Package ‘bmgarch’

December 14, 2021

Title Bayesian Multivariate GARCH Models

Version 1.1.0

Description Fit Bayesian multivariate GARCH models using 'Stan' for full Bayesian inference. Generate (weighted) forecasts for means, variances (volatility) and correlations. Currently DCC(P,Q), CCC(P,Q), pdBEKK(P,Q), and BEKK(P,Q) parameterizations are implemented, based either on a multivariate gaussian normal or student-t distribution. DCC and CCC models are based on Engle (2002) <doi:10.1198/073500102288618487> and Bollerslev (1990). The BEKK parameterization follows Engle and Kroner (1995) <doi:10.1017/S0266466600009063> while the pdBEKK as well as the estimation approach for this package is described in Rast et al. (2020) <doi:10.31234/osf.io/j57pk>. The fitted models contain 'rstan' objects and can be examined with 'rstan' functions.

License GPL (>= 3)

Depends methods, R (>= 4.0.0), Rcpp (>= 1.0.5)

Imports forecast, ggplot2, loo, MASS, Rdpack, rstan (>= 2.19.2), rstantools (>= 2.1.1)

LinkingTo BH (>= 1.72.0-0), Rcpp (>= 1.0.5), RcppParallel (>= 5.0.1), RcppEigen (>= 0.3.3.7.0), RcppParallel (>= 5.0.1), rstan (>= 2.19.2), StanHeaders (>= 2.21.0)

RdMacros Rdpack

Encoding UTF-8

LazyData true

NeedsCompilation yes

SystemRequirements GNU make

RoxygenNote 7.1.1

Suggests testthat (>= 2.3.2)

BugReports https://github.com/ph-rast/bmgarch/issues

Biarch true

Author Philippe Rast [aut, cre] (<https://orcid.org/0000-0003-3630-6629>), Stephen Martin [aut] (<https://orcid.org/0000-0001-8085-2390>)
The 'bmgarch' package.

Description

The *bmgarch* package fits Bayesian multivariate GARCH models specified via stan, a C++ package providing HMC methods for full Bayesian inference (cf. [http://mc-stan.org]). The currently implemented parameterizations are DCC(Q,P), CCC(Q,P), and BEKK(Q,P) with arbitrary lags defined in Q, and P. The package provides summaries and plots for the estimates as well as forecasted series with corresponding plots. The fitted objects are rstan class objects that can be inspected and manipulated accordingly.

Author(s)

Philippe Rast
References

Description
as.data.frame method for fitted.bmgarch objects.

Usage
## S3 method for class 'fitted.bmgarch'
as.data.frame(x, ...)

Arguments
x fitted.bmgarch object.
... Not used.

Value
Data frame.

Author(s)
Stephen R. Martin

Description
as.data.frame method for forecast.bmgarch objects.

Usage
## S3 method for class 'forecast.bmgarch'
as.data.frame(x, ..., backcast = TRUE)
Arguments

x  forecast.bmgarch object.
...
backcast Logical (Default: True). Whether to include "backcasted" values from fitted.bmgarch in data frame.

Value
Data frame.

Author(s)
Stephen R. Martin

Description

Draw samples from a specified multivariate GARCH model using 'Stan’, given multivariate time-series. Currently supports CCC, DCC, BEKK, and pdBEKK model parameterizations.

Usage

bmgarch(
  data,
  xC = NULL,
  parameterization = "CCC",
  P = 1,
  Q = 1,
  iterations = 2000,
  chains = 4,
  standardize_data = FALSE,
  distribution = "Student_t",
  meanstructure = "constant",
  sampling_algorithm = "MCMC",
  ...
)

Arguments

data  Time-series or matrix object. A time-series or matrix object containing observations at the same interval.
xC Numeric vector or matrix. Covariates(s) for the constant variance terms in C, or c, used in a log-linear model on the constant variance terms (Rast et al. 2020). If vector, then it acts as a covariate for all constant variance terms. If matrix, must have columns equal to number of time series, and each column acts as a covariate for the respective time series (e.g., column 1 predicts constant variance for time series 1).

parameterization Character (Default: "CCC"). The type of of parameterization. Must be one of "CCC", "DCC", "BEKK", or "pdBEKK".

P Integer. Dimension of GARCH component in MGARCH(P,Q).

Q Integer. Dimension of ARCH component in MGARCH(P,Q).

iterations Integer (Default: 2000). Number of iterations for each chain (including warmup).

chains Integer (Default: 4). The number of Markov chains.

standardize_data Logical (Default: FALSE). Whether data should be standardized to easy computations.

distribution Character (Default: "Student_t"). Distribution of innovation: "Student_t" or "Gaussian"

meanstructure Character (Default: "constant"). Defines model for means. Either 'constant' or 'ARMA'. Currently ARMA(1,1) only. OR 'VAR' (VAR1).

sampling_algorithm Character (Default" "MCMC"). Define sampling algorithm. Either 'MCMC' for Hamiltonian Monte Carlo or 'VB' for variational Bayes. 'VB' is inherited from stan and is currently in heavy development – do not trust estimates.

... Additional arguments can be 'chain_id', 'init_r', 'test_grad', 'append_samples', 'refresh', 'enable_random_init' etc. See the documentation in stan.

Details

Four types of parameterizations are implemented. The constant conditional correlation (CCC) and the dynamic conditional correlation (DCC; Engle2002,Engle2001a), as well as BEKK (Engle and Kroner 1995) and a BEKK model with positivity constraints on the diagonals of the ARCH and GARCH parameters "pdBEKK" (Rast et al. 2020).

The fitted models are 'rstan' objects and all posterior parameter estimates can be obtained and can be examined with either the 'rstan' toolbox, plotted and printed using generic functions or passed to 'bmgarch' functions to 'forecast' or compute 'model_weights' or compute fit statistics based on leave-future-out cross-validation.

Value

bmgarch object.

Author(s)

Philippe Rast, Stephen R. Martin
References


Examples

```r
## Not run:
data(panas)
# Fit BEKK(1,1) mgarch model with a ARMA(1,1) meanstructure, # and student-t residual distribution
fit <- bmgarch(panas, parameterization = "BEKK",
P = 1, Q = 1,
meanstructure = "arma",
distribution = "Student_t")

# Summarize the parameters
summary(fit)

# Forecast 5 ahead
fit.fc <- forecast(fit, ahead = 5)
print(fit.fc)

# Plot mean forecasts
plot(fit.fc, type = "mean")

# Plot variance forecasts
plot(fit.fc, type = "var")

# Plot correlation forecasts
plot(fit.fc, type = "cor")

# Plot modeled data ("backcasted values").
plot(fit, type = "mean")

# Save "backcasted" values
fit.bc <- fitted(fit)

# Save estimated and forecasted data as a data.frame
df.fc <- as.data.frame(fit.fc)

# Access rstan's model fit object
mf <- fit$model_fit

# Return diagnostics and a plot of the first 10 parameters
rstan::check_hmc_diagnostics(mf)
rstan::plot(mf)

## End(Not run)
```
Collect `bmgarch` objects into list.

**Usage**

```r
description

## S3 method for class 'bmgarch'
fitted(
  object,
  CrI = c(0.025, 0.975),
  digits = 2,
  weights = NULL,
  inc_samples = FALSE,
  ...)
```

**Arguments**

- `object` : `bmgarch` object.
- `CrI` : Numeric vector (Default: `c(0.025, 0.975)`). Lower and upper bound of predictive credible interval.
- `digits` : Integer (Default: 2, optional). Number of digits to round to when printing.
fitted.bmgarch

weights Takes weights from model_weight function. Defaults to 1 – this parameter is not typically set by user.

inc_samples Logical (Default: FALSE). Whether to return the MCMC samples for the fitted values.

Value

fitted.bmgarch object. List containing metadata and the backcast. Backcast is a list containing three elements:

- **mean** \([N,7,TS]\) array of mean backcasts, where N is the timeseries length, and TS is the number of time series. E.g., \(bc$backcast$mean[3,,"tsA"]\) is the mean backcast for the third observation in time series "tsA".

- **var** \([N,7,TS]\) array of variance backcasts, where N is the timeseries length, and TS is the number of time series. E.g., \(bc$backcast$var[3,,"tsA"]\) is the variance backcast for the third observation in time series "tsA".

- **cor** \([N,7,TS(TS-1)/2]\) array of correlation backcasts, where N is the timeseries length, and \(TS(TS-1)/2\) is the number of correlations. E.g., \(bc$backcast$cor[3,,"tsB_tsA"]\) is the backcast for the correlation between "tsB" and "tsA" on the third observation. Lower triangular correlations are saved.

- **samples** List. If inc_samples is TRUE, then a list of arrays of MCMC samples for means, vars, and cors. Each array is \([Iteration, Period, ..., ...]\).

Examples

```r
## Not run:
data(panas)
# Fit CCC(1,1) and constant meanstructure.
fit <- bmgarch(panas, parameterization = "CCC", meanstructure = "constant")

# Obtain fitted values
fit.bc <- fitted(fit)

# Print fitted values
print(fit.bc)

# Plot fitted values (plot.bmgarch calls fitted internally)
plot(fit, type = "var")

# Save fitted values as data frame
fit.bc.df <- as.data.frame(fit.bc)

## End(Not run)
```
Description

Estimates (weighted) forecasted means, variances, and correlations from a fitted bmgarch model.

Usage

```r
## S3 method for class 'bmgarch'
forecast(
  object,
  ahead = 1,
  xC = NULL,
  newdata = NULL,
  CrI = c(0.025, 0.975),
  seed = NA,
  digits = 2,
  weights = NULL,
  L = NA,
  method = "stacking",
  inc_samples = FALSE,
  ...
)
```

Arguments

- `object`: bmgarch object.
- `ahead`: Integer (Default: 1). Periods to be forecasted ahead.
- `xC`: Numeric vector or matrix. Covariates(s) for the constant variance terms in C, or c. Used in a log-linear model on the constant variance terms. If vector, then it acts as a covariate for all constant variance terms. If matrix, must have columns equal to number of time series, and each column acts as a covariate for the respective time series (e.g., column 1 predicts constant variance for time series 1).
- `newdata`: Future datapoints for LFO-CV computation.
- `CrI`: Numeric vector (Default: c(.025, .975)). Lower and upper bound of predictive credible interval.
- `seed`: Integer (Optional). Specify seed for sampling.
- `digits`: Integer (Default: 2, optional). Number of digits to round to when printing.
- `weights`: Takes weights from model_weight function. Defaults to 1 – this parameter is not typically set by user.
- `L`: Minimal length of time series before engaging in lfo_cv
- `method`: Ensemble methods, 'stacking' (default) or 'pseudobma'
inc_samples Logical (Default: FALSE). Whether to return the MCMC samples for the fitted values.

Value

forecast.bmgarch object. List containing forecast, backcast, and metadata. See fitted.bmgarch for information on backcast. forecast is a list of four components:

- **mean**: $[N, 7, TS]$ array of mean forecasts, where $N$ is the timeseries length, and $TS$ is the number of time series. E.g., fc$forecast$mean[3,,"tsA"] is the 3-ahead mean forecast for time series "tsA".
- **var**: $[N, 7, TS]$ array of variance forecasts, where $N$ is the timeseries length, and $TS$ is the number of time series. E.g., fc$forecast$var[3,,"tsA"] is the 3-ahead variance forecast for time series "tsA".
- **cor**: $[N, 7, TS(TS-1)/2]$ array of correlation forecasts, where $N$ is the timeseries length, and $TS(TS-1)/2$ is the number of correlations. E.g., fc$forecast$cor[3,,"tsB_tsA"] is the 3-ahead forecast for the correlation between "tsB" and "tsA". Lower triangular correlations are saved.
- **meta**: Meta-data specific to the forecast. I.e., TS_length (number ahead) and xC.

samples List. If inc_samples is TRUE, then a list of arrays of MCMC samples for means, vars, and cors. Each array is [Iteration, Period, ..., ...].

Examples

```r
## Not run:
data(panas)
# Fit DCC(2,2) with constant mean structure.
fit <- bmgarch(panas, parameterization = "DCC", P = 2, Q = 2, meanstructure = "constant")

# Forecast 8 ahead
fit.fc <- forecast(fit, ahead = 8)

# Print forecasts
fit.fc
print(fit.fc)

# Plot variance forecasts
plot(fit.fc, type = "var")

# Plot correlation forecasts
plot(fit.fc, type = "cor")

# Save backcasted and forecasted values as data frame.
fit.fc.df <- as.data.frame(fit.fc)

# Save only forecasted values as data frame.
fit.fc.df <- as.data.frame(fit.fc, backcast = FALSE)

# Add another model, compute model weights and perform a model weighted forecast
```
# Fit a DCC(1,1) model
fit1 <- bmgarch(panas, parameterization = "DCC", P = 1, Q = 1, meanstructure = "constant")

# Compute model stacking weights based on the last 19 time points (with L = 80)
blist <- bmgarch_list( fit1, fit )
mw <- model_weights(blist, L = 80)

# Weighted forecasts:
w.fc <- forecast(object = blist, ahead = 8, weights = mw)

## End(Not run)

---

##  loo.bmgarch

### Leave-Future-Out Cross Validation (LFO-CV)

**Description**

`lfocv` returns the LFO-CV ELDP by either computing the exact ELDP or by approximating it via forward or backward approximation strategies based on Pareto smoothed importance sampling described in (Bürkner et al. 2020).

**Usage**

```r
## S3 method for class 'bmgarch'
loo(x, ..., type = "lfo", L = NULL, M = 1, mode = "backward")
```

**Arguments**

- `x` Fitted bmgarch model. `lfocv` inherits all attributes from the bmgarch object
- `...` Not used
- `type` Takes `lfo` (default) or `loo`. LFO-CV is recommended for time-series but LOO-CV may be obtained to assess the structural part of the model.
- `L` Minimal length of times series before computing LFO
- `M` M step head predictions. Defines to what period the LFO-CV should be tuned to. Defaults to M=1.
- `mode` backward elpd_lfo approximation, or exact elpd-lfo; Takes 'backward', and 'exact'. 'exact' fits N-L models and may take a very long time to complete. forward works too but is not complete yet.

**Value**

Approximate LFO-CV value and log-likelihood values across (L+1):N timepoints

**References**

Examples

## Not run:
data(stocks)
# Fit a DCC model
fit <- bmgarch(data = stocks[1:100, c("toyota", "nissan")],
              parameterization = "DCC", standardize_data = TRUE,
              iterations = 500)

# Compute expected log-predictive density (elpd) using the backward mode
# L is the upper boundary of the time-series before we engage in LFO-CV
lfob <- loo(fit, mode = 'backward', L = 50 )
print(lfob)

## End(Not run)

---

model_weights

Model weights

Description

Compute model weights for a list of candidate models based on leave-future-out cross validation (lfocv) expected log-predictive density (elpd). elpd can be approximated via the 'backward' mode described in Bürkner et al. (2020) or via exact cross-validation. The obtained weights can be passed to the forecast function to obtain weighted forecasts. `bmgarch_objects` takes a `bmgarch_object` lists.

Usage

```r
model_weights(
  bmgarch_objects = NULL,
  L = NULL,
  M = 1,
  method = "stacking",
  mode = "backward"
)
```

Arguments

- `bmgarch_objects` list of bmgarch model objects in `bmgarch_object`
- `L` Minimal length of time series before engaging in lfocv
- `M` M step head predictions. Defines to what period the LFO-CV should be tuned to. Defaults to M=1.
- `method` Ensemble methods, 'stacking' (default) or 'pseudobma'
- `mode` Either 'backward' (default) or 'exact'
Details

$model_weights()$ is a wrapper around the leave-future-out 'lfo' type in 'loo.bmgarch()'. The weights can be either obtained from an approximate or exact leave-future-out cross-validation to compute expected log predictive density (ELPD).

We can either obtain stacking weights or pseudo-BMA+ weights as described in (Yao et al. 2018).

Value

Model weights

References


Examples

```r
## Not run:
data(stocks)
# Fit at least two models on a subset of the stocks data
# to compute model weights
fit <- bmgarch(data = stocks[1:100, c("toyota", "nissan")],
               parameterization = "DCC", standardize_data = TRUE,
               iterations = 500)

fit2 <- bmgarch(data = stocks[1:100, c("toyota", "nissan")],
               P = 2, Q = 2,
               parameterization = "DCC", standardize_data = TRUE,
               iterations = 500)
# create a bmgarch_list object
blist <- bmgarch_list(fit, fit2)

# Compute model weights with the default stacking method
# L is the upper boundary of the time-series before we engage in LFO-CV
mw <- model_weights( blist, L = 50, method = 'stacking', order = 'backwards' )

# Print model weights in the order of the bmgarch_list()
print(mw)
## End(Not run)
```
### panas

**Positive and Negative Affect Scores.**

#### Description

A dataset containing simulated values for Positive and Negative Affect scores across 200 measurement occasions for a single individual.

#### Usage

```r
panas
```

#### Format

Data frame with 200 rows and 2 variables:

- **Pos** Positive Affect score
- **Neg** Negative Affect score

### plot.bmgarch

**Plot method for bmgarch objects.**

#### Description

Plot method for bmgarch objects.

#### Usage

```r
## S3 method for class 'bmgarch'
plot(x, type = "mean", askNewPage = TRUE, CrI = c(0.025, 0.975), ...)
```

#### Arguments

- **x** bmgarch object.
- **type** String (Default: "mean"). Whether to plot conditional means ("mean"), variance ("var"), or correlations ("cor").
- **askNewPage** Logical (Default: True). Whether to ask for new plotting page.
- **CrI** Crl Numeric vector (Default: c(0.025, 0.975)). Lower and upper bound of predictive credible interval.
- ... Not used

#### Value

List of ggplot objects (one per time series).
plot.forecast.bmgarch

Author(s)

Stephen R. Martin

Description

Plot method for forecast.bmgarch objects.

Usage

## S3 method for class 'forecast.bmgarch'
plot(x, type = "mean", askNewPage = TRUE, last_t = 100, ...)

Arguments

x

forecast.bmgarch object. See forecast.bmgarch.

type

String (Default: "mean"). Whether to plot conditional means ("mean"), variance ("var"), or correlations ("cor").

askNewPage

Logical (Default: True). Whether to ask for new plotting page.

last_t

Integer (Default: 100). Only show last_t observations in plot.

...

Not used

Value

List of ggplot objects (one per time series).

Author(s)

Stephen R. Martin

print.fitted.bmgarch

Print method for fitted.bmgarch objects.

Description

Print method for fitted.bmgarch objects.

Usage

## S3 method for class 'fitted.bmgarch'
print(x, ...)

...
Arguments

x fitted.bmgarch object.

... Not used.

Value

object (invisible).

Author(s)

Stephen R. Martin
print.loo.bmgarch

print method for lfocv

Description

print method for lfocv

Usage

## S3 method for class 'loo.bmgarch'
print(x, ...)

Arguments

x  
lfoc object
...
Not used.

Value

Invisible lfocv object

Author(s)

philippe

print.model_weights

Print method for model_weights

Description

Print method for model_weights

Usage

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

Arguments

x  
Model weights object
...
Not used.

Value

model_weights objects with weights, list of log-likelihoods, and r_eff_list
print.summary.bmgarch  Print method for bmgarch.summary objects.

Description
Print method for bmgarch.summary objects.

Usage
## S3 method for class 'summary.bmgarch'
print(x, ...)

Arguments
x summary.bmgarch object.
...
Not used.

Value
x (invisible).

Author(s)
Philippe Rast, Stephen R. Martin

stocks  Daily data on returns of Toyota, Nissan, and Honda stocks.

Description
A dataset used by Stata to illustrate MGARCH models containing daily data on returns of Toyota, Nissan, and Honda stocks.

Usage

stocks
Format

Data frame with 2015 rows and 5 variables:

- **date** Date
- **t** Sequential time index
- **toyota** Daily returns for Toyota stock
- **nissan** Daily returns for Nissan stock
- **honda** Daily returns for Honda stock

---

Summary method for `bmgarch` objects.

Description

Computes posterior summaries for all parameters of interest for `bmgarch` objects.

Usage

```r
## S3 method for class 'bmgarch'
summary(object, CrI = c(0.025, 0.975), digits = 2, ...)
```

Arguments

- **object** `bmgarch` object.
- **CrI** Numeric vector (Default: `c(0.025, 0.975)`). Lower and upper bound of predictive credible interval.
- **digits** Integer (Default: 2, optional). Number of digits to round to when printing.
- **...** Not used.

Value

A named list containing "meta" and "model_summary". `model_summary` contains summary table for all model parameters.

Author(s)

Stephen R. Martin, Philippe Rast
Index

* datasets
  panas, 14
  stocks, 18

as.data.frame.fitted.bmgarch, 3
as.data.frame.forecast.bmgarch, 3

bmgarch, 4
bmgarch-package, 2
bmgarch_list, 7

fitted.bmgarch, 4, 7, 10
forecast(forecast.bmgarch), 9
forecast.bmgarch, 8, 9, 15, 16

loo(loo.bmgarch), 11
loo.bmgarch, 11

model_weights, 12

panas, 14
plot.bmgarch, 14
plot.forecast.bmgarch, 15
print.fitted.bmgarch, 15
print.forecast.bmgarch, 16
print.loo.bmgarch, 17
print.model_weights, 17
print.summary.bmgarch, 18

sampling, 9
stan, 5
stocks, 18
summary.bmgarch, 19