Package 'stochvol'

June 26, 2019

Encoding UTF-8

Type Package

Title Efficient Bayesian Inference for Stochastic Volatility (SV) Models

Version 2.0.4

Description Efficient algorithms for fully Bayesian estimation of stochastic volatility (SV) models via Markov chain Monte Carlo (MCMC) methods. Methodological details are given in Kastner and Frühwirth-Schnatter (2014) <doi:10.1016/j.csda.2013.01.002>; the most common use cases are described in Kastner (2016) <doi:10.18637/jss.v069.i05>. Also incorporates SV with leverage.

License GPL (>= 2)

Depends R (>= 3.0.2), coda

Imports Rcpp (>= 0.11), graphics, stats, utils

Suggests mvtnorm

LinkingTo Rcpp, RcppArmadillo (>= 0.4)

RoxygenNote 6.1.1

BuildResaveData best

NeedsCompilation yes

Author Gregor Kastner [aut] (<https://orcid.org/0000-0002-8237-8271>), Darjus Hosszejni [aut, cre] (<https://orcid.org/0000-0002-3803-691X>)

Maintainer Darjus Hosszejni <darjus.hosszejni@wu.ac.at>

Repository CRAN

Date/Publication 2019-06-26 10:30:03 UTC

R topics documented:

`stochvol-package` *Efficient Bayesian Inference for Stochastic Volatility (SV) Models*

Description

This package provides an efficient algorithm for fully Bayesian estimation of stochastic volatility (SV) models via Markov chain Monte Carlo (MCMC) methods. Methodological details are given in Kastner and Frühwirth-Schnatter (2014); the most common use cases are described in Kastner (2016). Recently, the package has been extended to allow for the leverage effect.

Details

Bayesian inference for stochastic volatility models using MCMC methods highly depends on actual parameter values in terms of sampling efficiency. While draws from the posterior utilizing the standard centered parameterization break down when the volatility of volatility parameter in the latent state equation is small, non-centered versions of the model show deficiencies for highly persistent latent variable series. The novel approach of ancillarity-sufficiency interweaving (Yu and Meng, 2011) has recently been shown to aid in overcoming these issues for a broad class of multilevel models. This package provides software for "combining best of different worlds" which allows for inference for parameter constellations that have previously been infeasible to estimate without the need to select a particular parameterization beforehand.

Note

This package is currently in active development. Your comments, suggestions and requests are warmly welcome!

Author(s)

Gregor Kastner <gregor.kastner@wu.ac.at>, Darjus Hosszejni <darjus.hosszejni@wu.ac.at>

References

Kastner, G. and Frühwirth-Schnatter, S. (2014). Ancillarity-Sufficiency Interweaving Strategy (ASIS) for Boosting MCMC Estimation of Stochastic Volatility Models. *Computational Statistics & Data Analysis*, **76**, 408–423, http://dx.doi.org/10.1016/j.csda.2013.01.002.

Kastner, G. (2016). Dealing with Stochastic Volatility in Time Series Using the R Package stochvol. *Journal of Statistical Software*, **69**(5), 1–30, http://dx.doi.org/10.18637/jss.v069.i05.

Yu, Y. and Meng, X.-L. (2011). To Center or Not to Center: That is Not the Question—An Ancillarity-Suffiency Interweaving Strategy (ASIS) for Boosting MCMC Efficiency. *Journal of Computational and Graphical Statistics*, **20**(3), 531–570, http://dx.doi.org/10.1198/jcgs.2011.203main.

Examples

```
## Simulate a highly persistent SV process
sim <- svsim(500, mu = -10, phi = 0.99, sigma = 0.2)

## Obtain 4000 draws from the sampler (that's too little!)
draws <- svsample(sim$y, draws = 4000, burnin = 100, priormu = c(-10, 1),
                  priorphi = c(20, 1.2), priorsigma = 0.2)

## Predict 20 days ahead
fore <- predict(draws, 20)

## plot the results
plot(draws, forecast = fore)

## Not run:
## Simulate an SV process with leverage
sim <- svsim(500, mu = -10, phi = 0.95, sigma = 0.2, rho=-0.5)

## Obtain 8000 draws from the sampler (that's too little!)
draws <- svlsample(sim$y, draws = 4000, burnin = 3000, priormu = c(-10, 1),
                   priorphi = c(20, 1.2), priorsigma = 0.2,
                   priorrho = c(1, 1))

## Predict 20 days ahead
fore <- predict(draws, 20)

## plot the results
plot(draws, forecast = fore)

## End(Not run)
```

`arpredict` *Dynamic prediction for the AR-SV model (deprecated)*

Description

Simulates draws from the posterior predictive density of a fitted AR-SV model. DEPRECATED: please use `predict.svdraws` on `svdraws` objects resulting from model estimation with an autoregressive mean structure.

Usage

```
arpredict(object, volpred)
```

Arguments

`object`	`svdraws` object as returned from `svsample`.
`volpred`	`svpredict` object as returned from `predict.svdraws`.

Value

Returns an object of class `c("distpredict", "mcmc")` containing simulations from the posterior predictive density of `y_(n+1),...,y_(n+steps)`.

Note

You can use the usual `coda` methods for `mcmc` objects to print, plot, or summarize the predictions.

Author(s)

Gregor Kastner <gregor.kastner@wu.ac.at>

See Also

`predict.svdraws`.

Examples

```
## Not run:
data(exrates)
y <- exrates$USD

## Fit AR(1)-SV model to EUR-USD exchange rates
res <- svsample(y, designmatrix = "ar1")

## Use predict.svdraws to obtain predictive volatilities
ahead <- 100
preds <- predict(res, steps = ahead)
predvol <- preds$h
class(predvol) <- "svpredict"

## Use arpredict to obtain draws from the posterior predictive
preddraws <- arpredict(res, predvol)

## Calculate predictive quantiles
predquants <- apply(preddraws, 2, quantile, c(.1, .5, .9))
```

```
## Visualize
ts.plot(y, xlim = c(length(y) - ahead, length(y) + ahead),
ylim = range(predquants))
for (i in 1:3) {
 lines((length(y) + 1):(length(y) + ahead), predquants[i,],
       col = 3, lty = c(2, 1, 2)[i])
}

## End(Not run)
```

---

exrates *Euro exchange rate data*

Description

The data set contains the daily bilateral prices of one Euro in 23 currencies from January 3, 2000, until April 4, 2012. Conversions to New Turkish Lira and Fourth Romanian Leu have been incorporated.

Source

ECB Statistical Data Warehouse (http://sdw.ecb.europa.eu)

See Also

svsample

Examples

```
## Not run:
data(exrates)
dat <- logret(exrates$USD, demean = TRUE)  ## de-meaned log-returns
res <- svsample(dat)                       ## run MCMC sampler
plot(res, forecast = 100)                  ## display results

## End(Not run)
```

extractors *Common Extractors for 'svdraws' Objects*

Description

Some simple extractors returning the corresponding element of an svdraws object.

Usage

```
para(x)

latent(x)

latent0(x)

priors(x)

thinning(x)

runtime(x)
```

Arguments

x	svdraws object.

Value

The return value depends on the actual funtion:

para(x)	extracts the parameter draws and returns them as an mcmc object.
latent(x)	extracts the latent contemporaneous log-volatility draws and returns them as an mcmc object.
latent0(x)	extracts the latent initial log-volatility draws and returns as an mcmc object.
priors(x)	extracts the prior parameters used and returns them in a list.
thinning(x)	extracts the thinning parameters used and returns them in a list.
runtime(x)	extracts the runtime and returns it as a proc_time object.

Author(s)

Gregor Kastner <gregor.kastner@wu.ac.at>

logret *Computes the Log Returns of a Time Series*

Description

`logret` computes the log returns of a time series, with optional de-meaning and/or standardization.

Usage

```
logret(dat, demean = FALSE, standardize = FALSE, ...)

## Default S3 method:
logret(dat, demean = FALSE, standardize = FALSE, ...)
```

Arguments

`dat`	The raw data.
`demean`	Logical value indicating whether the data should be de-meaned.
`standardize`	Logical value indicating whether the data should be standardized (in the sense that each component series has an empirical variance equal to one).
`...`	Ignored.

Value

Log returns of the (de-meaned / standardized) data.

Methods (by class)

- `default`: Log returns of vectors

paradensplot *Probability Density Function Plot for the Parameter Posteriors*

Description

Displays a plot of the density estimate for the posterior distribution of the parameters `mu`, `phi`, `sigma` (and potentially `nu` or `rho`), computed by the `density` function.

Usage

```
paradensplot(x, showobs = TRUE, showprior = TRUE, showxlab = TRUE,
  mar = c(1.9, 1.9, 1.9, 0.5), mgp = c(2, 0.6, 0), simobj = NULL,
  ...)
```

Arguments

x	svdraws or svldraws object.
showobs	logical value, indicating whether the observations should be displayed along the x-axis. If many draws have been obtained, the default (TRUE) can render the plotting to be quite slow, and you might want to try setting showobs to FALSE.
showprior	logical value, indicating whether the prior distribution should be displayed. The default value is TRUE.
showxlab	logical value, indicating whether the x-axis should be labelled with the number of iterations and the bandwith obtained from density. The default value is TRUE.
mar	numerical vector of length 4, indicating the plot margins. See par for details. The default value is c(1.9,1.9, 1.9, 0.5), which is slightly smaller than the R-defaults.
mgp	numerical vector of length 3, indicating the axis and label positions. See par for details. The default value is c(2, 0.6, 0), which is slightly smaller than the R-defaults.
simobj	object of class svsim as returned by the SV simulation function svsim. If provided, "true" data generating values will be added to the plots.
...	further arguments are passed on to the invoked plot function.

Details

paradensplot is modeled after densplot in the coda package, with some modifications for parameters that have (half-)bounded support.

Value

Called for its side effects. Returns argument x invisibly.

Note

You can call this function directly, but it is more commonly called by the plot.svdraws method.

Author(s)

Gregor Kastner <gregor.kastner@wu.ac.at>

See Also

Other plotting: paratraceplot.svdraws, paratraceplot, plot.svdraws, plot.svpredict, volplot

paratraceplot *Trace Plot of MCMC Draws from the Parameter Posteriors*

Description

Generic function for plotting iterations vs. sampled parameter values. A detailed help for the method implemented in **stochvol** can be found in `paratraceplot.svdraws`.

Usage

```
paratraceplot(x, ...)
```

Arguments

`x`	An object used to select a method.
`...`	Further arguments passed to or from other methods.

Value

Called for its side effects. Returns argument `x` invisibly.

See Also

Other plotting: `paradensplot`, `paratraceplot.svdraws`, `plot.svdraws`, `plot.svpredict`, `volplot`

paratraceplot.svdraws *Trace Plot of MCMC Draws from the Parameter Posteriors*

Description

Displays a plot of iterations vs. sampled values the parameters `mu`, `phi`, `sigma` (and potentially `nu` or `rho`), with a separate plot per variable.

Usage

```
## S3 method for class 'svdraws'
paratraceplot(x, mar = c(1.9, 1.9, 1.9, 0.5),
  mgp = c(2, 0.6, 0), simobj = NULL, ...)
```

Arguments

x	svdraws or svldraws object.
mar	numerical vector of length 4, indicating the plot margins. See par for details. The default value is c(1.9,1.9, 1.9, 0.5), which is slightly smaller than the R-defaults.
mgp	numerical vector of length 3, indicating the axis and label positions. See par for details. The default value is c(2, 0.6, 0), which is slightly smaller than the R-defaults.
simobj	object of class svsim as returned by the SV simulation function svsim. If provided, "true" data generating values will be added to the plots.
...	further arguments are passed on to the invoked matplot function.

Details

paratraceplot is modeled after traceplot in the coda package, with very minor modifications.

Value

Called for its side effects. Returns argument x invisibly.

Note

You can call this function directly, but it is more commonly called by the plot.svdraws method.

Author(s)

Gregor Kastner <gregor.kastner@wu.ac.at>

See Also

Other plotting: paradensplot, paratraceplot, plot.svdraws, plot.svpredict, volplot

---

plot.svdraws *Graphical Summary of the Posterior Distribution*

Description

plot.svdraws and plot.svldraws generate some plots visualizing the posterior distribution and can also be used to display predictive distributions of future volatilities.

Usage

```
## S3 method for class 'svdraws'
plot(x, forecast = NULL, dates = NULL,
  show0 = FALSE, showobs = TRUE, showprior = TRUE, col = NULL,
  forecastlty = NULL, tcl = -0.4, mar = c(1.9, 1.9, 1.7, 0.5),
  mgp = c(2, 0.6, 0), simobj = NULL, newdata = NULL, ...)

## S3 method for class 'svldraws'
plot(x, forecast = NULL, dates = NULL,
  show0 = FALSE, showobs = TRUE, showprior = TRUE, col = NULL,
  forecastlty = NULL, tcl = -0.4, mar = c(1.9, 1.9, 1.7, 0.5),
  mgp = c(2, 0.6, 0), simobj = NULL, newdata = NULL, ...)
```

Arguments

x	svdraws or svldraws object.
forecast	nonnegative integer or object of class svpredict, as returned by predict.svdraws. If an integer greater than 0 is provided, predict.svdraws is invoked to obtain the forecast-step-ahead prediction. The default value is 0.
dates	vector of length ncol(x$latent), providing optional dates for labelling the x-axis. The default value is NULL; in this case, the axis will be labelled with numbers.
show0	logical value, indicating whether the initial volatility exp(h_0/2) should be displayed. The default value is FALSE. Only available for inputs x of class svdraws.
showobs	logical value, indicating whether the observations should be displayed along the x-axis. If many draws have been obtained, the default (TRUE) can render the plotting to be quite slow, and you might want to try setting showobs to FALSE.
showprior	logical value, indicating whether the prior distribution should be displayed. The default value is TRUE.
col	vector of color values (see par) used for plotting the quantiles. The default value NULL results in gray lines for all quantiles expect the median, which is displayed in black.
forecastlty	vector of line type values (see par) used for plotting quantiles of predictive distributions. The default value NULL results in dashed lines.
tcl	The length of tick marks as a fraction of the height of a line of text. See par for details. The default value is -0.4, which results in slightly shorter tick marks than usual.
mar	numerical vector of length 4, indicating the plot margins. See par for details. The default value is c(1.9,1.9, 1.9, 0.5), which is slightly smaller than the R-defaults.
mgp	numerical vector of length 3, indicating the axis and label positions. See par for details. The default value is c(2, 0.6, 0), which is slightly smaller than the R-defaults.
simobj	object of class svsim as returned by the SV simulation function svsim. If provided, the "true" data generating values will be added to the plots.

newdata	corresponds to parameter newdata in predict.svdraws. *Only if* forecast *is a positive integer and* predict.svdraws *needs a* newdata *object.* Corresponds to input parameter designmatrix in svsample and svlsample. A matrix of regressors with number of rows equal to parameter forecast.
...	further arguments are passed on to the invoked plotting functions.

Details

These functions set up the page layout and call volplot, paratraceplot and paradensplot.

Value

Called for its side effects. Returns argument x invisibly.

Note

In case you want different quantiles to be plotted, use updatesummary on the svdraws object first. An example of doing so is given in the Examples section.

Author(s)

Gregor Kastner <gregor.kastner@wu.ac.at>

See Also

updatesummary, predict.svdraws

Other plotting: paradensplot, paratraceplot.svdraws, paratraceplot, plot.svpredict, volplot

Examples

```
## Simulate a short and highly persistent SV process
sim <- svsim(100, mu = -10, phi = 0.99, sigma = 0.2)

## Obtain 5000 draws from the sampler (that's not a lot)
draws <- svsample(sim$y, draws = 5000, burnin = 1000,
  priormu = c(-10, 1), priorphi = c(20, 1.5), priorsigma = 0.2)

## Plot the latent volatilities and some forecasts
plot(draws, forecast = 10)

## Re-plot with different quantiles
newquants <- c(0.01, 0.05, 0.25, 0.5, 0.75, 0.95, 0.99)
draws <- updatesummary(draws, quantiles = newquants)

plot(draws, forecast = 20, showobs = FALSE, col = seq(along = newquants),
     forecastlty = 3, showprior = FALSE)
```

plot.svpredict *Graphical Summary of the Posterior Predictive Distribution*

Description

plot.svpredict and plot.svlpredict generate some plots visualizing the posterior predictive distribution of future volatilites and future observations.

Usage

```
## S3 method for class 'svpredict'
plot(x, quantiles = c(0.05, 0.25, 0.5, 0.75, 0.95),
  ...)

## S3 method for class 'svlpredict'
plot(x, quantiles = c(0.05, 0.25, 0.5, 0.75, 0.95),
  ...)
```

Arguments

x	svpredict or svlpredict object.
quantiles	Which quantiles to plot? Defaults to c(.05, .25, .5, .75, .95).
...	further arguments are passed on to the invoked ts.plot or boxplot function.

Value

Called for its side effects. Returns argument x invisibly.

Note

Note that svpredict or svlpredict objects can also be used within plot.svdraws for a possibly more useful visualization. See the examples in predict.svdraws and those below for use cases.

Author(s)

Gregor Kastner <gregor.kastner@wu.ac.at>

See Also

Other plotting: paradensplot, paratraceplot.svdraws, paratraceplot, plot.svdraws, volplot

Examples

```
## Simulate a short and highly persistent SV process
sim <- svsim(100, mu = -10, phi = 0.99, sigma = 0.1)

## Obtain 5000 draws from the sampler (that's not a lot)
draws <- svsample(sim$y, draws = 5000, burnin = 1000)

## Predict 10 steps ahead
pred <- predict(draws, 10)

## Visualize the predicted distributions
plot(pred)

## Plot the latent volatilities and some forecasts
plot(draws, forecast = pred)
```

---

predict.svdraws *Prediction of Future Returns and Log-Volatilities*

Description

Simulates draws from the predictive density of the returns and the latent log-volatility process. The same mean model is used for prediction as was used for fitting, which is either a) no mean parameter, b) constant mean, c) AR(k) structure, or d) general Bayesian regression. In the last case, new regressors need to be provided for prediction.

Usage

```
## S3 method for class 'svdraws'
predict(object, steps = 1L, newdata = NULL, ...)

## S3 method for class 'svldraws'
predict(object, steps = 1L, newdata = NULL, ...)
```

Arguments

`object`	`svdraws` or `svldraws` object.
`steps`	*optional* single number, coercible to integer. Denotes the number of steps to forecast.
`newdata`	*only in case d) of the description* corresponds to input parameter `designmatrix` in `svsample` and `svlsample`. A matrix of regressors with number of rows equal to parameter `steps`.
`...`	currently ignored.

Value

Returns an object of class `svpredict`, a list containing two elements:

`h`	`mcmc` object of simulations from the predictive density of `h_(n+1),...,h_(n+steps)`
`y`	`mcmc` object of simulations from the predictive density of `y_(n+1),...,y_(n+steps)`

Note

You can use the resulting object within `plot.svdraws` (see example below), or use the list items in the usual coda methods for `mcmc` objects to print, plot, or summarize the predictions.

Author(s)

Gregor Kastner <gregor.kastner@wu.ac.at>

See Also

`plot.svdraws`, `volplot`.

Examples

```
# Example 1
## Simulate a short and highly persistent SV process
sim <- svsim(100, mu = -10, phi = 0.99, sigma = 0.2)

## Obtain 5000 draws from the sampler (that's not a lot)
draws <- svsample(sim$y, draws = 5000, burnin = 100,
  priormu = c(-10, 1), priorphi = c(20, 1.5), priorsigma = 0.2)

## Predict 10 days ahead
fore <- predict(draws, 10)

## Check out the results
summary(fore$h)
summary(fore$y)
plot(draws, forecast = fore)


# Example 2
## Simulate now an SV process with an AR(1) mean structure
len <- 109L
simar <- svsim(len, phi = 0.93, sigma = 0.15, mu = -9)
for (i in 2:len) {
  simar$y[i] <- 0.1 - 0.7 * simar$y[i-1] + simar$vol[i] * rnorm(1)
}

## Obtain 7000 draws
drawsar <- svsample(simar$y, draws = 7000, burnin = 300,
  designmatrix = "ar1", priormu = c(-10, 1), priorphi = c(20, 1.5),
  priorsigma = 0.2)
```

```
## Predict 7 days ahead (using AR(1) mean for the returns)
forear <- predict(drawsar, 7)

## Check out the results
plot(forear)
plot(drawsar, forecast = forear)

## Not run:
# Example 3
## Simulate now an SV process with leverage and with non-zero mean
len <- 96L
regressors <- cbind(rep_len(1, len), rgamma(len, 0.5, 0.25))
betas <- rbind(-1.1, 2)
simreg <- svsim(len, rho = -0.42)
simreg$y <- simreg$y + as.numeric(regressors %*% betas)

## Obtain 12000 draws
drawsreg <- svlsample(simreg$y, draws = 12000, burnin = 3000,
  designmatrix = regressors, priormu = c(-10, 1), priorphi = c(20, 1.5),
  priorsigma = 0.2)

## Predict 5 days ahead using new regressors
predlen <- 5L
predregressors <- cbind(rep_len(1, predlen), rgamma(predlen, 0.5, 0.25))
forereg <- predict(drawsreg, predlen, predregressors)

## Check out the results
summary(forereg$h)
summary(forereg$y)
plot(forereg)
plot(drawsreg, forecast = forereg)

## End(Not run)
```

`svlsample` *Markov Chain Monte Carlo (MCMC) Sampling for the Stochastic Volatility Model with Leverage (SVL)*

Description

`svlsample` simulates from the joint posterior distribution of the SVL parameters `mu`, `phi`, `sigma`, and `rho`, along with the latent log-volatilities `h_1`, . . . , `h_n` and returns the MCMC draws. If a design matrix is provided, simple Bayesian regression can also be conducted.

Usage

```
svlsample(y, draws = 10000, burnin = 1000, designmatrix = NA,
  priormu = c(0, 100), priorphi = c(5, 1.5), priorsigma = 1,
  priorrho = c(4, 4), priorbeta = c(0, 10000), thinpara = 1,
  thinlatent = 1, thintime = NULL, keeptime = "all", quiet = FALSE,
  startpara, startlatent, expert, ...)
```

Arguments

y	numeric vector containing the data (usually log-returns), which must not contain zeros. Alternatively, y can be an svsim object. In this case, the returns will be extracted and a warning is thrown.
draws	single number greater or equal to 1, indicating the number of draws after burn-in (see below). Will be automatically coerced to integer. The default value is 10000.
burnin	single number greater or equal to 0, indicating the number of draws discarded as burn-in. Will be automatically coerced to integer. The default value is 1000.
designmatrix	regression design matrix for modeling the mean. Must have length(y) rows. Alternatively, designmatrix may be a string of the form "arX", where X is a nonnegative integer. To fit a constant mean model, use designmatrix = "ar0" (which is equivalent to designmatrix = matrix(1, nrow = length(y))). To fit an AR(1) model, use designmatrix = "ar1", and so on. If some elements of designmatrix are NA, the mean is fixed to zero (pre-1.2.0 behavior of **stochvol**).
priormu	numeric vector of length 2, indicating mean and standard deviation for the Gaussian prior distribution of the parameter mu, the level of the log-volatility. The default value is c(0, 100), which constitutes a practically uninformative prior for common exchange rate datasets, stock returns and the like.
priorphi	numeric vector of length 2, indicating the shape parameters for the Beta prior distribution of the transformed parameter (phi + 1) / 2, where phi denotes the persistence of the log-volatility. The default value is c(5, 1.5), which constitutes a prior that puts some belief in a persistent log-volatility but also encompasses the region where phi is around 0.
priorsigma	single positive real number, which stands for the scaling of the transformed parameter sigma^2, where sigma denotes the volatility of log-volatility. More precisely, sigma^2 ~ priorsigma * chisq(df = 1). The default value is 1, which constitutes a reasonably vague prior for many common exchange rate datasets, stock returns and the like.
priorrho	numeric vector of length 2, indicating the shape parameters for the Beta prior distribution of the transformed parameter (rho + 1) / 2, where rho denotes the conditional correlation between observation and the increment of the log-volatility. The default value is c(4, 4), which constitutes a slightly informative prior around 0 (the no leverage case) to boost convergence.
priorbeta	numeric vector of length 2, indicating the mean and standard deviation of the Gaussian prior for the regression parameters. The default value is c(0, 10000), which constitutes a very vague prior for many common datasets. Not used if designmatrix is NA.
thinpara	single number greater or equal to 1, coercible to integer. Every thinparath parameter draw is kept and returned. The default value is 1, corresponding to no thinning of the parameter draws i.e. every draw is stored.
thinlatent	single number greater or equal to 1, coercible to integer. Every thinlatentth latent variable draw is kept and returned. The default value is 1, corresponding to no thinning of the latent variable draws, i.e. every draw is kept.
thintime	*Deprecated.* Use 'keeptime' instead.

`keeptime` Either 'all' (the default) or 'last'. Indicates which latent

`quiet` logical value indicating whether the progress bar and other informative output during sampling should be omitted. The default value is `FALSE`, implying verbose output.

`startpara` *optional* named list, containing the starting values for the parameter draws. If supplied, `startpara` must contain four elements named `mu`, `phi`, `sigma`, and `rho`, where `mu` is an arbitrary numerical value, `phi` is a real number between `-1` and `1`, `sigma` is a positive real number, and `rho` is a real number between `-1` and `1`. The default value is equal to the prior mean.

`startlatent` *optional* vector of length `length(y)`, containing the starting values for the latent log-volatility draws. The default value is `rep(-10, length(y))`.

`expert` *optional* named list of expert parameters. For most applications, the default values probably work best. If `expert` is provided, it may contain the following named elements:

`parameterization`: Character string containing values `"centered"`, and `"noncentered"`. Alternatively, it can be a single element character vector of the form `"asisX"`, where `X` is an integer, which is equivalent to `rep(c("centered", "noncentered"), X)`. Defaults to `"asis5"`.

`gammaprior`: Single logical value indicating whether a Gamma prior for `sigma^2` should be used. If set to `FALSE`, a moment-matched Inverse Gamma prior is employed. Defaults to `TRUE`.

`init.with.svsample`: Single integer indicating the length of a "pre-burnin" run using the computationally much more efficient `svsample`. This run helps in finding good initial values for the latent states, giving `svlsample` a considerable initial boost for convergence. Defaults to `1000L`.

`mhcontrol`: Either a single numeric value specifying the diagonal elements of a diagonal covariance matrix, or a list with two elements, both single numeric values (explained later), or a 4x4 covariance matrix. Argument `mhcontrol` controls the proposal density of a Metropolis-Hastings (MH) update step when jointly sampling `mu`, `phi`, `sigma`, and `rho`. It specifies the covariance matrix of a log-random-walk proposal. In case `mhcontrol` is a list of length two, its elements have to be `scale` and `rho.var`. In this case, the covariance matrix is calculated from the pre-burnin step with `svsample`, which gives an approximate posterior structure of the second moment for `mu`, `phi`, and `sigma`. This covariance matrix is then extended with `mhcontrol$rho.var`, specifying the variance for `rho`. The off-diagonal elements belonging to `rho` are set to 0. Finally, the whole covariance matrix is scaled by `mhcontrol$scale`. For this case to work, `init.with.svsample` has to be positive. Defaults to `list(scale=0.35, rho.var=0.02)`.

`correct.latent.draws`: Single logical value indicating whether to correct the draws obtained from the auxiliary model of Omori, et al. (2007). Defaults to `TRUE`.

`...` Any extra arguments will be forwarded to `updatesummary`, controlling the type of statistics calculated for the posterior draws.

Value

The value returned is a list object of class `svldraws` holding

`para`	`mcmc` object containing the *parameter* draws from the posterior distribution.
`latent`	`mcmc` object containing the *latent instantaneous log-volatility* draws from the posterior distribution.
`beta`	`mcmc` object containing the *regression coefficient* draws from the posterior distribution *(optional)*.
`y`	the argument `y`.
`runtime`	`proc_time` object containing the run time of the sampler.
`priors`	`list` containing the parameter values of the prior distribution, i.e. the arguments `priormu`, `priorphi`, `priorsigma`, and `priorrho`, and potentially `priorbeta`.
`thinning`	`list` containing the thinning parameters, i.e. the arguments `thinpara`, `thinlatent` and `keeptime`.
`summary`	`list` containing a collection of summary statistics of the posterior draws for `para`, and `latent`.
`meanmodel`	`character` containing information about how `designmatrix` was used.

To display the output, use `print`, `summary` and `plot`. The `print` method simply prints the posterior draws (which is very likely a lot of output); the `summary` method displays the summary statistics currently stored in the object; the `plot` method `plot.svdraws` gives a graphical overview of the posterior distribution by calling `volplot`, `traceplot` and `densplot` and displaying the results on a single page.

Note

If `y` contains zeros, you might want to consider de-meaning your returns or use `designmatrix = "ar0"`. We use the Metropolis-Hastings algorithm for sampling the latent vector `h`, where the proposal is a draw from an auxiliary mixture approximation model [Omori, et al. (2007)]. We draw the parameters `mu`, `phi`, `sigma`, and `rho` jointly by employing a Metropolis random walk step. By default, we boost the random walk through the repeated application of the ancillarity-sufficiency interweaving strategy (ASIS) [Yu, Meng (2011)]. A message in the beginning of sampling indicates the interweaving strategy used, which can be modified through parameter `expert`.

Author(s)

Darjus Hosszejni <darjus.hosszejni@wu.ac.at>

References

Yu, Y. and Meng, X.-L. (2011). To Center or not to Center: That is not the Question—An Ancillarity-Sufficiency Interweaving Strategy (ASIS) for Boosting MCMC Efficiency. *Journal of Computational and Graphical Statistics*, **20**(3), 531–570, http://dx.doi.org/10.1198/jcgs.2011.203main

Omori, Y. and Chib, S. and Shephard, N. and Nakajima, J. (2007). Stochastic Volatility with Leverage: Fast and Efficient Likelihood Inference. *Journal of Econometrics*, **140**(2), 425–449, http://dx.doi.org/10.1016/j.jeconom.2006.07.008

See Also

svsim, svsample, updatesummary, predict.svdraws, plot.svdraws.

Examples

```
## Not run:
# Example 1
## Simulate a short SVL process
sim <- svsim(200, mu = -10, phi = 0.95, sigma = 0.2, rho = -0.4)

## Obtain 5000 draws from the sampler (that's not a lot)
draws <- svlsample(sim$y)

## Check out the results
summary(draws)
plot(draws, simobj = sim)


# Example 2
## AR(1) structure for the mean
data(exrates)
len <- 1200
ahead <- 100
y <- head(exrates$USD, len)

## Fit AR(1)-SVL model to EUR-USD exchange rates
res <- svlsample(y, designmatrix = "ar1")

## Use predict.svdraws to obtain predictive distributions
preddraws <- predict(res, steps = ahead)

## Calculate predictive quantiles
predquants <- apply(preddraws$y, 2, quantile, c(.1, .5, .9))

## Visualize
expost <- tail(head(exrates$USD, len+ahead), ahead)
ts.plot(y, xlim = c(length(y)-4*ahead, length(y)+ahead),
       ylim = range(c(predquants, expost, tail(y, 4*ahead))))
for (i in 1:3) {
  lines((length(y)+1):(length(y)+ahead), predquants[i,],
        col = 3, lty = c(2, 1, 2)[i])
}
lines((length(y)+1):(length(y)+ahead), expost,
      col = 2)


# Example 3
## Predicting USD based on JPY and GBP in the mean
data(exrates)
len <- 1200
ahead <- 30
## Calculate log-returns
```

```
logreturns <- apply(exrates[, c("USD", "JPY", "GBP")], 2,
                    function (x) diff(log(x)))
logretUSD <- logreturns[2:(len+1), "USD"]
regressors <- cbind(1, as.matrix(logreturns[1:len, ]))  # lagged by 1 day

## Fit SV model to EUR-USD exchange rates
res <- svlsample(logretUSD, designmatrix = regressors)

## Use predict.svdraws to obtain predictive distributions
predregressors <- cbind(1, as.matrix(logreturns[(len+1):(len+ahead), ]))
preddraws <- predict(res, steps = ahead,
                     newdata = predregressors)
predprice <- exrates[len+2, "USD"] * exp(t(apply(preddraws$y, 1, cumsum)))

## Calculate predictive quantiles
predquants <- apply(predprice, 2, quantile, c(.1, .5, .9))

## Visualize
priceUSD <- exrates[3:(len+2), "USD"]
expost <- exrates[(len+3):(len+ahead+2), "USD"]
ts.plot(priceUSD, xlim = c(len-4*ahead, len+ahead+1),
       ylim = range(c(expost, predquants, tail(priceUSD, 4*ahead))))
for (i in 1:3) {
  lines(len:(len+ahead), c(tail(priceUSD, 1), predquants[i,]),
        col = 3, lty = c(2, 1, 2)[i])
}
lines(len:(len+ahead), c(tail(priceUSD, 1), expost),
      col = 2)

## End(Not run)
```

`svlsample2` *Minimal overhead version of* `svlsample`.

Description

`svlsample2` is a minimal overhead version of `svlsample` with slightly different default arguments and a simplified return value structure. It is intended to be used mainly for one-step updates where speed is an issue, e.g., as a plug-in into other MCMC samplers. Note that absolutely no input checking is performed, thus this function is to be used with proper care!

Usage

```
svlsample2(y, draws = 1, burnin = 0, priormu = c(0, 100),
  priorphi = c(5, 1.5), priorsigma = 1, priorrho = c(4, 4),
  thinpara = 1, thinlatent = 1, thintime = NULL, keeptime = "all",
  quiet = TRUE, startpara, startlatent)
```

Arguments

y	numeric vector containing the data (usually log-returns), which must not contain zeros. Alternatively, y can be an svsim object. In this case, the returns will be extracted and a warning is thrown.
draws	single number greater or equal to 1, indicating the number of draws after burn-in (see below). Will be automatically coerced to integer. The default value is 1.
burnin	single number greater or equal to 0, indicating the number of draws discarded as burn-in. Will be automatically coerced to integer. The default value is 0.
priormu	numeric vector of length 2, indicating mean and standard deviation for the Gaussian prior distribution of the parameter mu, the level of the log-volatility. The default value is c(0, 100), which constitutes a practically uninformative prior for common exchange rate datasets, stock returns and the like.
priorphi	numeric vector of length 2, indicating the shape parameters for the Beta prior distribution of the transformed parameter (phi + 1) / 2, where phi denotes the persistence of the log-volatility. The default value is c(5, 1.5), which constitutes a prior that puts some belief in a persistent log-volatility but also encompasses the region where phi is around 0.
priorsigma	single positive real number, which stands for the scaling of the transformed parameter sigma^2, where sigma denotes the volatility of log-volatility. More precisely, sigma^2 ~ priorsigma * chisq(df = 1). The default value is 1, which constitutes a reasonably vague prior for many common exchange rate datasets, stock returns and the like.
priorrho	numeric vector of length 2, indicating the shape parameters for the Beta prior distribution of the transformed parameter (rho + 1) / 2, where rho denotes the conditional correlation between observation and the increment of the log-volatility. The default value is c(4, 4), which constitutes a slightly informative prior around 0 (the no leverage case) to boost convergence.
thinpara	single number greater or equal to 1, coercible to integer. Every thinparath parameter draw is kept and returned. The default value is 1, corresponding to no thinning of the parameter draws i.e. every draw is stored.
thinlatent	single number greater or equal to 1, coercible to integer. Every thinlatentth latent variable draw is kept and returned. The default value is 1, corresponding to no thinning of the latent variable draws, i.e. every draw is kept.
thintime	*Deprecated.* Use 'keeptime' instead.
keeptime	Either 'all' (the default) or 'last'. Indicates which latent
quiet	logical value indicating whether the progress bar and other informative output during sampling should be omitted. The default value is TRUE.
startpara	*compulsory* named list, containing the starting values for the parameter draws. It must contain four elements named mu, phi, sigma, and rho, where mu is an arbitrary numerical value, phi is a real number between -1 and 1, sigma is a positive real number, and rho is a real number between -1 and 1.
startlatent	*compulsory* vector of length length(y), containing the starting values for the latent log-volatility draws.

Details

As opposed to the ordinary svlsample, the default values differ for draws, burnin, and quiet. Note that currently neither expert nor ...{} arguments are provided.

Value

The value returned is a list object holding

para	matrix of dimension 4 x draws containing the *parameter* draws from the posterior distribution.
latent	matrix of dimension length(y) x draws containing the *latent instantaneous log-volatility* draws from the posterior distribution.
meanmodel	always equals "none"

Author(s)

Darjus Hosszejni <darjus.hosszejni@wu.ac.at>

See Also

svlsample

Examples

```
data(exrates)
aud.price <- subset(exrates,
  as.Date("2010-01-01") <= date & date < as.Date("2011-01-01"),
  "AUD")[,1]
draws <- svlsample2(logret(aud.price),
                    draws = 10, burnin = 0,
                    startpara = list(phi=0.95, mu=-10, sigma=0.2, rho=-0.1),
                    startlatent = rep_len(-10, length(aud.price)-1))
```

---

svsample *Markov Chain Monte Carlo (MCMC) Sampling for the Stochastic Volatility (SV) Model*

Description

svsample simulates from the joint posterior distribution of the SV parameters mu, phi, sigma (and potentially nu), along with the latent log-volatilities h_0, . . . , h_n and returns the MCMC draws. If a design matrix is provided, simple Bayesian regression can also be conducted.

Usage

```
svsample(y, draws = 10000, burnin = 1000, designmatrix = NA,
  priormu = c(0, 100), priorphi = c(5, 1.5), priorsigma = 1,
  priornu = NA, priorbeta = c(0, 10000), priorlatent0 = "stationary",
  thinpara = 1, thinlatent = 1, keeptime = "all", thintime = NULL,
  keeptau = FALSE, quiet = FALSE, startpara, startlatent, expert, ...)

svtsample(y, draws = 10000, burnin = 1000, designmatrix = NA,
  priormu = c(0, 100), priorphi = c(5, 1.5), priorsigma = 1,
  priornu = c(2, 50), priorbeta = c(0, 10000),
  priorlatent0 = "stationary", thinpara = 1, thinlatent = 1,
  keeptime = "all", thintime = NULL, keeptau = FALSE,
  quiet = FALSE, startpara, startlatent, expert, ...)
```

Arguments

y	numeric vector containing the data (usually log-returns), which must not contain zeros. Alternatively, y can be an svsim object. In this case, the returns will be extracted and a warning is thrown.
draws	single number greater or equal to 1, indicating the number of draws after burn-in (see below). Will be automatically coerced to integer. The default value is 10000.
burnin	single number greater or equal to 0, indicating the number of draws discarded as burn-in. Will be automatically coerced to integer. The default value is 1000.
designmatrix	regression design matrix for modeling the mean. Must have length(y) rows. Alternatively, designmatrix may be a string of the form "arX", where X is a nonnegative integer. To fit a constant mean model, use designmatrix = "ar0" (which is equivalent to designmatrix = matrix(1, nrow = length(y))). To fit an AR(1) model, use designmatrix = "ar1", and so on. If some elements of designmatrix are NA, the mean is fixed to zero (pre-1.2.0 behavior of **stochvol**).
priormu	numeric vector of length 2, indicating mean and standard deviation for the Gaussian prior distribution of the parameter mu, the level of the log-volatility. The default value is c(0, 100), which constitutes a practically uninformative prior for common exchange rate datasets, stock returns and the like.
priorphi	numeric vector of length 2, indicating the shape parameters for the Beta prior distribution of the transformed parameter (phi + 1) / 2, where phi denotes the persistence of the log-volatility. The default value is c(5, 1.5), which constitutes a prior that puts some belief in a persistent log-volatility but also encompasses the region where phi is around 0.
priorsigma	single positive real number, which stands for the scaling of the transformed parameter sigma^2, where sigma denotes the volatility of log-volatility. More precisely, sigma^2 ~ priorsigma * chisq(df = 1). The default value is 1, which constitutes a reasonably vague prior for many common exchange rate datasets, stock returns and the like.
priornu	numeric vector of length 2 (or NA), indicating the lower and upper bounds for the uniform prior distribution of the parameter nu, the degrees-of-freedom parameter of the conditional innovations t-distribution. The default value is NA,

fixing the degrees-of-freedom to infinity. This corresponds to conditional standard normal innovations, the pre-1.1.0 behavior of **stochvol**.

`priorbeta` numeric vector of length 2, indicating the mean and standard deviation of the Gaussian prior for the regression parameters. The default value is `c(0, 10000)`, which constitutes a very vague prior for many common datasets. Not used if `designmatrix` is `NA`.

`priorlatent0` either a single non-negative number or the string `'stationary'` (the default, also the behavior before version 1.3.0). When `priorlatent0` is equal to `'stationary'`, the stationary distribution of the latent AR(1)-process is used as the prior for the initial log-volatility `h_0`. When `priorlatent0` is equal to a number B, we have $h_0 \sim N(\mu, B\sigma^2)$ a priori.

`thinpara` single number greater or equal to 1, coercible to integer. Every `thinparath` parameter draw is kept and returned. The default value is 1, corresponding to no thinning of the parameter draws i.e. every draw is stored.

`thinlatent` single number greater or equal to 1, coercible to integer. Every `thinlatententh` latent variable draw is kept and returned. The default value is 1, corresponding to no thinning of the latent variable draws, i.e. every draw is kept.

`keeptime` Either 'all' (the default) or 'last'. Indicates which latent

`thintime` *Deprecated.* Use 'keeptime' instead.

`keeptau` logical value indicating whether the 'variance inflation factors' should be stored (used for the sampler with conditional t innovations only). This may be useful to check at what point(s) in time the normal disturbance had to be 'upscaled' by a mixture factor and when the series behaved 'normally'.

`quiet` logical value indicating whether the progress bar and other informative output during sampling should be omitted. The default value is `FALSE`, implying verbose output.

`startpara` *optional* named list, containing the starting values for the parameter draws. If supplied, `startpara` must contain three elements named `mu`, `phi`, and `sigma`, where `mu` is an arbitrary numerical value, `phi` is a real number between `-1` and `1`, and `sigma` is a positive real number. Moreover, if `priornu` is not `NA`, `startpara` must also contain an element named `nu` (the degrees of freedom parameter for the t-innovations). The default value is equal to the prior mean.

`startlatent` *optional* vector of length `length(y)`, containing the starting values for the latent log-volatility draws. The default value is `rep(-10, length(y))`.

`expert` *optional* named list of expert parameters. For most applications, the default values probably work best. Interested users are referred to the literature provided in the References section. If `expert` is provided, it may contain the following named elements:

`parameterization`: Character string equal to `"centered"`, `"noncentered"`, `"GIS_C"`, or `"GIS_NC"`. Defaults to `"GIS_C"`.

`mhcontrol`: Single numeric value controlling the proposal density of a Metropolis-Hastings (MH) update step when sampling `sigma`. If `mhcontrol` is smaller than 0, an independence proposal will be used, while values greater than zero control the stepsize of a log-random-walk proposal. Defaults to `-1`.

`gammaprior`: Single logical value indicating whether a Gamma prior for `sigma^2` should be used. If set to `FALSE`, an Inverse Gamma prior is employed. Defaults to `TRUE`.

`truncnormal`: Single logical value indicating whether a truncated Gaussian distribution should be used as proposal for draws of `phi`. If set to `FALSE`, a regular Gaussian prior is employed and the draw is immediately discarded when values outside the unit ball happen to be drawn. Defaults to `FALSE`.

`mhsteps`: Either `1`, `2`, or `3`. Indicates the number of blocks used for drawing from the posterior of the parameters. Defaults to `2`.

`proposalvar4sigmaphi`: Single positive number indicating the conditional prior variance of `sigma*phi` in the ridge *proposal* density for sampling `(mu, phi)`. Defaults to `10^8`.

`proposalvar4sigmatheta`: Single positive number indicating the conditional prior variance of `sigma*theta` in the ridge *proposal* density for sampling `(mu, phi)`. Defaults to `10^12`.

`...` Any extra arguments will be forwarded to `updatesummary`, controlling the type of statistics calculated for the posterior draws.

Details

For details concerning the algorithm please see the paper by Kastner and Frühwirth-Schnatter (2014).

Value

The value returned is a list object of class `svdraws` holding

`para`	`mcmc` object containing the *parameter* draws from the posterior distribution.
`latent`	`mcmc` object containing the *latent instantaneous log-volatility* draws from the posterior distribution.
`latent0`	`mcmc` object containing the *latent initial log-volatility* draws from the posterior distribution.
`tau`	`mcmc` object containing the *latent variance inflation factors* for the sampler with conditional t-innovations *(optional)*.
`beta`	`mcmc` object containing the *regression coefficient* draws from the posterior distribution *(optional)*.
`y`	the argument `y`.
`runtime`	`proc_time` object containing the run time of the sampler.
`priors`	`list` containing the parameter values of the prior distribution, i.e. the arguments `priormu`, `priorphi`, `priorsigma`, and potentially `priornu` and `priorbeta`.
`thinning`	`list` containing the thinning parameters, i.e. the arguments `thinpara`, `thinlatent` and `keeptime`.
`summary`	`list` containing a collection of summary statistics of the posterior draws for `para`, `latent`, and `latent0`.
`meanmodel`	`character` containing information about how `designmatrix` was employed.

To display the output, use `print`, `summary` and `plot`. The `print` method simply prints the posterior draws (which is very likely a lot of output); the `summary` method displays the summary statistics currently stored in the object; the `plot` method `plot.svdraws` gives a graphical overview of the posterior distribution by calling `volplot`, `traceplot` and `densplot` and displaying the results on a single page.

Note

If y contains zeros, you might want to consider de-meaning your returns or use `designmatrix = "ar0"`.

Author(s)

Gregor Kastner <gregor.kastner@wu.ac.at>

References

Kastner, G. and Frühwirth-Schnatter, S. (2014). Ancillarity-sufficiency interweaving strategy (ASIS) for boosting MCMC estimation of stochastic volatility models. *Computational Statistics & Data Analysis*, **76**, 408–423, http://dx.doi.org/10.1016/j.csda.2013.01.002.

See Also

`svsim`, `svlsample`, `updatesummary`, `predict.svdraws`, `plot.svdraws`.

Examples

```
# Example 1
## Simulate a short and highly persistent SV process
sim <- svsim(100, mu = -10, phi = 0.99, sigma = 0.2)

## Obtain 5000 draws from the sampler (that's not a lot)
draws <- svsample(sim$y, draws = 5000, burnin = 100,
  priormu = c(-10, 1), priorphi = c(20, 1.5), priorsigma = 0.2)

## Check out the results
summary(draws)
plot(draws)


## Not run:
# Example 2
## AR(1) structure for the mean
data(exrates)
len <- 3000
ahead <- 100
y <- head(exrates$USD, len)

## Fit AR(1)-SVL model to EUR-USD exchange rates
res <- svsample(y, designmatrix = "ar1")

## Use predict.svdraws to obtain predictive distributions
preddraws <- predict(res, steps = ahead)
```

```
## Calculate predictive quantiles
predquants <- apply(preddraws$y, 2, quantile, c(.1, .5, .9))

## Visualize
expost <- tail(head(exrates$USD, len+ahead), ahead)
ts.plot(y, xlim = c(length(y)-4*ahead, length(y)+ahead),
       ylim = range(c(predquants, expost, tail(y, 4*ahead))))
for (i in 1:3) {
  lines((length(y)+1):(length(y)+ahead), predquants[i,],
        col = 3, lty = c(2, 1, 2)[i])
}
lines((length(y)+1):(length(y)+ahead), expost,
      col = 2)


# Example 3
## Predicting USD based on JPY and GBP in the mean
data(exrates)
len <- 3000
ahead <- 30
## Calculate log-returns
logreturns <- apply(exrates[, c("USD", "JPY", "GBP")], 2,
                    function (x) diff(log(x)))
logretUSD <- logreturns[2:(len+1), "USD"]
regressors <- cbind(1, as.matrix(logreturns[1:len, ]))  # lagged by 1 day

## Fit SV model to EUR-USD exchange rates
res <- svsample(logretUSD, designmatrix = regressors)

## Use predict.svdraws to obtain predictive distributions
predregressors <- cbind(1, as.matrix(logreturns[(len+1):(len+ahead), ]))
preddraws <- predict(res, steps = ahead,
                     newdata = predregressors)
predprice <- exrates[len+2, "USD"] * exp(t(apply(preddraws$y, 1, cumsum)))

## Calculate predictive quantiles
predquants <- apply(predprice, 2, quantile, c(.1, .5, .9))

## Visualize
priceUSD <- exrates[3:(len+2), "USD"]
expost <- exrates[(len+3):(len+ahead+2), "USD"]
ts.plot(priceUSD, xlim = c(len-4*ahead, len+ahead+1),
       ylim = range(c(expost, predquants, tail(priceUSD, 4*ahead))))
for (i in 1:3) {
  lines(len:(len+ahead), c(tail(priceUSD, 1), predquants[i,]),
        col = 3, lty = c(2, 1, 2)[i])
}
lines(len:(len+ahead), c(tail(priceUSD, 1), expost),
      col = 2)

## End(Not run)
```

svsample2 *Minimal overhead version of* svsample.

Description

svsample2 is a minimal overhead version of svsample with slightly different default arguments and a simplified return value structure. It is intended to be used mainly for one-step updates where speed is an issue, e.g., as a plug-in into other MCMC samplers. Note that absolutely no input checking is performed, thus this function is to be used with proper care!

Usage

```
svsample2(y, draws = 1, burnin = 0, priormu = c(0, 100),
  priorphi = c(5, 1.5), priorsigma = 1, priornu = NA,
  priorlatent0 = "stationary", thinpara = 1, thinlatent = 1,
  thintime = NULL, keeptime = "all", keeptau = FALSE, quiet = TRUE,
  startpara, startlatent)
```

Arguments

y	numeric vector containing the data (usually log-returns), which must not contain zeroes.
draws	single number greater or equal to 1, indicating the number of draws after burn-in (see below). Will be automatically coerced to integer. The defaults value is 1.
burnin	single number greater or equal to 0, indicating the number of draws discarded as burn-in. Will be automatically coerced to integer. The default value is 0.
priormu	numeric vector of length 2, indicating mean and standard deviation for the Gaussian prior distribution of the parameter mu, the level of the log-volatility. The default value is c(0, 100), which constitutes a practically uninformative prior for common exchange rate datasets, stock returns and the like.
priorphi	numeric vector of length 2, indicating the shape parameters for the Beta prior distribution of the transformed parameter (phi + 1) / 2, where phi denotes the persistence of the log-volatility. The default value is c(5, 1.5), which constitutes a prior that puts some belief in a persistent log-volatility but also encompasses the region where phi is around 0.
priorsigma	single positive real number, which stands for the scaling of the transformed parameter sigma^2, where sigma denotes the volatility of log-volatility. More precisely, sigma^2 ~ priorsigma * chisq(df = 1). The default value is 1, which constitutes a reasonably vague prior for many common exchange rate datasets, stock returns and the like.
priornu	numeric vector of length 2 (or NA), indicating the lower and upper bounds for the uniform prior distribution of the parameter nu, the degrees-of-freedom parameter of the conditional innovations t-distribution. The default value is NA, fixing the degrees-of-freedom to infinity. This corresponds to conditional standard normal innovations, the pre-1.1.0 behavior of **stochvol**.

`priorlatent0` either a single non-negative number or the string `'stationary'` (the default, also the behavior before version 1.3.0). When `priorlatent0` is equal to `'stationary'`, the stationary distribution of the latent AR(1)-process is used as the prior for the initial log-volatility `h_0`. When `priorlatent0` is equal to a number B, we have $h_0 \sim N(\mu, B\sigma^2)$ a priori.

`thinpara` single number greater or equal to 1, coercible to integer. Every `thinparath` parameter draw is kept and returned. The default value is 1, corresponding to no thinning of the parameter draws – every draw is stored.

`thinlatent` single number greater or equal to 1, coercible to integer. Every `thinlatentth` latent variable draw is kept and returned. The default value is 1, corresponding to no thinning of the latent variable draws, i.e. every draw is kept.

`thintime` *Deprecated.* Use 'keeptime' instead.

`keeptime` Either 'all' (the default) or 'last'. Indicates which latent

`keeptau` logical value indicating whether the 'variance inflation factors' should be stored (used for the sampler with conditional t innovations only). This may be useful to check at what point(s) in time the normal disturbance had to be 'upscaled' by a mixture factor and when the series behaved 'normally'.

`quiet` logical value indicating whether the progress bar and other informative output during sampling should be omitted. The default value is `TRUE`, implying non-verbose output.

`startpara` *compulsory* named list, containing the starting values for the parameter draws. `startpara` must contain three elements named `mu`, `phi`, and `sigma`, where `mu` is an arbitrary numerical value, `phi` is a real number between `-1` and `1`, and `sigma` is a positive real number. Moreover, if `priornu` is not `NA`, `startpara` must also contain an element named `nu` (the degrees of freedom parameter for the t-innovations).

`startlatent` *compulsory* vector of length `length(x$y)`, containing the starting values for the latent log-volatility draws.

Details

As opposed to the ordinary `svsample`, the default values differ for `draws`, `burnin`, and `quiet`. Note that currently neither `expert` nor `...{}` arguments are provided.

Value

A list with three components:

`para` `3` times `draws` matrix containing the parameter draws. If `priornu` is not `NA`, this is a `4` times `draws` matrix.

`latent` `length(y)` times `draws` matrix containing draws of the latent variables `h_1, ...{}, h_n`.

`latent0` Vector of length `draws` containing the draw(s) of the initial latent variable `h_0`.

Warning

Expert use only! For most applications, the use of `svsample` is recommended.

Note

Please refer to the package vignette for an example.

Author(s)

Gregor Kastner <gregor.kastner@wu.ac.at>

See Also

svsample

Examples

```
data(exrates)
aud.price <- subset(exrates,
  as.Date("2010-01-01") <= date & date < as.Date("2011-01-01"),
  "AUD")[,1]
draws <- svsample2(logret(aud.price),
                   draws = 10, burnin = 0,
                   startpara = list(phi = 0.95, mu = -10, sigma = 0.2, rho = -0.1),
                   startlatent = rep_len(-10, length(aud.price) - 1))
```

---

svsim *Simulating a Stochastic Volatility Process*

Description

svsim is used to produce realizations of a stochastic volatility (SV) process.

Usage

```
svsim(len, mu = -10, phi = 0.98, sigma = 0.2, nu = Inf, rho = 0)
```

Arguments

len	length of the simulated time series.
mu	level of the latent log-volatility AR(1) process. The defaults value is -10.
phi	persistence of the latent log-volatility AR(1) process. The default value is 0.98.
sigma	volatility of the latent log-volatility AR(1) process. The default value is 0.2.
nu	degrees-of-freedom for the conditional innovations distribution. The default value is Inf, corresponding to standard normal conditional innovations.
rho	correlation between the observation and the increment of the log-volatility. The default value is 0, corresponding to the basic SV model with symmetric "log-returns".

Details

This function draws an initial log-volatility h_0 from the stationary distribution of the AR(1) process defined by phi, sigma, and mu. Then the function jointly simulates the log-volatility series h_1,...,h_n with the given AR(1) structure, and the "log-return" series y_1,...,y_n with mean 0 and standard deviation exp(h/2). Additionally, for each index i, y_i can be set to have a conditionally heavy-tailed residual (through nu) and/or to be correlated with (h_{i+1}-h_i) (through rho, the so-called leverage effect, resulting in asymmetric "log-returns").

Value

The output is a list object of class svsim containing

y	a vector of length len containing the simulated data, usually interpreted as "log-returns".
vol	a vector of length len containing the simulated instantaneous volatilities exp(h_t/2).
vol0	The initial volatility exp(h_0/2), drawn from the stationary distribution of the latent AR(1) process.
para	a named list with five elements mu, phi, sigma, nu, and rho, containing the corresponding arguments.

Note

The function generates the "log-returns" by y <- exp(-h/2)*rt(h, df=nu). That means that in the case of nu < Inf the (conditional) volatility is sqrt(nu/(nu-2))*exp(h/2), and that corrected value is shown in the print, summary and plot methods.

To display the output use print, summary and plot. The print method simply prints the content of the object in a moderately formatted manner. The summary method provides some summary statistics (in %), and the plot method plots the the simulated 'log-returns' y along with the corresponding volatilities vol.

Author(s)

Gregor Kastner <gregor.kastner@wu.ac.at>

See Also

svsample

Examples

```
## Simulate a highly persistent SV process of length 500
sim <- svsim(500, phi = 0.99, sigma = 0.1)

print(sim)
summary(sim)
plot(sim)

## Simulate an SV process with leverage
```

```
sim <- svsim(200, phi = 0.94, sigma = 0.15, rho = -0.6)

print(sim)
summary(sim)
plot(sim)

## Simulate an SV process with conditionally heavy-tails
sim <- svsim(250, phi = 0.91, sigma = 0.05, nu = 5)

print(sim)
summary(sim)
plot(sim)
```

`updatesummary` *Updating the Summary of MCMC Draws*

Description

Creates or updates a summary of an `svdraws` object.

Usage

```
updatesummary(x, quantiles = c(0.05, 0.5, 0.95), esspara = TRUE,
  esslatent = FALSE)
```

Arguments

`x`	`svdraws` object.
`quantiles`	numeric vector of posterior quantiles to be computed. The default is `c(0.05, 0.5, 0.95)`.
`esspara`	logical value which indicates whether the effective sample size (ESS) should be calculated for the *parameter draws*. This is achieved by calling `effectiveSize` from the `coda` package. The default is `TRUE`.
`esslatent`	logical value which indicates whether the effective sample size (ESS) should be calculated for the *latent log-volatility* draws. This is achieved by calling `effectiveSize` from the `coda` package. The default is `FALSE`, because this can be quite time-consuming when many latent variables are present.

Details

`updatesummary` will always calculate the posterior mean and the posterior standard deviation of the raw draws and some common transformations thereof. Moroever, the posterior quantiles, specified by the argument `quantiles`, are computed. If `esspara` and/or `esslatent` are `TRUE`, the corresponding effective sample size (ESS) will also be included.

Value

The value returned is an updated list object of class `svdraws` holding

`para`	`mcmc` object containing the *parameter* draws from the posterior distribution.
`latent`	`mcmc` object containing the *latent instantaneous log-volatility* draws from the posterior distribution.
`latent0`	`mcmc` object containing the *latent initial log-volatility* draws from the posterior distribution.
`y`	argument `y`.
`runtime`	`"proc_time"` object containing the run time of the sampler.
`priors`	`list` containing the parameter values of the prior distribution, i.e. the arguments `priormu`, `priorphi`, `priorsigma` (and potentially `nu`).
`thinning`	`list` containing the thinning parameters, i.e. the arguments `thinpara`, `thinlatent` and `keeptime`.
`summary`	`list` containing a collection of summary statistics of the posterior draws for `para`, `latent`, and `latent0`.

To display the output, use `print`, `summary` and `plot`. The `print` method simply prints the posterior draws (which is very likely a lot of output); the `summary` method displays the summary statistics currently stored in the object; the `plot` method gives a graphical overview of the posterior distribution by calling `volplot`, `traceplot` and `densplot` and displaying the results on a single page.

Note

`updatesummary` does not actually overwrite the object's current summary, but in fact creates a new object with an updated summary. Thus, don't forget to overwrite the old object if this is want you intend to do. See the examples below for more details.

Author(s)

Gregor Kastner <gregor.kastner@wu.ac.at>

See Also

`svsample`, `svlsample`

Examples

```
## Here is a baby-example to illustrate the idea.
## Simulate an SV time series of length 51 with default parameters:
sim <- svsim(51)

## Draw from the posterior:
res <- svsample(sim$y, draws = 7000, priorphi = c(10, 1.5))

## Check out the results:
summary(res)
```

```
plot(res)

## Look at other quantiles and calculate ESS of latents:
newquants <- c(0.01, 0.05, 0.25, 0.5, 0.75, 0.95, 0.99)
res <- updatesummary(res, quantiles = newquants, esslatent = TRUE)

## See the difference?
summary(res)
plot(res)
```

---

volplot *Plotting Quantiles of the Latent Volatilities*

Description

Displays quantiles of the posterior distribution of the volatilities over time as well as predictive distributions of future volatilities.

Usage

```
volplot(x, forecast = 0, dates = NULL, show0 = FALSE, col = NULL,
  forecastlty = NULL, tcl = -0.4, mar = c(1.9, 1.9, 1.9, 0.5),
  mgp = c(2, 0.6, 0), simobj = NULL, newdata = NULL, ...)
```

Arguments

x	svdraws or svldraws object.
forecast	nonnegative integer or object of class svpredict, as returned by predict.svdraws. If an integer greater than 0 is provided, predict.svdraws is invoked to obtain the forecast-step-ahead prediction. The default value is 0.
dates	vector of length ncol(x$latent), providing optional dates for labeling the x-axis. The default value is NULL; in this case, the axis will be labeled with numbers.
show0	logical value, indicating whether the initial volatility exp(h_0/2) should be displayed. The default value is FALSE. Only available for inputs x of class svdraws.
col	vector of color values (see par) used for plotting the quantiles. The default value NULL results in gray lines for all quantiles expect the median, which is displayed in black.
forecastlty	vector of line type values (see par) used for plotting quantiles of predictive distributions. The default value NULL results in dashed lines.
tcl	The length of tick marks as a fraction of the height of a line of text. See par for details. The default value is -0.4, which results in slightly shorter tick marks than usual.
mar	numerical vector of length 4, indicating the plot margins. See par for details. The default value is c(1.9,1.9, 1.9, 0.5), which is slightly smaller than the R-defaults.

mgp	numerical vector of length 3, indicating the axis and label positions. See par for details. The default value is c(2, 0.6, 0), which is slightly smaller than the R-defaults.
simobj	object of class svsim as returned by the SV simulation function svsim. If provided, "true" data generating values will be added to the plot(s).
newdata	corresponds to parameter newdata in predict.svdraws. *Only if* forecast *is a positive integer and* predict.svdraws *needs a* newdata *object.* Corresponds to input parameter designmatrix in svsample and svlsample. A matrix of regressors with number of rows equal to parameter forecast.
...	further arguments are passed on to the invoked ts.plot function.

Value

Called for its side effects. Returns argument x invisibly.

Note

In case you want different quantiles to be plotted, use updatesummary on the svdraws object first. An example of doing so is given below.

Author(s)

Gregor Kastner <gregor.kastner@wu.ac.at>

See Also

updatesummary, predict.svdraws

Other plotting: paradensplot, paratraceplot.svdraws, paratraceplot, plot.svdraws, plot.svpredict

Examples

```
## Simulate a short and highly persistent SV process
sim <- svsim(100, mu = -10, phi = 0.99, sigma = 0.2)

## Obtain 5000 draws from the sampler (that's not a lot)
draws <- svsample(sim$y, draws = 5000, burnin = 100,
  priormu = c(-10, 1), priorphi = c(20, 1.5),
  priorsigma = 0.2)

## Plot the latent volatilities and some forecasts
volplot(draws, forecast = 10)

## Re-plot with different quantiles
newquants <- c(0.01, 0.05, 0.25, 0.5, 0.75, 0.95, 0.99)
draws <- updatesummary(draws, quantiles = newquants)

volplot(draws, forecast = 10)
```

Index