan object or one of the defined current defined reports in varstan package
...	additional values need in print methods

### Details

if object is a varstan object the function will print the information of the defined model inside of the object. If object is one of the model classes (like Sarima or garch) then it will print the report information as well.

### Value

none. prints a string with the defined time series model report

### Author(s)

Asael Alonzo Matamoros

### Examples

```
library(astsa)
dat2 = garch(birth, order = c(1,1,0))
report(dat2)
```

---

residuals.varstan	<i>Generic function and method for extract the residual of a varstan object</i>
-------------------	---

---

### Description

The function returns the posterior estimate of the residuals of a varstan model, similar to the residual functions of other packages.

### Usage

```
## S3 method for class 'varstan'
residuals(object, robust = FALSE, ...)
```

### Arguments

object	A varstan object, <a href="#">varstan</a>
robust	A boolean value, if its TRUE it returns the median of the posterior distribution, and if its FALSE it returns the mean, by default is the FALSE value
...	Further arguments passed to posterior_residual.

### Details

This function only extracts the point-wise estimate of the time series residuals for extracting all the data use `extract_stan()` or `posterior_intervals` function

### Value

An array with the posterior estimate of the residuals of the time series model,

### Author(s)

Asael Alonzo Matamoros

---

Sarima	<i>Constructor a Multiplicative Seasonal ARIMA model.</i>
--------	---

---

### Description

Constructor of the SARIMA model for Bayesian estimation in **Stan**.

### Usage

```
Sarima(ts, order = c(1, 0, 0), seasonal = c(0, 0, 0), xreg = NULL, period = 0, series.name = NULL)
```

**Arguments**

<code>ts</code>	a numeric or ts object with the univariate time series.
<code>order</code>	A specification of the non-seasonal part of the ARIMA model: the three components (p, d, q) are the AR order, the number of differences, and the MA order.
<code>seasonal</code>	A specification of the seasonal part of the ARIMA model, same as order parameter: the three components (p, d, q) are the seasonal AR order, the degree of seasonal differences, and the seasonal MA order.
<code>xreg</code>	Optionally, a numerical matrix of external regressors, which must have the same number of rows as <code>ts</code> . It should not be a data frame.
<code>period</code>	an integer specifying the periodicity of the time series by default the value <code>frequency(ts)</code> is used.
<code>series.name</code>	an optional string vector with the series names.

**Details**

The function returns a list with the data for running `stan()` function of **rstan** package

If `xreg` option is used, the model by default will cancel the seasonal differences adjusted ( $D = 0$ ). If a value  $d > 0$  is used, all the regressor variables in `xreg` will be difference as well.

The default priors used in Sarima are:

- `ar` ~ normal(0,0.5)
- `ma` ~ normal(0,0.5)
- `mu0` ~ t-student(0,2.5,6)
- `sigma0` ~ t-student(0,1,7)
- `sar` ~ normal(0,0.5)
- `sma` ~ normal(0,0.5)
- `breg` ~ t-student(0,2.5,6)

For changing the default prior use the function `set_prior`

**Value**

The function returns a list with the data for running `stan()` function of **rstan** package.

**Author(s)**

Asael Alonzo Matamoros

**References**

Box, G. E. P. and Jenkins, G.M. (1978). Time series analysis: Forecasting and control. San Francisco: Holden-Day. *Biometrika*, 60(2), 297-303. doi:10.1093/biomet/65.2.297.

Kennedy, P. (1992). Forecasting with dynamic regression models: Alan Pankratz, 1991. *International Journal of Forecasting*. 8(4), 647-648. url: <https://EconPapers.repec.org/RePEc:eee:intfor:v:8:y:1992:i:4>

Hyndman, R. & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*. 26(3), 1-22. doi: 10.18637/jss.v027.i03

**See Also**[garch set\\_prior](#)**Examples**

```
# Declare a multiplicative seasonal ARIMA model for the birth data.

library(astsa)
model = Sarima(birth,order = c(0,1,2),seasonal = c(1,1,1))
model

#Declare an Dynamic Harmonic Regression model for the birth data.
model = Sarima(birth,order = c(1,0,1),xreg = fourier(birth,K = 2))
model
```

set\_prior

*Set a prior distribution to a model parameter.***Description**

setting a prior distribution to an specify model parameter.

**Usage**

```
set_prior(model, par = "ar", dist = normal(), lag = 0)
```

**Arguments**

model	a time series model class specified in <b>varstan</b> .
par	a string value with the desired parameter which a prior is defined could be: "mu0", "sigma0", "ar", "ma", "arch", "garch", "mgarch", "dfv", "df", "LKJ" or "breg".
dist	the distribution of the prior parameter. The only accepted is a prior_dist object.
lag	an optional integer value, indicates the desired lag of the parameter which the prior is defined if lag = 0, then the prior distribution will be applied for all lags.

**Details**

varstan provides its own functions to manipulate the parameter prior, this functions return a prior\_dist class, the dist argument only accepts this objects.

lag parameter is an optional value to change the prior distribution of one parameter in particular, this argument is only valid for: "ar", "ma", "arch", "garch", "mgarch", or "breg" par arguments. lag has to be a integer lower than the total amount of lagged parameters of the model. For example, to ONLY change the prior of the second "arch" parameter in a garch(3,1) model, a lag = 2 values must be specified.

For varma and Bekk models the covariance matrix Sigma is factorized as follows:

$$\text{Sigma} = \text{D}' \text{Omega} \text{D}$$

Where Omega is the correlation matrix that accepts an LKJ prior distribution D is a diagonal matrix with the inverse std deviations

For changing the degree freedom in a LKJ distribution for omega use `par = "LKJ"` and `dist = LKJ(df)`, where df are the desired degree freedom.

For changing the the priors in the diagonal D use `par = "sigma0"` and select one of the available prior distributions.

For ar, ma garch, arch parameters in varma and Bekk models only normal distributions priors with different mu and sd are accepted. Even if `get_prior` accepts its change, Stan will change it to a normal(0,1) prior.

### Value

a time series model class specified in `varstan` with the changed prior.

### Author(s)

Asael Alonzo Matamoros

### Examples

```
library(astsa)
dat = Sarima(birth,order = c(1,1,2))
dat = set_prior(model = dat,par = "ar",dist = normal(0,2))
dat

dat = set_prior(model = dat,par = "mu0",dist = student(mu=0,sd = 2.5,df = 7))
dat

dat = set_prior(model = dat,par = "ma",dist= beta(shape1 = 2,shape2 = 2),lag = 2)
dat
```

---

ssm

*A constructor for a Additive linear State space model.*

---

### Description

Constructor of the ets("Z", "Z", "Z") object for Bayesian estimation in **Stan**.

### Usage

```
ssm(ts,trend = FALSE,damped = FALSE,seasonal = FALSE,xreg = NULL,
    period = 0,genT = FALSE,series.name = NULL)
```

### Arguments

ts	a numeric or ts object with the univariate time series.
trend	a boolean value to specify a trend local level model. By default is FALSE.
damped	a boolean value to specify a damped trend local level model. By default is FALSE. If trend option is FALSE then damped is set to FALSE automatically.
seasonal	a boolean value to specify a seasonal local level model. By default is FALSE.
xreg	Optionally, a numerical matrix of external regressors, which must have the same number of rows as ts. It should not be a data frame.
period	an integer specifying the periodicity of the time series by default the value frequency(ts) is used.
genT	a boolean value to specify for a generalized t-student SSM model.
series.name	an optional string vector with the time series names.

### Details

The function returns a list with the data for running `stan()` function of **rstan** package.

By default the `ssm()` function generates a local level model (or a `ets("A","N","N")` or exponential smoothing model from the **forecast** package). If `trend` is set TRUE, then a local trend ssm model is defined (a equivalent `ets("A","A","N")` or Holt model from the **forecast** package). For damped trend models set `damped` to TRUE. If `seasonal` is set to TRUE a seasonal local level model is defined (a equivalent `ets("A","N","A")` model from the **forecast** package). For a Holt-Winters method (`ets("A","A","A")`) set `Trend` and `seasonal` to TRUE.

When `genT` option is TRUE a t-student innovations ssm model (see Ardia (2010)) is generated see Fonseca, et. al (2019) for more details.

The default priors used in a `ssm()` model are:

- `level ~ normal(0,0.5)`
- `Trend ~ normal(0,0.5)`
- `damped ~ normal(0,0.5)`
- `Seasonal ~ normal(0,0.5)`
- `sigma0 ~ t-student(0,1,7)`
- `level1 ~ normal(0,1)`
- `trend1 ~ normal(0,1)`
- `seasonal1 ~ normal(0,1)`
- `dfv ~ gamma(2,0.1)`
- `breg ~ t-student(0,2.5,6)`

For changing the default prior use the function `set_prior()`.

### Value

The function returns a list with the data for running `stan()` function of **rstan** package.

**Author(s)**

Asael Alonzo Matamoros.

**References**

Fonseca, T. and Cequeira, V. and Migon, H. and Torres, C. (2019). The effects of degrees of freedom estimation in the Asymmetric GARCH model with Student-t Innovations. *arXiv* doi: arXiv: 1910.01398.

**See Also**

[Sarima auto.arima set\\_prior garch](#)

**Examples**

```
mod1 = ssm(ipc)

# Declaring a Holt model for the ipc data.
mod2 = ssm(ipc,trend = TRUE,damped = TRUE)

# Declaring an additive Holt-Winters model for the birth data
mod3 = ssm(birth,trend = TRUE,damped = TRUE,seasonal = TRUE)
```

---

stan\_garch

*Fitting for a GARCH(s,k,h) model.*

---

**Description**

Fitting a GARCH(s,k,h) model in **Stan**.

**Usage**

```
stan_garch(
  ts,
  order = c(1, 1, 0),
  arma = c(0, 0),
  xreg = NULL,
  genT = FALSE,
  asym = "none",
  chains = 4,
  iter = 2000,
  warmup = floor(iter/2),
  adapt.delta = 0.9,
  tree.depth = 10,
  stepwise = TRUE,
  prior_mu0 = NULL,
  prior_sigma0 = NULL,
```



```

prior_ar = NULL,
prior_ma = NULL,
prior_mgarch = NULL,
prior_arch = NULL,
prior_garch = NULL,
prior_breg = NULL,
prior_gamma = NULL,
prior_df = NULL,
series.name = NULL,
...
)

```

### Arguments

<code>ts</code>	a numeric or ts object with the univariate time series.
<code>order</code>	A specification of the garch model: the three components (s, k, h) are the arch order, the garch order, and the mgarch order.
<code>arma</code>	A specification of the ARMA model, same as order parameter: the two components (p, q) are the AR order, and the MA order.
<code>xreg</code>	Optionally, a numerical matrix of external regressors, which must have the same number of rows as ts. It should not be a data frame.
<code>genT</code>	a boolean value to specify for a generalized t-student garch model.
<code>asym</code>	a string value for the asymmetric function for an asymmetric GARCH process. By default the value "none" for standard GARCH process. If "logit" a logistic function is used for asymmetry, and if "exp" an exponential function is used.
<code>chains</code>	An integer of the number of Markov Chains chains to be run, by default 4 chains are run.
<code>iter</code>	An integer of total iterations per chain including the warm-up, by default the number of iterations are 2000.
<code>warmup</code>	A positive integer specifying number of warm-up (aka burn-in) iterations. This also specifies the number of iterations used for step-size adaptation, so warm-up samples should not be used for inference. The number of warmup should not be larger than iter and the default is iter/2.
<code>adapt.delta</code>	An optional real value between 0 and 1, the thin of the jumps in a HMC method. By default is 0.9.
<code>tree.depth</code>	An integer of the maximum depth of the trees evaluated during each iteration. By default is 10.
<code>stepwise</code>	If TRUE, will do stepwise selection (faster). Otherwise, it searches over all models. Non-stepwise selection can be very slow, especially for seasonal models.
<code>prior_mu0</code>	The prior distribution for the location parameter in an ARMA model. By default the value is set NULL, then the default normal(0,1) prior is used.
<code>prior_sigma0</code>	The prior distribution for the scale parameter in an ARMA model. By default the value is set NULL, then the default student(7,0,1) prior is used.

prior_ar	The prior distribution for the auto-regressive parameters in an ARMA model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_ma	The prior distribution for the moving average parameters in an ARMA model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_mgarch	The prior distribution for the mean GARCH parameters in a GARCH model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_arch	The prior distribution for the arch parameters in a GARCH model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_garch	The prior distribution for the GARCH parameters in a GARCH model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_breg	The prior distribution for the regression coefficient parameters in a ARIMAX model. By default the value is set NULL, then the default student(7,0,1) priors are used.
prior_gamma	The prior distribution for the asymmetric parameters in an Asymmetric GARCH model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_df	The prior distribution for the degree freedom parameters in a t-student innovations GARCH model. By default the value is set NULL, then the default gamma(2,0.1) priors are used.
series.name	an optional string vector with the series names.
...	Further arguments passed to varstan function.

## Details

The function returns a varstan object with the fitted model.

By default the garch() function generates a GARCH(1,1) model, when genT option is TRUE a t-student innovations GARCH model (see Ardia (2010)) is generated, and for Asymmetric GARCH models use the option asym for specify the asymmetric function, see Fonseca, et. al (2019) for more details.

The default priors used in a GARCH(s,k,h) model are:

- ar ~ normal(0,0.5)
- ma ~ normal(0,0.5)
- mu0 ~ t-student(0,2.5,6)
- sigma0 ~ t-student(0,1,7)
- arch ~ normal(0,0.5)
- garch ~ normal(0,0.5)
- mgarch ~ normal(0,0.5)
- dfv ~ gamma(2,0.1)
- breg ~ t-student(0,2.5,6)

For changing the default prior use the function set\_prior().

**Value**

A varstan object with the fitted GARCH model.

**Author(s)**

Asael Alonzo Matamoros.

**References**

- Engle, R. (1982). Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. *Econometrica*, 50(4), 987-1007. url: <http://www.jstor.org/stable/1912773>.
- Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*. 31(3), 307-327. doi: [https://doi.org/10.1016/0304-4076\(86\)90063-1](https://doi.org/10.1016/0304-4076(86)90063-1).
- Fonseca, T. and Cequeira, V. and Migon, H. and Torres, C. (2019). The effects of degrees of freedom estimation in the Asymmetric GARCH model with Student-t Innovations. *arXiv* doi: arXiv: 1910.01398.
- Ardia, D. and Hoogerheide, L. (2010). Bayesian Estimation of the GARCH(1,1) Model with Student-t Innovations. *The R Journal*. 2(7), 41-47. doi: 10.32614/RJ-2010-014.

**See Also**

[Sarima auto.arima set\\_prior](#)

**Examples**

```
# Declaring a garch(1,1) model for the ipc data.
sf1 = stan_garch(ipc,order = c(1,1,0),iter = 500,chains = 1)

# Declaring a t-student M-GARCH(2,3,1)-ARMA(1,1) process for the ipc data.
sf2 = stan_garch(ipc,order = c(2,3,1),arma = c(1,1),genT = TRUE,iter = 500,chains = 1)
```

---

stan\_Holt

*Fitting an Holt state-space model.*

---

**Description**

Fitting an Holt state-space model in **Stan**.

**Usage**

```

stan_Holt(
  ts,
  damped = FALSE,
  xreg = NULL,
  genT = FALSE,
  chains = 4,
  iter = 2000,
  warmup = floor(iter/2),
  adapt.delta = 0.9,
  tree.depth = 10,
  stepwise = TRUE,
  prior_sigma0 = NULL,
  prior_level = NULL,
  prior_level1 = NULL,
  prior_trend = NULL,
  prior_trend1 = NULL,
  prior_damped = NULL,
  prior_breg = NULL,
  prior_df = NULL,
  series.name = NULL,
  ...
)

```

**Arguments**

<code>ts</code>	a numeric or ts object with the univariate time series.
<code>damped</code>	a boolean value to specify a damped trend local level model. By default is FALSE. If trend option is FALSE then damped is set to FALSE automatically.
<code>xreg</code>	Optionally, a numerical matrix of external regressors, which must have the same number of rows as ts. It should not be a data frame.
<code>genT</code>	a boolean value to specify for a generalized t-student SSM model.
<code>chains</code>	An integer of the number of Markov Chains chains to be run, by default 4 chains are run.
<code>iter</code>	An integer of total iterations per chain including the warm-up, by default the number of iterations are 2000.
<code>warmup</code>	A positive integer specifying number of warm-up (aka burn-in) iterations. This also specifies the number of iterations used for step-size adaptation, so warm-up samples should not be used for inference. The number of warmup should not be larger than iter and the default is iter/2.
<code>adapt.delta</code>	An optional real value between 0 and 1, the thin of the jumps in a HMC method. By default is 0.9.
<code>tree.depth</code>	An integer of the maximum depth of the trees evaluated during each iteration. By default is 10.

stepwise	If TRUE, will do stepwise selection (faster). Otherwise, it searches over all models. Non-stepwise selection can be very slow, especially for seasonal models.
prior_sigma0	The prior distribution for the scale parameter in an SSM model. By default the value is set NULL, then the default student(7,0,1) prior is used.
prior_level	The prior distribution for the level parameter in a SSM model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_level1	The prior distribution for the initial level parameter in a SSM model. By default the value is set NULL, then the default student(6,0,2.5) priors are used.
prior_trend	The prior distribution for the trend parameter in a SSM model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_trend1	The prior distribution for the initial trend parameter in a SSM model. By default the value is set NULL, then the default student(6,0,2.5) priors are used.
prior_damped	The prior distribution for the damped trend parameter in a SSM model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_breg	The prior distribution for the regression coefficient parameters in a ARMAX model. By default the value is set NULL, then the default student(7,0,1) priors are used.
prior_df	The prior distribution for the degree freedom parameters in a t-student innovations SSM model. By default the value is set NULL, then the default gamma(2,0.1) priors are used.
series.name	an optional string vector with the series names.
...	Further arguments passed to varstan function.

## Details

The function returns a varstan object with the fitted model.

When genT option is TRUE a t-student innovations ssm model (see Ardia (2010)) is generated see Fonseca, et. al (2019) for more details.

The default priors used in a ssm( ) model are:

- level ~ normal(0,0.5)
- Trend ~ normal(0,0.5)
- damped~ normal(0,0.5)
- sigma0 ~ t-student(0,1,7)
- level1 ~ normal(0,1)
- trend1 ~ normal(0,1)
- dfv ~ gamma(2,0.1)
- breg ~ t-student(0,2.5,6)

For changing the default prior use the function set\_prior().

**Value**

A varstan object with the fitted SSM model.

**Author(s)**

Asael Alonzo Matamoros.

**References**

Fonseca, T. and Cequeira, V. and Migon, H. and Torres, C. (2019). The effects of degrees of freedom estimation in the Asymmetric GARCH model with Student-t Innovations. *arXiv* doi: arXiv: 1910.01398.

**See Also**

[Sarima auto.arima set\\_prior garch](#)

**Examples**

```
# Declaring a Holt model for the ipc data.
sf1 = stan_Holt(ipc,iter = 500,chains = 1)

# Declaring a Holt damped trend model for the ipc data.
sf2 = stan_Holt(ipc,damped = TRUE,iter = 500,chains = 1)
```

---

stan\_Hw

*Fitting a Holt-Winters state-space model.*

---

**Description**

Fitting a Holt-Winters state-space model in **Stan**.

**Usage**

```
stan_Hw(  
  ts,  
  damped = FALSE,  
  xreg = NULL,  
  period = 0,  
  genT = FALSE,  
  chains = 4,  
  iter = 2000,  
  warmup = floor(iter/2),  
  adapt.delta = 0.9,  
  tree.depth = 10,  
  stepwise = TRUE,
```

```

    prior_sigma0 = NULL,
    prior_level = NULL,
    prior_level1 = NULL,
    prior_trend = NULL,
    prior_trend1 = NULL,
    prior_damped = NULL,
    prior_seasonal = NULL,
    prior_seasonal1 = NULL,
    prior_breg = NULL,
    prior_df = NULL,
    series.name = NULL,
    ...
)

```

### Arguments

<code>ts</code>	a numeric or ts object with the univariate time series.
<code>damped</code>	a boolean value to specify a damped trend local level model. By default is FALSE. If trend option is FALSE then damped is set to FALSE automatically.
<code>xreg</code>	Optionally, a numerical matrix of external regressors, which must have the same number of rows as ts. It should not be a data frame.
<code>period</code>	an integer specifying the periodicity of the time series by default the value frequency(ts) is used.
<code>genT</code>	a boolean value to specify for a generalized t-student SSM model.
<code>chains</code>	An integer of the number of Markov Chains chains to be run, by default 4 chains are run.
<code>iter</code>	An integer of total iterations per chain including the warm-up, by default the number of iterations are 2000.
<code>warmup</code>	A positive integer specifying number of warm-up (aka burn-in) iterations. This also specifies the number of iterations used for step-size adaptation, so warm-up samples should not be used for inference. The number of warmup should not be larger than iter and the default is iter/2.
<code>adapt.delta</code>	An optional real value between 0 and 1, the thin of the jumps in a HMC method. By default is 0.9.
<code>tree.depth</code>	An integer of the maximum depth of the trees evaluated during each iteration. By default is 10.
<code>stepwise</code>	If TRUE, will do stepwise selection (faster). Otherwise, it searches over all models. Non-stepwise selection can be very slow, especially for seasonal models.
<code>prior_sigma0</code>	The prior distribution for the scale parameter in an SSM model. By default the value is set NULL, then the default student(7,0,1) prior is used.
<code>prior_level</code>	The prior distribution for the level parameter in a SSM model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_level1</code>	The prior distribution for the initial level parameter in a SSM model. By default the value is set NULL, then the default student(6,0,2.5) priors are used.

<code>prior_trend</code>	The prior distribution for the trend parameter in a SSM model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_trend1</code>	The prior distribution for the initial trend parameter in a SSM model. By default the value is set NULL, then the default student(6,0,2.5) priors are used.
<code>prior_damped</code>	The prior distribution for the damped trend parameter in a SSM model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_seasonal</code>	The prior distribution for the seasonal parameter in a SSM model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_seasonal1</code>	The prior distribution for the initial seasonal parameters in a SSM model. The prior is specified for the first m seasonal parameters, where m is the periodicity of the defined time series. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_breg</code>	The prior distribution for the regression coefficient parameters in a ARMAX model. By default the value is set NULL, then the default student(7,0,1) priors are used.
<code>prior_df</code>	The prior distribution for the degree freedom parameters in a t-student innovations SSM model. By default the value is set NULL, then the default gamma(2,0.1) priors are used.
<code>series.name</code>	an optional string vector with the series names.
<code>...</code>	Further arguments passed to <code>varstan</code> function.

### Details

The function returns a `varstan` object with the fitted model.

When `genT` option is TRUE a t-student innovations ssm model (see Ardia (2010)) is generated see Fonseca, et. al (2019) for more details.

The default priors used in a `ssm( )` model are:

- level ~ normal(0,0.5)
- Trend ~ normal(0,0.5)
- damped~ normal(0,0.5)
- Seasonal ~ normal(0,0.5)
- sigma0 ~ t-student(0,1,7)
- level1 ~ normal(0,1)
- trend1 ~ normal(0,1)
- seasonal1 ~ normal(0,1)
- dfv ~ gamma(2,0.1)
- breg ~ t-student(0,2.5,6)

For changing the default prior use the function `set_prior()`.

### Value

A `varstan` object with the fitted SSM model.



**Author(s)**

Asael Alonzo Matamoros.

**References**

Fonseca, T. and Cequeira, V. and Migon, H. and Torres, C. (2019). The effects of degrees of freedom estimation in the Asymmetric GARCH model with Student-t Innovations. *arXiv* doi: arXiv: 1910.01398.

**See Also**

[Sarima auto.arima set\\_prior garch](#)

**Examples**

```
# Declaring a Holt-Winters model for the ipc data.
sf1 = stan_hw(ipc, iter = 500, chains = 1)

# Declaring a Holt-Winters damped trend model for the ipc data.
sf2 = stan_ssm(ipc, damped = TRUE, iter = 500, chains = 1)
```

---

stan\_LocalLevel      *Fitting a Local level state-space model.*

---

**Description**

Fitting a Local level state-space model in **Stan**.

**Usage**

```
stan_LocalLevel(
  ts,
  xreg = NULL,
  genT = FALSE,
  chains = 4,
  iter = 2000,
  warmup = floor(iter/2),
  adapt.delta = 0.9,
  tree.depth = 10,
  stepwise = TRUE,
  prior_sigma0 = NULL,
  prior_level = NULL,
  prior_level1 = NULL,
  prior_breg = NULL,
  prior_df = NULL,
  series.name = NULL,
```

```
    ...
  )
```

### Arguments

<code>ts</code>	a numeric or <code>ts</code> object with the univariate time series.
<code>xreg</code>	Optionally, a numerical matrix of external regressors, which must have the same number of rows as <code>ts</code> . It should not be a data frame.
<code>genT</code>	a boolean value to specify for a generalized t-student SSM model.
<code>chains</code>	An integer of the number of Markov Chains chains to be run, by default 4 chains are run.
<code>iter</code>	An integer of total iterations per chain including the warm-up, by default the number of iterations are 2000.
<code>warmup</code>	A positive integer specifying number of warm-up (aka burn-in) iterations. This also specifies the number of iterations used for step-size adaptation, so warm-up samples should not be used for inference. The number of warmup should not be larger than <code>iter</code> and the default is <code>iter/2</code> .
<code>adapt.delta</code>	An optional real value between 0 and 1, the thin of the jumps in a HMC method. By default is 0.9.
<code>tree.depth</code>	An integer of the maximum depth of the trees evaluated during each iteration. By default is 10.
<code>stepwise</code>	If TRUE, will do stepwise selection (faster). Otherwise, it searches over all models. Non-stepwise selection can be very slow, especially for seasonal models.
<code>prior_sigma0</code>	The prior distribution for the scale parameter in an SSM model. By default the value is set NULL, then the default student(7,0,1) prior is used.
<code>prior_level</code>	The prior distribution for the level parameter in a SSM model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_level1</code>	The prior distribution for the initial level parameter in a SSM model. By default the value is set NULL, then the default student(6,0,2.5) priors are used.
<code>prior_breg</code>	The prior distribution for the regression coefficient parameters in a ARMAX model. By default the value is set NULL, then the default student(7,0,1) priors are used.
<code>prior_df</code>	The prior distribution for the degree freedom parameters in a t-student innovations SSM model. By default the value is set NULL, then the default gamma(2,0.1) priors are used.
<code>series.name</code>	an optional string vector with the series names.
<code>...</code>	Further arguments passed to <code>varstan</code> function.

### Details

The function returns a `varstan` object with the fitted model.

When `genT` option is TRUE a t-student innovations ssm model (see Ardia (2010)) is generated see Fonseca, et. al (2019) for more details.

The default priors used in a `Local_level()` model are:

- level ~ normal(0,0.5)
- sigma0 ~ t-student(0,1,7)
- level1 ~ normal(0,1)
- dfv ~ gamma(2,0.1)
- breg ~ t-student(0,2.5,6)

For changing the default prior use the function `set_prior()`.

### Value

A varstan object with the fitted Local Level model.

### Author(s)

Asael Alonzo Matamoros.

### References

Fonseca, T. and Cequeira, V. and Migon, H. and Torres, C. (2019). The effects of degrees of freedom estimation in the Asymmetric GARCH model with Student-t Innovations. *arXiv* doi: arXiv: 1910.01398.

### See Also

[Sarima](#) [auto.arima](#) [set\\_prior](#) [garch](#)

### Examples

```
# Declaring a local level model for the ipc data.
sf1 = stan_LocalLevel(ipc, iter = 500, chains = 1)
```

---

stan\_naive

*Naive and Random Walk models.*

---

### Description

naive is the model constructor for a random walk model applied to y. This is equivalent to an ARIMA(0,1,0) model. naive() is simply a wrapper to maintain forecast package similitude. seasonal returns the model constructor for a seasonal random walk equivalent to an ARIMA(0,0,0)(0,1,0)<sub>m</sub> model where m is the seasonal period.

**Usage**

```
stan_naive(
  ts,
  seasonal = FALSE,
  m = 0,
  chains = 4,
  iter = 2000,
  warmup = floor(iter/2),
  adapt.delta = 0.9,
  tree.depth = 10,
  stepwise = TRUE,
  prior_mu0 = NULL,
  prior_sigma0 = NULL,
  series.name = NULL,
  ...
)
```

**Arguments**

<code>ts</code>	a numeric or <code>ts</code> object with the univariate time series.
<code>seasonal</code>	a Boolean value for select a seasonal random walk instead.
<code>m</code>	an optional integer value for the seasonal period.
<code>chains</code>	An integer of the number of Markov Chains chains to be run, by default 4 chains are run.
<code>iter</code>	An integer of total iterations per chain including the warm-up, by default the number of iterations are 2000.
<code>warmup</code>	A positive integer specifying number of warm-up (aka burn-in) iterations. This also specifies the number of iterations used for step-size adaptation, so warm-up samples should not be used for inference. The number of warmup should not be larger than <code>iter</code> and the default is <code>iter/2</code> .
<code>adapt.delta</code>	An optional real value between 0 and 1, the thin of the jumps in a HMC method. By default is 0.9.
<code>tree.depth</code>	An integer of the maximum depth of the trees evaluated during each iteration. By default is 10.
<code>stepwise</code>	If TRUE, will do stepwise selection (faster). Otherwise, it searches over all models. Non-stepwise selection can be very slow, especially for seasonal models.
<code>prior_mu0</code>	The prior distribution for the location parameter in an ARIMA model. By default the value is set NULL, then the default student(7,0,1) prior is used.
<code>prior_sigma0</code>	The prior distribution for the scale parameter in an ARIMA model. By default the value is set NULL, then the default student(7,0,1) prior is used.
<code>series.name</code>	an optional string vector with the series names.
<code>...</code>	Further arguments passed to <code>varstan</code> function.

## Details

The random walk with drift model is

$$Y_t = \mu u_0 + Y_{t-1} + \epsilon_t$$

where  $\epsilon_t$  is a normal iid error.

The seasonal naive model is

$$Y_t = \mu u_0 + Y_{t-m} + \epsilon_t$$

where  $\epsilon_t$  is a normal iid error.

## Value

A varstan object with the fitted naive Random Walk model.

## Author(s)

Asael Alonzo Matamoros

## References

Hyndman, R. & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*. 26(3), 1-22. doi: 10.18637/jss.v027.i03.

Box, G. E. P. and Jenkins, G.M. (1978). Time series analysis: Forecasting and control. San Francisco: Holden-Day. *Biometrika*, 60(2), 297-303. doi: 10.1093/biomet/65.2.297.

Kennedy, P. (1992). Forecasting with dynamic regression models: Alan Pankratz, 1991. *International Journal of Forecasting*. 8(4), 647-648. url: <https://EconPapers.repec.org/RePEc:eee:intfor:v:8:y:1992:i>

## See Also

[Sarima](#)

## Examples

```
library(astsa)
# A seasonal Random-walk model.
sf1 = stan_naive(birth,seasonal = TRUE,iter = 500,chains = 1)
```

---

 stan\_sarima

*Fitting a Multiplicative Seasonal ARIMA model.*


---

## Description

Fitting a SARIMA model in **Stan**.

## Usage

```
stan_sarima(
  ts,
  order = c(1, 0, 0),
  seasonal = c(0, 0, 0),
  xreg = NULL,
  period = 0,
  chains = 4,
  iter = 2000,
  warmup = floor(iter/2),
  adapt.delta = 0.9,
  tree.depth = 10,
  stepwise = TRUE,
  prior_mu0 = NULL,
  prior_sigma0 = NULL,
  prior_ar = NULL,
  prior_ma = NULL,
  prior_sar = NULL,
  prior_sma = NULL,
  prior_breg = NULL,
  series.name = NULL,
  ...
)
```

## Arguments

ts	a numeric or ts object with the univariate time series.
order	A specification of the non-seasonal part of the ARIMA model: the three components (p, d, q) are the AR order, the number of differences, and the MA order.
seasonal	A specification of the seasonal part of the ARIMA model, same as order parameter: the three components (p, d, q) are the seasonal AR order, the degree of seasonal differences, and the seasonal MA order.
xreg	Optionally, a numerical matrix of external regressors, which must have the same number of rows as ts. It should not be a data frame.
period	an integer specifying the periodicity of the time series by default the value frequency(ts) is used.
chains	An integer of the number of Markov Chains chains to be run, by default 4 chains are run.

<code>iter</code>	An integer of total iterations per chain including the warm-up, by default the number of iterations are 2000.
<code>warmup</code>	A positive integer specifying number of warm-up (aka burn-in) iterations. This also specifies the number of iterations used for step-size adaptation, so warm-up samples should not be used for inference. The number of warmup should not be larger than <code>iter</code> and the default is <code>iter/2</code> .
<code>adapt.delta</code>	An optional real value between 0 and 1, the thin of the jumps in a HMC method. By default is 0.9.
<code>tree.depth</code>	An integer of the maximum depth of the trees evaluated during each iteration. By default is 10.
<code>stepwise</code>	If TRUE, will do stepwise selection (faster). Otherwise, it searches over all models. Non-stepwise selection can be very slow, especially for seasonal models.
<code>prior_mu0</code>	The prior distribution for the location parameter in an ARIMA model. By default the value is set NULL, then the default student(7,0,1) prior is used.
<code>prior_sigma0</code>	The prior distribution for the scale parameter in an ARIMA model. By default the value is set NULL, then the default student(7,0,1) prior is used.
<code>prior_ar</code>	The prior distribution for the auto-regressive parameters in an ARIMA model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_ma</code>	The prior distribution for the moving average parameters in an ARIMA model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_sar</code>	The prior distribution for the seasonal auto-regressive parameters in a SARIMA model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_sma</code>	The prior distribution for the seasonal moving average parameters in a SARIMA model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_breg</code>	The prior distribution for the regression coefficient parameters in a ARIMAX model. By default the value is set NULL, then the default student(7,0,1) priors are used.
<code>series.name</code>	an optional string vector with the series names.
<code>...</code>	Further arguments passed to <code>varstan</code> function.

## Details

The function returns a `varstan` object with the fitted model.

If `xreg` option is used, the model by default will cancel the seasonal differences adjusted ( $D = 0$ ).

If a value  $d > 0$  is used, all the regressor variables in `xreg` will be difference as well.

The default priors used in Sarima are:

- $ar \sim \text{normal}(0,0.5)$
- $ma \sim \text{normal}(0,0.5)$
- $mu0 \sim \text{t-student}(0,2.5,6)$

- $\text{sigma0} \sim \text{t-student}(0,1,7)$
- $\text{sar} \sim \text{normal}(0,0.5)$
- $\text{sma} \sim \text{normal}(0,0.5)$
- $\text{breg} \sim \text{t-student}(0,2.5,6)$

### Value

A varstan object with the fitted SARIMA model.

### Author(s)

Asael Alonzo Matamoros

### References

Box, G. E. P. and Jenkins, G.M. (1978). Time series analysis: Forecasting and control. San Francisco: Holden-Day. *Biometrika*, 60(2), 297-303. doi:10.1093/biomet/65.2.297.

Kennedy, P. (1992). Forecasting with dynamic regression models: Alan Pankratz, 1991. *International Journal of Forecasting*. 8(4), 647-648. url: <https://EconPapers.repec.org/RePEc:eee:intfor:v:8:y:1992:i:4>

Hyndman, R. & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*. 26(3), 1-22. doi: 10.18637/jss.v027.i03

### See Also

[garch set\\_prior](#)

### Examples

```
library(astsa)
# Declare a multiplicative seasonal ARIMA model for the birth data.
sf1 = stan_sarima(birth,order = c(0,1,2),
                 seasonal = c(1,1,1),iter = 500,chains = 1)

#Declare an Dynamic Harmonic Regression model for the birth data.
sf2 = stan_sarima(birth,order = c(1,0,1),
                 xreg = fourier(birth,K = 2),iter = 500,chains = 1)
```



---

 stan\_ssm

*Fitting an Additive linear State space model.*


---

### Description

Fitting an Additive linear State space model in **Stan**.

### Usage

```
stan_ssm(
  ts,
  trend = FALSE,
  damped = FALSE,
  seasonal = FALSE,
  xreg = NULL,
  period = 0,
  genT = FALSE,
  chains = 4,
  iter = 2000,
  warmup = floor(iter/2),
  adapt.delta = 0.9,
  tree.depth = 10,
  stepwise = TRUE,
  prior_sigma0 = NULL,
  prior_level = NULL,
  prior_level1 = NULL,
  prior_trend = NULL,
  prior_trend1 = NULL,
  prior_damped = NULL,
  prior_seasonal = NULL,
  prior_seasonal1 = NULL,
  prior_breg = NULL,
  prior_df = NULL,
  series.name = NULL,
  ...
)
```

### Arguments

ts	a numeric or ts object with the univariate time series.
trend	a boolean value to specify a trend local level model. By default is FALSE.
damped	a boolean value to specify a damped trend local level model. By default is FALSE. If trend option is FALSE then damped is set to FALSE automatically.
seasonal	a boolean value to specify a seasonal local level model. By default is FALSE.
xreg	Optionally, a numerical matrix of external regressors, which must have the same number of rows as ts. It should not be a data frame.

period	an integer specifying the periodicity of the time series by default the value frequency(ts) is used.
genT	a boolean value to specify for a generalized t-student SSM model.
chains	An integer of the number of Markov Chains chains to be run, by default 4 chains are run.
iter	An integer of total iterations per chain including the warm-up, by default the number of iterations are 2000.
warmup	A positive integer specifying number of warm-up (aka burn-in) iterations. This also specifies the number of iterations used for step-size adaptation, so warm-up samples should not be used for inference. The number of warmup should not be larger than iter and the default is iter/2.
adapt.delta	An optional real value between 0 and 1, the thin of the jumps in a HMC method. By default is 0.9.
tree.depth	An integer of the maximum depth of the trees evaluated during each iteration. By default is 10.
stepwise	If TRUE, will do stepwise selection (faster). Otherwise, it searches over all models. Non-stepwise selection can be very slow, especially for seasonal models.
prior_sigma0	The prior distribution for the scale parameter in an SSM model. By default the value is set NULL, then the default student(7,0,1) prior is used.
prior_level	The prior distribution for the level parameter in a SSM model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_level1	The prior distribution for the initial level parameter in a SSM model. By default the value is set NULL, then the default student(6,0,2.5) priors are used.
prior_trend	The prior distribution for the trend parameter in a SSM model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_trend1	The prior distribution for the initial trend parameter in a SSM model. By default the value is set NULL, then the default student(6,0,2.5) priors are used.
prior_damped	The prior distribution for the damped trend parameter in a SSM model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_seasonal	The prior distribution for the seasonal parameter in a SSM model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_seasonal1	The prior distribution for the initial seasonal parameters in a SSM model. The prior is specified for the first m seasonal parameters, where m is the periodicity of the defined time series. By default the value is set NULL, then the default normal(0,0.5) priors are used.
prior_breg	The prior distribution for the regression coefficient parameters in a ARMAX model. By default the value is set NULL, then the default student(7,0,1) priors are used.
prior_df	The prior distribution for the degree freedom parameters in a t-student innovations SSM model. By default the value is set NULL, then the default gamma(2,0.1) priors are used.
series.name	an optional string vector with the series names.
...	Further arguments passed to varstan function.

## Details

The function returns a `varstan` object with the fitted model.

By default the `ssm()` function generates a local level model (or a `ets("A","N","N")` or exponential smoothing model from the **forecast** package). If `trend` is set `TRUE`, then a local trend `ssm` model is defined (a equivalent `ets("A","A","N")` or Holt model from the **forecast** package). For damped trend models set `damped` to `TRUE`. If `seasonal` is set to `TRUE` a seasonal local level model is defined (a equivalent `ets("A","N","A")` model from the **forecast** package). For a Holt-Winters method (`ets("A","A","A")`) set `Trend` and `seasonal` to `TRUE`.

When `genT` option is `TRUE` a t-student innovations `ssm` model (see Ardia (2010)) is generated see Fonseca, et. al (2019) for more details.

The default priors used in a `ssm()` model are:

- `level` ~ `normal(0,0.5)`
- `Trend` ~ `normal(0,0.5)`
- `damped` ~ `normal(0,0.5)`
- `Seasonal` ~ `normal(0,0.5)`
- `sigma0` ~ `t-student(0,1,7)`
- `level1` ~ `normal(0,1)`
- `trend1` ~ `normal(0,1)`
- `seasonal1` ~ `normal(0,1)`
- `dfv` ~ `gamma(2,0.1)`
- `breg` ~ `t-student(0,2.5,6)`

For changing the default prior use the function `set_prior()`.

## Value

A `varstan` object with the fitted SSM model.

## Author(s)

Asael Alonzo Matamoros.

## References

Fonseca, T. and Cequeira, V. and Migon, H. and Torres, C. (2019). The effects of degrees of freedom estimation in the Asymmetric GARCH model with Student-t Innovations. *arXiv* doi: arXiv: 1910.01398.

## See Also

[Sarima](#) [auto.arima](#) [set\\_prior](#) [garch](#)

**Examples**

```
# Declaring a local level model for the ipc data.
sf1 = stan_ssm(ipc,iter = 500,chains = 1)

# Declaring a Holt model for the ipc data.
sf2 = stan_ssm(ipc,trend = TRUE,damped = TRUE,iter = 500,chains = 1)
```

---

 stan\_SVM

*Fitting a Stochastic volatility model*


---

**Description**

Fitting a Stochastic Volatility model (SVM) in **Stan**.

**Usage**

```
stan_SVM(
  ts,
  arma = c(0, 0),
  xreg = NULL,
  chains = 4,
  iter = 2000,
  warmup = floor(iter/2),
  adapt.delta = 0.9,
  tree.depth = 10,
  stepwise = TRUE,
  prior_mu0 = NULL,
  prior_sigma0 = NULL,
  prior_ar = NULL,
  prior_ma = NULL,
  prior_alpha = NULL,
  prior_beta = NULL,
  prior_breg = NULL,
  series.name = NULL,
  ...
)
```

**Arguments**

ts	a numeric or ts object with the univariate time series.
arma	Optionally, a specification of the ARMA model,same as order parameter: the two components (p, q) are the AR order,and the MA order.
xreg	Optionally, a numerical matrix of external regressors, which must have the same number of rows as ts. It should not be a data frame.

<code>chains</code>	An integer of the number of Markov Chains chains to be run, by default 4 chains are run.
<code>iter</code>	An integer of total iterations per chain including the warm-up, by default the number of iterations are 2000.
<code>warmup</code>	A positive integer specifying number of warm-up (aka burn-in) iterations. This also specifies the number of iterations used for step-size adaptation, so warm-up samples should not be used for inference. The number of warmup should not be larger than <code>iter</code> and the default is <code>iter/2</code> .
<code>adapt.delta</code>	An optional real value between 0 and 1, the thin of the jumps in a HMC method. By default is 0.9.
<code>tree.depth</code>	An integer of the maximum depth of the trees evaluated during each iteration. By default is 10.
<code>stepwise</code>	If TRUE, will do stepwise selection (faster). Otherwise, it searches over all models. Non-stepwise selection can be very slow, especially for seasonal models.
<code>prior_mu0</code>	The prior distribution for the location parameter in an SVM model. By default the value is set NULL, then the default normal(0,1) prior is used.
<code>prior_sigma0</code>	The prior distribution for the scale parameter in an SVM model. By default the value is set NULL, then the default student(7,0,1) prior is used.
<code>prior_ar</code>	The prior distribution for the auto-regressive parameters in an ARMA model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_ma</code>	The prior distribution for the moving average parameters in an ARMA model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_alpha</code>	The prior distribution for the arch parameters in a GARCH model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_beta</code>	The prior distribution for the GARCH parameters in a GARCH model. By default the value is set NULL, then the default normal(0,0.5) priors are used.
<code>prior_breg</code>	The prior distribution for the regression coefficient parameters in a ARIMAX model. By default the value is set NULL, then the default student(7,0,1) priors are used.
<code>series.name</code>	an optional string vector with the series names.
<code>...</code>	Further arguments passed to <code>varstan</code> function.

**Details**

The function returns a `varstan` object with the fitted model.

**Value**

A `varstan` object with the fitted SVM model.

**Author(s)**

Asael Alonzo Matamoros

**References**

- Sangjoon, K. and Shephard, N. and Chib, S. (1998). Stochastic Volatility: Likelihood Inference and Comparison with ARCH Models. *Review of Economic Studies*. 65(1), 361-93. url: <https://www.jstor.org/stable/256>
- Tsay, R. (2010). Analysis of Financial Time Series. *Wiley-Interscience*. 978-0470414354, second edition.
- Shumway, R.H. and Stoffer, D.S. (2010). Time Series Analysis and Its Applications: With R Examples. *Springer Texts in Statistics*. isbn: 9781441978646. First edition.

**See Also**

[garch set\\_prior](#)

**Examples**

```
# Declares a SVM model for the IPC data
sf1 = stan_SVM(ipc, arma = c(1,1), iter = 500, chains = 1)
```

---

student

*Define a t student prior distribution*

---

**Description**

student(mu,sd)

**Usage**

```
student(mu = 0, sd = 1, df = 5)
```

**Arguments**

mu	the location parameter mu
sd	the standard deviation parameter sigma
df	the degree freedom parameter df

**Details**

Define a t student prior distribution using the hyper parameters mu, sigma and df as degree freedom, by default a standard t-student(0,1,5) distribution with 5 degree freedom is return.

**Value**

a numerical vector interpreted as a prior in Stan

---

summary.varstan	<i>Summary method for a varstan object</i>
-----------------	--

---

**Description**

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

**Usage**

```
## S3 method for class 'varstan'
summary(object, robust = FALSE, prob = 0.9, ...)
```

**Arguments**

object	A varstan object.
robust	A boolean value, if its TRUE it returns the median of the posterior distribution, And if its FALSE it returns the mean, by default is the FALSE value
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%.
...	Further arguments passed to summary.

**Value**

A data.frame with the posterior mean, standard error, credible intervals, effective sample size (ess),and Rhat for all the model parameters in a varstan model, if robust is TRUE then the posterior mean and standard error, are replaced by the posterior mean and MAD.

**Author(s)**

Asael Alonzo Matamoros.

---

SVM	<i>Constructor of an Stochastic volatility model object</i>
-----	---

---

**Description**

Constructor of the Stochastic Volatility model (SVM) for Bayesian estimation in **Stan**.

**Usage**

```
SVM(ts, arma = c(0,0), xreg = NULL, series.name = NULL)
```

**Arguments**

<code>ts</code>	a numeric or <code>ts</code> object with the univariate time series.
<code>arma</code>	Optionally, a specification of the ARMA model, same as order parameter: the two components ( <code>p</code> , <code>q</code> ) are the AR order, and the MA order.
<code>xreg</code>	Optionally, a numerical matrix of external regressors, which must have the same number of rows as <code>ts</code> . It should not be a data frame.
<code>series.name</code>	an optional string vector with the time series names.

**Details**

The function returns a list with the data for running `stan()` function of **rstan** package.

**Value**

The function returns a list with the data for running `stan()` function of **rstan** package.

**Author(s)**

Asael Alonzo Matamoros

**References**

- Sangjoon, K. and Shephard, N. and Chib, S. (1998). Stochastic Volatility: Likelihood Inference and Comparison with ARCH Models. *Review of Economic Studies*. 65(1), 361-93. url: <https://www.jstor.org/stable/256>
- Tsay, R. (2010). Analysis of Financial Time Series. *Wiley-Interscience*. 978-0470414354, second edition.
- Shumway, R.H. and Stoffer, D.S. (2010). Time Series Analysis and Its Applications: With R Examples. *Springer Texts in Statistics*. isbn: 9781441978646. First edition.

**See Also**

[garch set\\_prior](#)

**Examples**

```
# Declares a SVM model for the IPC data

model = SVM(ipc, arma = c(1,1))
model
```



---

uniform	<i>Define a uniform prior distribution</i>
---------	--

---

**Description**

```
uniform(shape1,shape2)
```

**Usage**

```
uniform(min = 0, max = 1)
```

**Arguments**

min	the first form parameter
max	the second form parameter

**Details**

Define a beta prior distribution using the hyper parameters min and max, by default a uniform(0,1) distribution is return.

**Value**

a numerical vector interpreted as a prior in Stan

---

varstan	<i>Constructor of a varstan object.</i>
---------	---

---

**Description**

Constructor of the varstan object for Bayesian estimation in **Stan**.

**Usage**

```
varstan(model,chains=4,iter=2000,warmup=floor(iter/2),
        adapt.delta = 0.90,tree.depth =10,...)
```

**Arguments**

model	One of the varstan model classes defined in the package.
chains	An integer of the number of Markov Chains chains to be run, by default 4 chains are run.
iter	An integer of total iterations per chain including the warm-up, by default the number of iterations are 2000.

warmup	A positive integer specifying number of warm-up (aka burn-in) iterations. This also specifies the number of iterations used for stepsize adaptation, so warmup samples should not be used for inference. The number of warmup should not be larger than <code>iter</code> and the default is <code>iter/2</code> .
<code>adapt.delta</code>	An optional real value between 0 and 1, the thin of the jumps in a HMC method. By default is 0.9.
<code>tree.depth</code>	An integer of the maximum depth of the trees evaluated during each iteration. By default is 10.
...	Further arguments passed to <code>rstan()</code> function.

## Details

The function estimates one of the defined models in **Stan** using the `stan()` function for sampling. This is the principal package's function and the link with **Stan**, this function fits the posterior distribution of every parameter for a defined model using a HMC method.

Every estimated model become a `varstan` object, with different methods for summary, diagnostic, forecast and plotting.

### Defining priors

Default priors are chosen to be non or very weakly informative so that their influence on the results will. However, after getting more familiar with Bayesian statistics, I recommend you to start thinking about reasonable informative priors for your model parameters.

Those can be changed using the function `set_prior()` before estimating the model with the `varstan()` function. For checking the defined priors use `get_prior()` and `report()` functions.

### Adjusting the sampling behavior of Stan

In addition to choosing the number of iterations, warmup samples, and chains, users can control the behavior of the NUTS sampler, by using the `control` argument. The most important reason to use `control` is to decrease (or eliminate at best) the number of divergent transitions that cause a bias in the obtained posterior samples. Whenever you see the warning "There were x divergent transitions after warmup." you should really think about increasing `adapt_delta`. Increasing `adapt_delta` will slow down the sampler but will decrease the number of divergent transitions threatening the validity of your posterior samples.

Another problem arises when the depth of the tree being evaluated in each iteration is exceeded. This is less common than having divergent transitions, but may also bias the posterior samples. When it happens, **Stan** will throw out a warning suggesting to increase `max_treedepth`. For more details on the `control` argument see [stan](#).

## Value

a `varstan` object with the estimated time series model.

A `varstan` object is a list that contains the following values:

- `Stanfit` a Stanfit object returned by `rstan` package.
- `stan.parmaters` The parameters used in Stan for the sample.
- `model` The defined model for the time series.
- `series.name` The time series' name.
- `ts` The provided time series data.

**Author(s)**

Asael Alonzo Matamoros

**References**

Carpenter, B. and Gelman, A. and Hoffman, D. and Lee, D. and Goodrich, B. and Betancourt, M. and Brubaker, and Guo, L. and Riddell. 2017. Stan: A probabilistic programming language. *Journal of Statistical Software* 76(1). doi: 10.18637/jss.v076.i01.

Stan Development Team. (2018). Stan Modeling Language Users Guide and Reference Manual, Version 2.18.0. url: <http://mc-stan.org>.

Paul-Christian Buerkner (2017). brms: An R Package for Bayesian Multilevel Models Using Stan. *Journal of Statistical Software*, 80(1), 1-28. doi:10.18637/jss.v080.i01

Hyndman, R. & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software*. 26(3), 1-22. doi: 10.18637/jss.v027.i03.

**See Also**

[rstan:stan](#).

**Examples**

```
library(astsa)
# Fitting a seasonal ARIMA model
mod1 = Sarima(birth,order = c(0,1,2),seasonal = c(1,1,1))
fit1 = varstan(mod1,iter = 500,chains = 1)
fit1

# Fitting a GARCH(1,1) model
dat = garch(ipc,order = c(1,1,0))
fit2 = varstan(dat,iter = 500,chains = 1)
fit2
```

---

waic.varstan

*Widely Applicable Information Criterion (WAIC)*

---

**Description**

Compute the widely applicable information criterion (WAIC) based on the posterior likelihood using the **loo** package. For more details see [waic](#).

**Usage**

```
## S3 method for class 'varstan'
waic(x, ...)
```

**Arguments**

x                    A varstan object  
...                  additional values need in waic methods

**Details**

See the `loo_compare` function of the **loo** package for more details on model comparisons.

**Value**

An object of class `loo`. With the estimates of the Watanabe-Akaike Information criteria.

**References**

Vehtari, A., Gelman, A., & Gabry J. (2016). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *In Statistics and Computing*, doi:10.1007/s11222-016-9696-4.

Gelman, A., Hwang, J., & Vehtari, A. (2014). Understanding predictive information criteria for Bayesian models. *Statistics and Computing*. 24, 997-1016.

Watanabe, S. (2010). Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. *The Journal of Machine Learning Research*. 11, 3571-3594.

**Examples**

```
library(astsa)
model = Sarima(birth,order = c(0,1,2),seasonal = c(1,1,1))
fit1 = varstan(model,iter = 500,chains = 1)

waic1 = waic(fit1)
waic1
```

# Index

- \* **datasets**
  - air, [6](#)
  - aust, [8](#)
  - birth, [16](#)
  - dengbp, [19](#)
  - ipc, [36](#)
  - oildata, [47](#)
- \* **forecast**
  - fourier, [24](#)
  
- aic, [5](#)
- AICc, [5](#)
- air, [6](#)
- as.stan, [7](#)
- aust, [8](#)
- auto.arima, [27](#), [33](#), [35](#), [39](#), [64](#), [67](#), [70](#), [73](#), [75](#), [83](#)
- auto.sarima, [8](#), [24](#)
- autoplot.ts, [11](#)
- autoplot.varstan, [12](#)
  
- bayes\_factor (bayes\_factor.varstan), [13](#)
- bayes\_factor.varstan, [13](#)
- bayesforecast (bayesforecast-package), [4](#)
- bayesforecast-package, [4](#)
- beta, [14](#)
- bic, [15](#)
- birth, [16](#)
- bridge\_sampler
  - (bridge\_sampler.varstan), [16](#)
- bridge\_sampler.stanfit, [16](#)
- bridge\_sampler.varstan, [16](#)
  
- cauchy, [17](#)
- check\_residuals, [18](#)
- chisq, [19](#)
  
- dengbp, [19](#)
  
- ets, [23](#)
- exponential, [20](#)
  
- extract\_stan, [20](#)
  
- fitted.varstan, [21](#)
- forecast (forecast.varstan), [22](#)
- forecast.ts, [23](#)
- forecast.varstan, [22](#)
- fortify, [11](#), [12](#)
- fortify.ts (autoplot.ts), [11](#)
- fourier, [24](#)
  
- gamma, [25](#)
- garch, [26](#), [33](#), [35](#), [39](#), [61](#), [64](#), [70](#), [73](#), [75](#), [80](#), [83](#), [86](#), [88](#)
- geom\_forecast, [11](#)
- get\_parameters, [28](#)
- get\_prior, [28](#)
- ggacf, [29](#)
- gghist, [30](#)
- ggnorm, [31](#)
- ggpacf, [31](#)
- ggplot, [44](#)
  
- Holt, [32](#)
- Hw, [33](#)
  
- inverse.chisq, [35](#)
- inverse.gamma, [36](#)
- ipc, [36](#)
  
- jeffrey, [37](#)
  
- laplace, [37](#)
- LKJ, [38](#)
- LocalLevel, [38](#)
- log\_lik (log\_lik.varstan), [41](#)
- log\_lik.varstan, [41](#)
- loglik, [40](#)
- loo (loo.varstan), [42](#)
- loo.varstan, [42](#)
- loo\_compare(), [42](#)

mcmc\_plot (mcmc\_plot.varstan), 43  
 mcmc\_plot.varstan, 43  
 model, 44  
  
 naive, 45  
 normal, 46  
  
 oildata, 47  
  
 pareto-k-diagnostic, 42  
 plot.ts, 12  
 plot.varstan, 47  
 posterior\_epred  
     (posterior\_epred.varstan), 48  
 posterior\_epred.varstan, 48  
 posterior\_interval, 49  
 posterior\_predict, 51  
 posterior\_predict  
     (posterior\_predict.varstan), 50  
 posterior\_predict.varstan, 22, 48, 50  
 predictive\_error  
     (predictive\_error.varstan), 51  
 predictive\_error.varstan, 51  
 print.garch, 52  
 print.Holt, 53  
 print.Hw, 53  
 print.LocalLevel, 54  
 print.naive, 54  
 print.Sarima, 55  
 print.ssm, 55  
 print.SVM, 56  
 print.varstan, 56  
 prior\_summary (prior\_summary.varstan),  
     57  
 prior\_summary.varstan, 57  
 psis(), 42  
  
 report, 58  
 residuals.varstan, 59  
 rstan:stan, 91  
  
 Sarima, 24, 27, 33, 35, 39, 46, 59, 64, 67, 70,  
     73, 75, 77, 83  
 seasonaldummy, 25  
 seed, 23, 49, 50  
 set\_prior, 27, 33, 35, 39, 61, 61, 64, 67, 70,  
     73, 75, 80, 83, 86, 88  
 ssm, 62  
 stan, 90  
  
 stan\_garch, 64  
 stan\_Holt, 67  
 stan\_Hw, 70  
 stan\_LocalLevel, 73  
 stan\_naive, 75  
 stan\_sarima, 78  
 stan\_ssm, 81  
 stan\_SVM, 84  
 stlf, 23  
 student, 86  
 summary.varstan, 87  
 SVM, 87  
  
 uniform, 89  
  
 varstan, 22, 59, 89  
  
 waic, 91  
 waic (waic.varstan), 91  
 waic.varstan, 91