Package: gasmodel (via r-universe)

September 3, 2024

Type Package

Title Generalized Autoregressive Score Models

Version 0.6.0.9000

Description Estimation, forecasting, and simulation of generalized
autoregressive score (GAS) models of Creal, Koopman, and Lucas
(2013) <doi:10.1002/jae.1279> and Harvey (2013)
<doi:10.1017/cbo9781139540933>. Model specification allows for
various data types and distributions, different
parametrizations, exogenous variables, joint and separate
modeling of exogenous variables and dynamics, higher score and
autoregressive orders, custom and unconditional initial values
of time-varying parameters, fixed and bounded values of
coefficients, and missing values. Model estimation is performed
by the maximum likelihood method.

License GPL-3

Encoding UTF-8

LazyData true

Imports abind, arrangements, CircStats, copula, dplyr, ggplot2,
Matrix, mvnfast, nloptr, numDeriv, pracma, tidyr

Suggests hms, knitr, rmarkdown, testthat

RoxygenNote 7.3.1

Roxygen list(old_usage = TRUE)

Depends R (>= 2.10)

VignetteBuilder knitr

Config/testthat/edition 3

URL https://github.com/vladimirholy/gasmodel

BugReports https://github.com/vladimirholy/gasmodel/issues

Repository https://vladimirholy.r-universe.dev

RemoteUrl https://github.com/vladimirholy/gasmodel

RemoteRef HEAD

RemoteSha 9d009496e2e1f1aa4ba586ab8d75b957045e6968

Contents

---

bookshop_orders   *Antiquarian Bookshop Orders*

Description

Individual orders of a Czech antiquarian bookshop from June 8, 2018 to December 20, 2018. This dataset is analyzed in Tomanová and Holý (2021).

Usage

```
bookshop_orders
```

Format

A data frame with columns:

id ID of the order.

datetime Date and time of the order.

quantity Number of purchased books.

duration Number of minutes since the last order.

duration_adj Duration since the last order adjusted for diurnal pattern.

References

Tomanová, P. and Holý, V. (2021). Clustering of Arrivals in Queueing Systems: Autoregressive Conditional Duration Approach. *Central European Journal of Operations Research*, **29**(3), 859–874. doi:10.1007/s10100021007447.

---

distr   *Get the Table of Supported Distributions*

Description

A function listing distributions and their parametrizations supported by the `gas()` function. Output can be filtered using several arguments.

Usage

```
distr(filter_distr = NULL, filter_param = NULL, filter_type = NULL,
  filter_dim = NULL, filter_orthog = NULL, filter_default = NULL)
```

Arguments

`filter_distr`	An optional vector of distributions by which the output is filtered.
`filter_param`	An optional vector of parametrizations by which the output is filtered.
`filter_type`	An optional vector of data types by which the output is filtered.
`filter_dim`	An optional vector of dimensions by which the output is filtered.
`filter_orthog`	An optional logical value indicating whether the parametrization is orthogonal by which the output is filtered.
`filter_default`	An optional logical value indicating whether the parameterization is the default for the distribution by which the output is filtered.

Value

A `data.frame` with columns:

`distr_title`	The title of the distribution.
`param_title`	The title of the parametrization.
`distr`	The distribution.
`param`	The parametrization.
`type`	The data type.
`dim`	The dimension.
`orthog`	The indication of whether the parametrization is orthogonal.
`default`	The indication of whether the parameterization is the default for the distribution.

See Also

distr_density(), distr_mean(), distr_var(), distr_score(), distr_fisher(), distr_random(),
gas()

Examples

```
# List all available distributions
distr()

# List only distributions for count data
distr(filter_type = "count")

# Show default parametrization for the negative binomial distribution
distr(filter_dist = "negbin", filter_default = TRUE)
```

---

distr_density   *Compute Density*

Description

A function computing density or its logarithm of a given distribution.

Usage

```
distr_density(y, f, distr, param = NULL, par_link = NULL, trans = NULL)
```

Arguments

y	Observations. For an univariate distribution, a numeric vector. For a multivariate distribution, a numeric matrix with observations in rows or a numeric vector of a single observation.
f	Parameters. For the same parameters for all observations, a numeric vector. For individual parameters for each observation, a numeric matrix with rows corresponding to observations.
distr	A distribution.
param	A parametrization of the distribution.
par_link	An optional logical vector indicating whether the logarithmic/logistic link should be applied to restricted parameters in order to obtain unrestricted values. Defaults to keeping the original link for all parameters.
trans	An optional transformation of the density. The supported transformation is the logarithm of the density (trans = "log").

Value

The (transformed) density.

See Also

[distr()](#)

Examples

```
# Density of the negative binomial distribution
distr_density(y = c(1, 8, 5, 0, 0), f = c(13.50, 0.03), distr = "negbin")
```

---

distr_fisher	*Compute Fisher Information*

Description

A function computing Fisher information, its inverse, or its inverse square root for a given distribution.

Usage

```
distr_fisher(f, distr, param = NULL, par_link = NULL, trans = NULL)
```

Arguments

f	Parameters. For the same parameters for all observations, a numeric vector. For individual parameters for each observation, a numeric matrix with rows corresponding to observations.
distr	A distribution.
param	A parametrization of the distribution.
par_link	An optional logical vector indicating whether the logarithmic/logistic link should be applied to restricted parameters in order to obtain unrestricted values. Defaults to keeping the original link for all parameters.
trans	An optional transformation of the Fisher information. The supported transformations are the inverse of the Fisher information (trans = "inv") and the inverse square root of the Fisher information (trans = "inv_sqrt").

Value

The (transformed) Fisher information.

See Also

[distr()](#)

Examples

```
# Fisher information for the negative binomial distribution
distr_fisher(f = c(13.50, 0.03), distr = "negbin")
```

distr_mean *Compute Mean*

Description

A function computing mean for a given distribution.

Usage

```
distr_mean(f, distr, param = NULL, par_link = NULL)
```

Arguments

f	Parameters. For the same parameters for all observations, a numeric vector. For individual parameters for each observation, a numeric matrix with rows corresponding to observations.
distr	A distribution.
param	A parametrization of the distribution.
par_link	An optional logical vector indicating whether the logarithmic/logistic link should be applied to restricted parameters in order to obtain unrestricted values. Defaults to keeping the original link for all parameters.

Value

The mean.

See Also

[distr()](distr)

Examples

```
# Mean for the negative binomial distribution
distr_mean(f = c(13.50, 0.03), distr = "negbin")
```

---

distr_random   *Generate Random Observations*

Description

A function generating random observations from a given distribution.

Usage

```
distr_random(t, f, distr, param = NULL, par_link = NULL)
```

Arguments

t	A number of generated observations.
f	A numeric vector of parameters. The same parameters are used for each generated observation.
distr	A distribution.
param	A parametrization of the distribution.
par_link	An optional logical vector indicating whether the logarithmic/logistic link should be applied to restricted parameters in order to obtain unrestricted values. Defaults to keeping the original link for all parameters.

Value

The generated observations.

See Also

[distr()]

Examples

```
# Random observations from the negative binomial distribution
distr_random(t = 10, f = c(13.50, 0.03), distr = "negbin")
```

---

distr_score   *Compute Score*

Description

A function computing score or scaled score for a given distribution.

Usage

```
distr_score(y, f, distr, param = NULL, par_link = NULL, scaling = NULL)
```

Arguments

y	Observations. For an univariate distribution, a numeric vector. For a multivariate distribution, a numeric matrix with observations in rows or a numeric vector of a single observation.
f	Parameters. For the same parameters for all observations, a numeric vector. For individual parameters for each observation, a numeric matrix with rows corresponding to observations.
distr	A distribution.
param	A parametrization of the distribution.
par_link	An optional logical vector indicating whether the logarithmic/logistic link should be applied to restricted parameters in order to obtain unrestricted values. Defaults to keeping the original link for all parameters.
scaling	An optional scaling function for the score. The supported scaling functions are the unit scaling (scaling = "unit"), the inverse of the Fisher information matrix scaling (scaling = "fisher_inv"), and the inverse square root of the Fisher information matrix scaling (scaling = "fisher_inv_sqrt").

Value

The (scaled) score.

See Also

[distr()](#)

Examples

```
# Score for the negative binomial distribution
distr_score(y = c(1, 8, 5, 0, 0), f = c(13.50, 0.03), distr = "negbin")
```

distr_var   *Compute Variance*

Description

A function computing variance for a given distribution.

Usage

```
distr_var(f, distr, param = NULL, par_link = NULL)
```

Arguments

f
: Parameters. For the same parameters for all observations, a numeric vector. For individual parameters for each observation, a numeric matrix with rows corresponding to observations.

distr
: A distribution.

param
: A parametrization of the distribution.

par_link
: An optional logical vector indicating whether the logarithmic/logistic link should be applied to restricted parameters in order to obtain unrestricted values. Defaults to keeping the original link for all parameters.

Value

The variance.

See Also

distr()

Examples

```
# Variance for the negative binomial distribution
distr_var(f = c(13.50, 0.03), distr = "negbin")
```

---

gas   *Estimate GAS Model*

Description

A versatile function for estimation of generalized autoregressive score (GAS) models of Creal et al. (2013) and Harvey (2013). Model specification allows for various conditional distributions, different parametrizations, exogenous variables, higher score and autoregressive orders, custom and unconditional initial values of time-varying parameters, fixed and bounded values of coefficients, and NA values. Model estimation is performed by the maximum likelihood method and the Hessian matrix. The function can be supplied with any optimization and Hessian functions.

Usage

```
gas(y, x = NULL, distr, param = NULL, scaling = "unit",
  regress = "joint", p = 1L, q = 1L, par_static = NULL,
  par_link = NULL, par_init = NULL, lik_skip = 0L,
  coef_fix_value = NULL, coef_fix_other = NULL, coef_fix_special = NULL,
  coef_bound_lower = NULL, coef_bound_upper = NULL, coef_start = NULL,
  optim_function = wrapper_optim_nloptr, optim_arguments = list(opts =
  list(algorithm = "NLOPT_LN_NELDERMEAD", xtol_rel = 0, maxeval = 1e+06)),
  hessian_function = wrapper_hessian_stats, hessian_arguments = list(),
  print_progress = FALSE)
```

Arguments

y	A time series. For univariate time series, a numeric vector or a matrix with a single column. For multivariate times series, a numeric matrix with observations in rows.
x	Optional exogenous variables. For a single variable common for all time-varying parameters, a numeric vector. For multiple variables common for all time-varying parameters, a numeric matrix with observations in rows. For individual variables for each time-varying parameter, a list of numeric vectors or matrices in the above form. The number of observation must be equal to the number of observations of y.
distr	A conditional distribution. See distr() for available distributions.
param	A parametrization of the conditional distribution. If NULL, default parametrization is used. See distr() for available parametrizations.
scaling	A scaling function for the score. The supported scaling functions are the unit scaling (scaling = "unit"), the inverse of the Fisher information matrix scaling (scaling = "fisher_inv"), and the inverse square root of the Fisher information matrix scaling (scaling = "fisher_inv_sqrt"). The latter two scalings use the Fisher information for the time-varying parameters only. For the full Fisher information matrix for both time-varying and static parameters, there are the "full_fisher_inv" and "full_fisher_inv_sqrt" scalings. For the individual Fisher information for each parameter, there are the "diag_fisher_inv"

and `"diag_fisher_inv_sqrt"` scalings. Note that when the parametrization is orthogonal (see [`distr()`](distr())), there are no differences between these scaling variants.

regress
: A specification of the regression and dynamic equation with regard to exogenous variables. The supported specifications are exogenous variables and dynamics within the same equation (regress = `"joint"`) and separate equations for exogenous variables and dynamics in the fashion of regression models with dynamic errors (regress = `"sep"`). In a stationary model without exogenous variables, the two specifications are equivalent, although with differently parametrized intercept.

p
: A score order. For order common for all parameters, a numeric vector of length 1. For individual order for each parameter, a numeric vector of length equal to the number of parameters. Defaults to `1L`.

q
: An autoregressive order. For order common for all parameters, a numeric vector of length 1. For individual order for each parameter, a numeric vector of length equal to the number of parameters. Defaults to `1L`.

par_static
: An optional logical vector indicating static parameters. Overrides x, p, and q.

par_link
: An optional logical vector indicating whether the logarithmic/logistic link should be applied to restricted parameters in order to obtain unrestricted values. Defaults to applying the logarithmic/logistic link for time-varying parameters and keeping the original link for constant parameters.

par_init
: An optional numeric vector of initial values of time-varying parameters. For NA values or when NULL, set initial values to unconditional values of time-varying parameters. For example, in the case of GAS(1,1) model with regress = `"joint"`, to omega / (1 - phi1). Not to be confused with starting values for the optimization `coef_start`.

lik_skip
: A numeric value specifying the number of skipped observations at the beginning of the time series or after NA values in the likelihood computation. Defaults to `0L`, i.e. the full likelihood. If NULL, it is selected as `max(p, q)`, i.e. the conditional likelihood.

coef_fix_value
: An optional numeric vector of values to which coefficients are to be fixed. NA values represent coefficients to be estimated.

coef_fix_other
: An optional square numeric matrix of multiples of the estimated coefficients, which are to be added to the fixed coefficients. This allows the fixed coefficients to be linear combinations of the estimated coefficients. A coefficient given by row is fixed on coefficient given by column. By this logic, all rows corresponding to the estimated coefficients should contain only NA values. Furthermore, all columns corresponding to the fixed coefficients should also contain only NA values.

coef_fix_special

: An optional character vector of predefined structures of `coef_fix_value` and `coef_fix_other`. Useful mainly for multidimensional models. Value `"panel_structure"` forces all regression, autoregression, and score coefficients to be the same for all time-varying parameters within their group. Value `"zero_sum_intercept"` forces all constant parameters to sum up to zero within their group. Value `"random_walk"` forces all autoregressive coefficients to be equal to one (should

be used with caution due to nonstationarity; `par_init` must be specified). Multiple predefined structures can be used together. Also can be used in combination with custom `coef_fix_value` and `coef_fix_other`.

coef_bound_lower

An optional numeric vector of lower bounds on coefficients.

coef_bound_upper

An optional numeric vector of upper bounds on coefficients.

coef_start   An optional numeric vector of starting values for the optimization. If not supplied, starting values are selected from a small grid of values.

optim_function   An optimization function. For suitable wrappers of common R optimization functions, see `wrappers_optim`. Can be set to NULL if the optimal solution should not be computed, which can be useful if the goal is only to evaluate the fit for the coefficients specified in argument `coef_start`.

optim_arguments

An optional list of arguments to be passed to the optimization function.

hessian_function

A Hessian function. For suitable wrappers of common R Hessian functions, see `wrappers_hessian`. Can be set to NULL if the Hessian matrix should not be computed, which can speed up computations when asymptotic inference is not desired.

hessian_arguments

An optional list of arguments to be passed to the Hessian function.

print_progress A logical value indicating whether to progressively print a detailed report on computation.

Details

The generalized autoregressive score (GAS) models of Creal et al. (2013) and Harvey (2013), also known as dynamic conditional score (DCS) models or score-driven (SD) models, have established themselves as a useful modern framework for time series modeling.

The GAS models are observation-driven models allowing for any underlying probability distribution $p(y_t|f_t)$ with any time-varying parameters f_t for time series y_t. They capture the dynamics of time-varying parameters using the autoregressive term and the lagged score, i.e. the gradient of the log-likelihood function. Exogenous variables can also be included. Specifically, time-varying parameters f_t follow the recursion

$$f_t = \omega + \sum_{i=1}^{M} \beta_i x_{ti} + \sum_{j=1}^{P} \alpha_j S(f_{t-j}) \nabla(y_{t-j}, f_{t-j}) + \sum_{k=1}^{Q} \varphi_k f_{t-k},$$

where ω is the intercept, β_i are the regression parameters, α_j are the score parameters, φ_k are the autoregressive parameters, x_{ti} are the exogenous variables, $S(f_t)$ is a scaling function for the score, and $\nabla(y_t, f_t)$ is the score given by

$$\nabla(y_t, f_t) = \frac{\partial \ln p(y_t|f_t)}{\partial f_t}.$$

In the case of a single time-varying parameter, ω, β_i, α_j, φ_k, x_{ti}, $S(f_t)$, and $\nabla(y_t, f_t)$ are all scalar. In the case of multiple time-varying parameters, x_{ti} are scalar, ω, β_i, and $\nabla(y_{t-j}, f_{t-j})$ are vectors,

α_j and φ_k are diagonal matrices, and $S(f_t)$ is a square matrix. Alternatively, a different model can be obtained by defining the recursion in the fashion of regression models with dynamic errors as

$$f_t = \omega + \sum_{i=1}^{M} \beta_i x_{ti} + e_t, \quad e_t = \sum_{j=1}^{P} \alpha_j S(f_{t-j}) \nabla(y_{t-j}, f_{t-j}) + \sum_{k=1}^{Q} \varphi_k e_{t-k}.$$

The GAS models can be straightforwardly estimated by the maximum likelihood method. For the asymptotic theory regarding the GAS models and maximum likelihood estimation, see Blasques et al. (2014), Blasques et al. (2018), and Blasques et al. (2022).

The use of the score for updating time-varying parameters is optimal in an information theoretic sense. For an investigation of the optimality properties of GAS models, see Blasques et al. (2015) and Blasques et al. (2021).

Generally, the GAS models perform quite well when compared to alternatives, including parameter-driven models. For a comparison of the GAS models to alternative models, see Koopman et al. (2016) and Blazsek and Licht (2020).

The GAS class includes many well-known econometric models, such as the generalized autoregressive conditional heteroskedasticity (GARCH) model of Bollerslev (1986), the autoregressive conditional duration (ACD) model of Engle and Russel (1998), and the Poisson count model of Davis et al. (2003). More recently, a variety of novel score-driven models has been proposed, such as the Beta-t-(E)GARCH model of Harvey and Chakravarty (2008), the discrete price changes model of Koopman et al. (2018), the circular model of Harvey (2019), the bivariate Poisson model of Koopman and Lit (2019), and the ranking model of Holý and Zouhar (2022). For an overview of various GAS models, see Harvey (2022).

The extensive GAS literature is listed on www.gasmodel.com.

Value

A list of S3 class gas with components:

data$y	The time series.
data$x	The exogenous variables.
model$distr	The conditional distribution.
model$param	The parametrization of the conditional distribution.
model$scaling	The scaling function.
model$regress	The specification of the regression and dynamic equation.
model$t	The length of the time series.
model$n	The dimension of the model.
model$m	The number of exogenous variables.
model$p	The score order.
model$q	The autoregressive order.
model$par_static	
	The static parameters.
model$par_link	The parameters with the logarithmic/logistic links.

model$par_init   The initial values of the time-varying parameters.

model$lik_skip   The number of skipped observations at the beginning of the time series or after
                  NA values in the likelihood computation.

model$coef_fix_value

                  The values to which coefficients are fixed.

model$coef_fix_other

                  The multiples of the estimated coefficients, which are added to the fixed coeffi-
                  cients.

model$coef_fix_special

                  The predefined structures of `coef_fix_value` and `coef_fix_other`.

model$coef_bound_lower

                  The lower bounds on coefficients.

model$coef_bound_upper

                  The upper bounds on coefficients.

model$num_obs   The actual number of observations used in the likelihood.

model$num_coef   The actual number of estimated coefficients.

control$optim_function

                  The optimization function.

control$optim_arguments

                  The arguments which are passed to the optimization function.

control$hessian_function

                  The Hessian function.

control$hessian_arguments

                  The arguments which are passed to the Hessian function.

solution$status_start

                  The status of the starting values computation.

solution$theta_start

                  The computed starting values.

solution$status_optim

                  The status of the optimization computation.

solution$theta_optim

                  The computed optimal values.

solution$status_hessian

                  The status of the Hessian computation.

solution$theta_hessian

                  The computed Hessian.

fit$coef_est   The estimated coefficients.

fit$coef_vcov   The estimated variance-covariance matrix.

fit$coef_sd   The estimated standard deviations.

fit$coef_zstat   The statistics of the Z-test.

fit$coef_pval   The p-values of the Z-test.

fit$par_unc   The unconditional values of time-varying parameters.

fit$par_tv   The individual values of time-varying parameter.

`fit$score_tv`	The individual scores of time-varying parameters.
`fit$mean_tv`	The expected values given by the model.
`fit$var_tv`	The variances given by the model.
`fit$resid_tv`	The residuals of the model.
`fit$loglik_tv`	The log-likelihoods for the individual observations.
`fit$loglik_sum`	The overall log-likelihood.
`fit$aic`	The Akaike information criterion.
`fit$bic`	The Bayesian information criterion.

Note

Supported generic functions for S3 class gas include summary(), plot(), coef(), vcov(), fitted(), residuals(), logLik(), AIC(), BIC(), and confint().

References

Blasques, F., Gorgi, P., Koopman, S. J., and Wintenberger, O. (2018). Feasible Invertibility Conditions and Maximum Likelihood Estimation for Observation-Driven Models. *Electronic Journal of Statistics*, **12**(1), 1019–1052. doi:10.1214/18ejs1416.

Blasques, F., Koopman, S. J., and Lucas, A. (2014). Stationarity and Ergodicity of Univariate Generalized Autoregressive Score Processes. *Electronic Journal of Statistics*, **8**(1), 1088–1112. doi:10.1214/14ejs924.

Blasques, F., Koopman, S. J., and Lucas, A. (2015). Information-Theoretic Optimality of Observation-Driven Time Series Models for Continuous Responses. *Biometrika*, **102**(2), 325–343. doi:10.1093/biomet/asu076.

Blasques, F., Lucas, A., and van Vlodrop, A. C. (2021). Finite Sample Optimality of Score-Driven Volatility Models: Some Monte Carlo Evidence. *Econometrics and Statistics*, **19**, 47–57. doi:10.1016/j.ecosta.2020.03.010.

Blasques, F., van Brummelen, J., Koopman, S. J., and Lucas, A. (2022). Maximum Likelihood Estimation for Score-Driven Models. *Journal of Econometrics*, **227**(2), 325–346. doi:10.1016/j.jeconom.2021.06.003.

Blazsek, S. and Licht, A. (2020). Dynamic Conditional Score Models: A Review of Their Applications. *Applied Economics*, **52**(11), 1181–1199. doi:10.1080/00036846.2019.1659498.

Bollerslev, T. (1986). Generalized Autoregressive Conditional Heteroskedasticity. *Journal of Econometrics*, **31**(3), 307–327. doi:10.1016/03044076(86)900631.

Creal, D., Koopman, S. J., and Lucas, A. (2013). Generalized Autoregressive Score Models with Applications. *Journal of Applied Econometrics*, **28**(5), 777–795. doi:10.1002/jae.1279.

Davis, R. A., Dunsmuir, W. T. M., and Street, S. B. (2003). Observation-Driven Models for Poisson Counts. *Biometrika*, **90**(4), 777–790. doi:10.1093/biomet/90.4.777.

Engle, R. F. and Russell, J. R. (1998). Autoregressive Conditional Duration: A New Model for Irregularly Spaced Transaction Data. *Econometrica*, **66**(5), 1127–1162. doi:10.2307/2999632.

Harvey, A. C. (2013). *Dynamic Models for Volatility and Heavy Tails: With Applications to Financial and Economic Time Series*. Cambridge University Press. doi:10.1017/cbo9781139540933.

Harvey, A. C. (2022). Score-Driven Time Series Models. *Annual Review of Statistics and Its Application*, **9**(1), 321–342. doi:10.1146/annurevstatistics040120021023.

Harvey, A. C. and Chakravarty, T. (2008). Beta-t-(E)GARCH. *Cambridge Working Papers in Economics*, CWPE 0840. doi:10.17863/cam.5286.

Harvey, A., Hurn, S., and Thiele, S. (2019). Modeling Directional (Circular) Time Series. *Cambridge Working Papers in Economics*, CWPE 1971. doi:10.17863/cam.43915.

Holý, V. and Zouhar, J. (2022). Modelling Time-Varying Rankings with Autoregressive and Score-Driven Dynamics. Journal of the Royal Statistical Society: Series C (Applied Statistics), **71**(5). doi:10.1111/rssc.12584.

Koopman, S. J. and Lit, R. (2019). Forecasting Football Match Results in National League Competitions Using Score-Driven Time Series Models. *International Journal of Forecasting*, **35**(2), 797–809. doi:10.1016/j.ijforecast.2018.10.011.

Koopman, S. J., Lit, R., Lucas, A., and Opschoor, A. (2018). Dynamic Discrete Copula Models for High-Frequency Stock Price Changes. *Journal of Applied Econometrics*, **33**(7), 966–985. doi:10.1002/jae.2645.

Koopman, S. J., Lucas, A., and Scharth, M. (2016). Predicting Time-Varying Parameters with Parameter-Driven and Observation-Driven Models. *Review of Economics and Statistics*, **98**(1), 97–110. doi:10.1162/rest_a_00533.

See Also

distr(), gas_bootstrap(), gas_filter(), gas_forecast(), gas_simulate(), wrappers_optim, wrappers_hessian

Examples

```
# Load the Daily Toilet Paper Sales dataset
data("toilet_paper_sales")
y <- toilet_paper_sales$quantity
x <- as.matrix(toilet_paper_sales[3:9])

# Estimate GAS model based on the negative binomial distribution
est_negbin <- gas(y = y, x = x, distr = "negbin", regress = "sep")
est_negbin

# Obtain the estimated coefficients
coef(est_negbin)

# Obtain the estimated variance-covariance matrix
vcov(est_negbin)

# Obtain the log-likelihood, AIC, and BIC
logLik(est_negbin)
AIC(est_negbin)
BIC(est_negbin)

# Obtain the confidence intervals of coefficients
confint(est_negbin)
```

```
# Plot the time-varying parameters
plot(est_negbin)
```

---

gas_bootstrap   *Bootstrap GAS Model*

Description

A function for bootstrapping coefficients of generalized autoregressive score (GAS) models of Creal
et al. (2013) and Harvey (2013). Method "parametric" repeatedly simulates time series using the
parametric model and re-estimates coefficients. Methods "simple_block", "moving_block", and
"stationary_block" perform the standard variations of the circular block bootstrap. Instead of
supplying arguments about the model, the function can be applied to the gas object obtained by the
[gas()](#) function. The function enables parallelization.

Usage

```
gas_bootstrap(gas_object = NULL, method = "parametric", rep_boot = 1000L,
  block_length = NULL, quant = c(0.025, 0.975), y = NULL, x = NULL,
  distr = NULL, param = NULL, scaling = "unit", regress = "joint",
  p = 1L, q = 1L, par_static = NULL, par_link = NULL,
  par_init = NULL, lik_skip = 0L, coef_fix_value = NULL,
  coef_fix_other = NULL, coef_fix_special = NULL,
  coef_bound_lower = NULL, coef_bound_upper = NULL, coef_est = NULL,
  optim_function = wrapper_optim_nloptr, optim_arguments = list(opts =
  list(algorithm = "NLOPT_LN_NELDERMEAD", xtol_rel = 0, maxeval = 1e+06)),
  parallel_function = NULL, parallel_arguments = list())
```

Arguments

gas_object	An optional GAS estimate, i.e. a list of S3 class gas returned by function [gas()](#).
method	A method used for bootstrapping. Supported methods are "parametric", "simple_block", "moving_block", and "stationary_block".
rep_boot	A number of bootstrapping repetitions.
block_length	A length of blocks for methods "simple_block" and "moving_block". A mean length of blocks for method "stationary_block".
quant	A numeric vector of probabilities determining quantiles.
y	A time series. For univariate time series, a numeric vector or a matrix with a single column. For multivariate times series, a numeric matrix with observations in rows.
x	Optional exogenous variables. For a single variable common for all time-varying parameters, a numeric vector. For multiple variables common for all time-varying parameters, a numeric matrix with observations in rows. For individual variables for each time-varying parameter, a list of numeric vectors or matrices in the above form. The number of observation must be equal to the number of observations of y.

distr	A conditional distribution. See distr() for available distributions.
param	A parametrization of the conditional distribution. If NULL, default parametrization is used. See distr() for available parametrizations.
scaling	A scaling function for the score. The supported scaling functions are the unit scaling (scaling = "unit"), the inverse of the Fisher information matrix scaling (scaling = "fisher_inv"), and the inverse square root of the Fisher information matrix scaling (scaling = "fisher_inv_sqrt"). The latter two scalings use the Fisher information for the time-varying parameters only. For the full Fisher information matrix for both time-varying and static parameters, there are the "full_fisher_inv" and "full_fisher_inv_sqrt" scalings. For the individual Fisher information for each parameter, there are the "diag_fisher_inv" and "diag_fisher_inv_sqrt" scalings. Note that when the parametrization is orthogonal (see distr()), there are no differences between these scaling variants.
regress	A specification of the regression and dynamic equation with regard to exogenous variables. The supported specifications are exogenous variables and dynamics within the same equation (regress = "joint") and separate equations for exogenous variables and dynamics in the fashion of regression models with dynamic errors (regress = "sep"). In a stationary model without exogenous variables, the two specifications are equivalent, although with differently parametrized intercept.
p	A score order. For order common for all parameters, a numeric vector of length 1. For individual order for each parameter, a numeric vector of length equal to the number of parameters. Defaults to 1L.
q	An autoregressive order. For order common for all parameters, a numeric vector of length 1. For individual order for each parameter, a numeric vector of length equal to the number of parameters. Defaults to 1L.
par_static	An optional logical vector indicating static parameters. Overrides x, p, and q.
par_link	An optional logical vector indicating whether the logarithmic/logistic link should be applied to restricted parameters in order to obtain unrestricted values. Defaults to applying the logarithmic/logistic link for time-varying parameters and keeping the original link for constant parameters.
par_init	An optional numeric vector of initial values of time-varying parameters. For NA values or when NULL, set initial values to unconditional values of time-varying parameters. For example, in the case of GAS(1,1) model with regress = "joint", to omega / (1 - phi1). Not to be confused with starting values for the optimization coef_start.
lik_skip	A numeric value specifying the number of skipped observations at the beginning of the time series or after NA values in the likelihood computation. Defaults to 0L, i.e. the full likelihood. If NULL, it is selected as max(p,q), i.e. the conditional likelihood.
coef_fix_value	An optional numeric vector of values to which coefficients are to be fixed. NA values represent coefficients to be estimated.
coef_fix_other	An optional square numeric matrix of multiples of the estimated coefficients, which are to be added to the fixed coefficients. This allows the fixed coefficients

to be linear combinations of the estimated coefficients. A coefficient given by row is fixed on coefficient given by column. By this logic, all rows corresponding to the estimated coefficients should contain only NA values. Furthermore, all columns corresponding to the fixed coefficients should also contain only NA values.

coef_fix_special

An optional character vector of predefined structures of `coef_fix_value` and `coef_fix_other`. Useful mainly for multidimensional models. Value `"panel_structure"` forces all regression, autoregression, and score coefficients to be the same for all time-varying parameters within their group. Value `"zero_sum_intercept"` forces all constant parameters to sum up to zero within their group. Value `"random_walk"` forces all autoregressive coefficients to be equal to one (should be used with caution due to nonstationarity; `par_init` must be specified). Multiple predefined structures can be used together. Also can be used in combination with custom `coef_fix_value` and `coef_fix_other`.

coef_bound_lower

An optional numeric vector of lower bounds on coefficients.

coef_bound_upper

An optional numeric vector of upper bounds on coefficients.

coef_est   A numeric vector of estimated coefficients.

optim_function   An optimization function. For suitable wrappers of common R optimization functions, see [wrappers_optim](). Can be set to NULL if the optimal solution should not be computed, which can be useful if the goal is only to evaluate the fit for the coefficients specified in argument `coef_start`.

optim_arguments

An optional list of arguments to be passed to the optimization function.

parallel_function

A parallelization function. For suitable wrappers of common R parallelization functions, see [wrappers_parallel](). Can be set to NULL if no parallelization is to be used.

parallel_arguments

An optional list of arguments to be passed to the optimization function.

Value

A `list` of S3 class `gas_bootstrap` with components:

data$y	The time series.
data$x	The exogenous variables.
model$distr	The conditional distribution.
model$param	The parametrization of the conditional distribution.
model$scaling	The scaling function.
model$regress	The specification of the regression and dynamic equation.
model$t	The length of the time series.
model$n	The dimension of the model.

model$m	The number of exogenous variables.
model$p	The score order.
model$q	The autoregressive order.
model$par_static	
	The static parameters.
model$par_link	The parameters with the logarithmic/logistic links.
model$par_init	The initial values of the time-varying parameters.
model$lik_skip	The number of skipped observations at the beginning of the time series or after NA values in the likelihood computation.
model$coef_fix_value	
	The values to which coefficients are fixed.
model$coef_fix_other	
	The multiples of the estimated coefficients, which are added to the fixed coefficients.
model$coef_fix_special	
	The predefined structures of coef_fix_value and coef_fix_other.
model$coef_bound_lower	
	The lower bounds on coefficients.
model$coef_bound_upper	
	The upper bounds on coefficients.
model$coef_est	The estimated coefficients.
bootstrap$method	
	The method used for bootstrapping.
bootstrap$coef_set	
	The bootstrapped sets of coefficients.
bootstrap$coef_mean	
	The mean of bootstrapped coefficients.
bootstrap$coef_vcov	
	The variance-covariance matrix of bootstrapped coefficients.
bootstrap$coef_sd	
	The standard deviation of bootstrapped coefficients.
bootstrap$coef_pval	
	The p-value of bootstrapped coefficients.
bootstrap$coef_quant	
	The quantiles of bootstrapped coefficients.

Note

Supported generic functions for S3 class gas_bootstrap include summary(), plot(), coef(), and vcov().

References

Creal, D., Koopman, S. J., and Lucas, A. (2013). Generalized Autoregressive Score Models with Applications. *Journal of Applied Econometrics*, **28**(5), 777–795. doi:10.1002/jae.1279.

Harvey, A. C. (2013). *Dynamic Models for Volatility and Heavy Tails: With Applications to Financial and Economic Time Series*. Cambridge University Press. doi:10.1017/cbo9781139540933.

See Also

[gas()](gas()) wrappers_parallel

Examples

```
# Load the Daily Toilet Paper Sales dataset
data("toilet_paper_sales")
y <- toilet_paper_sales$quantity
x <- as.matrix(toilet_paper_sales[3:9])

# Estimate GAS model based on the negative binomial distribution
est_negbin <- gas(y = y, x = x, distr = "negbin", regress = "sep")
est_negbin

# Bootstrap the model (can be time-consuming for a larger number of samples)
boot_negbin <- gas_bootstrap(est_negbin, rep_boot = 10)
boot_negbin

# Plot boxplot of bootstrapped coefficients
plot(boot_negbin)
```

---

gas_filter   *Filter GAS Model*

Description

A function for obtaining filtered time-varying parameters of generalized autoregressive score (GAS) models of Creal et al. (2013) and Harvey (2013). It captures parameter uncertainty and can also be used for forecasting. Method "simulated_coefs" computes a path of time-varying parameters for each simulated coefficient set under assumption of asymptotic normality with given variance-covariance matrix (see Blasques et al., 2016). Method "given_coefs" computes a path of time-varying parameters for each supplied coefficient set. Instead of supplying arguments about the model, the function can be applied to the gas object obtained by the [gas()](gas()) function.

Usage

```
gas_filter(gas_object = NULL, method = "simulated_coefs",
  coef_set = NULL, rep_gen = 1000L, t_ahead = 0L, x_ahead = NULL,
  rep_ahead = 1000L, quant = c(0.025, 0.975), y = NULL, x = NULL,
  distr = NULL, param = NULL, scaling = "unit", regress = "joint",
  p = 1L, q = 1L, par_static = NULL, par_link = NULL,
  par_init = NULL, coef_fix_value = NULL, coef_fix_other = NULL,
  coef_fix_special = NULL, coef_bound_lower = NULL,
  coef_bound_upper = NULL, coef_est = NULL, coef_vcov = NULL)
```

Arguments

gas_object	An optional GAS estimate, i.e. a list of S3 class gas returned by function [gas()]().
method	A method used for parameter uncertainty. Supported methods are "given_coefs" and "simulated_coefs".
coef_set	A numeric matrix of coefficient sets in rows for method = "given_coefs". Can be generated for example by [gas_bootstrap()]().
rep_gen	A number of generated coefficient sets for method = "simulated_coefs".
t_ahead	A number of observations to forecast.
x_ahead	Out-of-sample exogenous variables. For a single variable common for all time-varying parameters, a numeric vector. For multiple variables common for all time-varying parameters, a numeric matrix with observations in rows. For individual variables for each time-varying parameter, a list of numeric vectors or matrices in the above form. The number of observation must be equal to t_ahead.
rep_ahead	A number of simulation repetitions for forecasting when t_ahead > 0.
quant	A numeric vector of probabilities determining quantiles.
y	A time series. For univariate time series, a numeric vector or a matrix with a single column. For multivariate times series, a numeric matrix with observations in rows.
x	Optional exogenous variables. For a single variable common for all time-varying parameters, a numeric vector. For multiple variables common for all time-varying parameters, a numeric matrix with observations in rows. For individual variables for each time-varying parameter, a list of numeric vectors or matrices in the above form. The number of observation must be equal to the number of observations of y.
distr	A conditional distribution. See [distr()]() for available distributions.
param	A parametrization of the conditional distribution. If NULL, default parametrization is used. See [distr()]() for available parametrizations.
scaling	A scaling function for the score. The supported scaling functions are the unit scaling (scaling = "unit"), the inverse of the Fisher information matrix scaling (scaling = "fisher_inv"), and the inverse square root of the Fisher information matrix scaling (scaling = "fisher_inv_sqrt"). The latter two scalings use the Fisher information for the time-varying parameters only. For the full Fisher information matrix for both time-varying and static parameters, there are the "full_fisher_inv" and "full_fisher_inv_sqrt" scalings. For the individual Fisher information for each parameter, there are the "diag_fisher_inv" and "diag_fisher_inv_sqrt" scalings. Note that when the parametrization is orthogonal (see [distr()]()), there are no differences between these scaling variants.
regress	A specification of the regression and dynamic equation with regard to exogenous variables. The supported specifications are exogenous variables and dynamics within the same equation (regress = "joint") and separate equations for exogenous variables and dynamics in the fashion of regression models with dynamic errors (regress = "sep"). In a stationary model without exogenous variables, the two specifications are equivalent, although with differently parametrized intercept.

p	A score order. For order common for all parameters, a numeric vector of length 1. For individual order for each parameter, a numeric vector of length equal to the number of parameters. Defaults to 1L.
q	An autoregressive order. For order common for all parameters, a numeric vector of length 1. For individual order for each parameter, a numeric vector of length equal to the number of parameters. Defaults to 1L.
par_static	An optional logical vector indicating static parameters. Overrides x, p, and q.
par_link	An optional logical vector indicating whether the logarithmic/logistic link should be applied to restricted parameters in order to obtain unrestricted values. Defaults to applying the logarithmic/logistic link for time-varying parameters and keeping the original link for constant parameters.
par_init	An optional numeric vector of initial values of time-varying parameters. For NA values or when NULL, set initial values to unconditional values of time-varying parameters. For example, in the case of GAS(1,1) model with regress = "joint", to omega / (1 - phi1). Not to be confused with starting values for the optimization coef_start.
coef_fix_value	An optional numeric vector of values to which coefficients are to be fixed. NA values represent coefficients to be estimated.
coef_fix_other	An optional square numeric matrix of multiples of the estimated coefficients, which are to be added to the fixed coefficients. This allows the fixed coefficients to be linear combinations of the estimated coefficients. A coefficient given by row is fixed on coefficient given by column. By this logic, all rows corresponding to the estimated coefficients should contain only NA values. Furthermore, all columns corresponding to the fixed coefficients should also contain only NA values.
coef_fix_special	
	An optional character vector of predefined structures of coef_fix_value and coef_fix_other. Useful mainly for multidimensional models. Value "panel_structure" forces all regression, autoregression, and score coefficients to be the same for all time-varying parameters within their group. Value "zero_sum_intercept" forces all constant parameters to sum up to zero within their group. Value "random_walk" forces all autoregressive coefficients to be equal to one (should be used with caution due to nonstationarity; par_init must be specified). Multiple predefined structures can be used together. Also can be used in combination with custom coef_fix_value and coef_fix_other.
coef_bound_lower	
	An optional numeric vector of lower bounds on coefficients.
coef_bound_upper	
	An optional numeric vector of upper bounds on coefficients.
coef_est	A numeric vector of estimated coefficients.
coef_vcov	A numeric matrix of estimated covariances between coefficients.

Value

A list of S3 class gas_filter with components:

data$y	The time series.

data$x   The exogenous variables.

data$x_ahead   The out-of-sample exogenous variables. Only when t_ahead > 0.

model$distr   The conditional distribution.

model$param   The parametrization of the conditional distribution.

model$scaling   The scaling function.

model$regress   The specification of the regression and dynamic equation.

model$t   The length of the time series.

model$t_ahead   The length of the out-of-sample time series. Only when t_ahead > 0.

model$n   The dimension of the model.

model$m   The number of exogenous variables.

model$p   The score order.

model$q   The autoregressive order.

model$par_static
                  The static parameters.

model$par_link   The parameters with the logarithmic/logistic links.

model$par_init   The initial values of the time-varying parameters.

model$coef_fix_value
                  The values to which coefficients are fixed.

model$coef_fix_other
                  The multiples of the estimated coefficients, which are added to the fixed coeffi-
                  cients.

model$coef_fix_special
                  The predefined structures of coef_fix_value and coef_fix_other.

model$coef_bound_lower
                  The lower bounds on coefficients.

model$coef_bound_upper
                  The upper bounds on coefficients.

model$coef_set   The coefficient sets.

filter$method   The method used for parameter uncertainty.

filter$par_tv_mean
                  The mean of the time-varying parameters.

filter$par_tv_sd
                  The standard deviation of the time-varying parameters.

filter$par_tv_quant
                  The quantiles of the time-varying parameters.

filter$score_tv_mean
                  The mean of the scores.

filter$score_tv_sd
                  The standard deviation of the scores.

filter$score_tv_quant
                  The quantiles of the scores.

```
filter$y_ahead_mean
```
            The mean of the forecasted time series. Only when t_ahead > 0.

```
filter$y_ahead_sd
```
            The standard deviation of the forecasted time series. Only when t_ahead > 0.

```
filter$y_ahead_quant
```
            The quantiles of the forecasted time series. Only when t_ahead > 0.

```
filter$par_tv_ahead_mean
```
            The mean of the forecasted time-varying parameters. Only when t_ahead > 0.

```
filter$par_tv_ahead_sd
```
            The standard deviation of the forecasted time-varying parameters. Only when
            t_ahead > 0.

```
filter$par_tv_ahead_quant
```
            The quantiles of the forecasted time-varying parameters. Only when t_ahead >
            0.

```
filter$score_tv_ahead_mean
```
            The mean of the forecasted scores. Only when t_ahead > 0.

```
filter$score_tv_ahead_sd
```
            The standard deviation of the forecasted scores. Only when t_ahead > 0.

```
filter$score_tv_ahead_quant
```
            The quantiles of the forecasted scores. Only when t_ahead > 0.

Note

Supported generic functions for S3 class gas_filter include summary() ans plot().

References

Blasques, F., Koopman, S. J., Łasak, K., and Lucas, A. (2016). In-Sample Confidence Bands and Out-of-Sample Forecast Bands for Time-Varying Parameters in Observation-Driven Models. *International Journal of Forecasting*, **32**(3), 875–887. doi:10.1016/j.ijforecast.2015.11.018.

Creal, D., Koopman, S. J., and Lucas, A. (2013). Generalized Autoregressive Score Models with Applications. *Journal of Applied Econometrics*, **28**(5), 777–795. doi:10.1002/jae.1279.

Harvey, A. C. (2013). *Dynamic Models for Volatility and Heavy Tails: With Applications to Financial and Economic Time Series*. Cambridge University Press. doi:10.1017/cbo9781139540933.

See Also

gas()

Examples

```
# Load the Daily Toilet Paper Sales dataset
data("toilet_paper_sales")
y <- toilet_paper_sales$quantity
x <- as.matrix(toilet_paper_sales[3:9])

# Estimate GAS model based on the negative binomial distribution
est_negbin <- gas(y = y, x = x, distr = "negbin", regress = "sep")
```

```
est_negbin

# Filter the time-varying parameters by the "simulated_coefs" method
flt_negbin <- gas_filter(est_negbin, rep_gen = 100)
flt_negbin

# Plot the time-varying parameters with confidence bands
plot(flt_negbin)
```

---

gas_forecast   *Forecast GAS Model*

Description

A function for forecasting of generalized autoregressive score (GAS) models of Creal et al. (2013) and Harvey (2013). Method "mean_path" filters time-varying parameters based on zero score and then generates mean of time series. Method "simulated_paths" repeatedly simulates time series, simultaneously filters time-varying parameters, and then estimates mean, standard deviation, and quantiles (see Blasques et al., 2016). Instead of supplying arguments about the model, the function can be applied to the gas object obtained by the gas() function.

Usage

```
gas_forecast(gas_object = NULL, method = "mean_path", t_ahead = 1L,
  x_ahead = NULL, rep_ahead = 1000L, quant = c(0.025, 0.975), y = NULL,
  x = NULL, distr = NULL, param = NULL, scaling = "unit",
  regress = "joint", p = 1L, q = 1L, par_static = NULL,
  par_link = NULL, par_init = NULL, coef_est = NULL)
```

Arguments

gas_object	An optional GAS estimate, i.e. a list of S3 class gas returned by function gas().
method	A method used for forecasting. Supported methods are "mean_path" and "simulated_paths".
t_ahead	A number of observations to forecast.
x_ahead	Out-of-sample exogenous variables. For a single variable common for all time-varying parameters, a numeric vector. For multiple variables common for all time-varying parameters, a numeric matrix with observations in rows. For individual variables for each time-varying parameter, a list of numeric vectors or matrices in the above form. The number of observation must be equal to t_ahead.
rep_ahead	A number of simulation repetitions for method = "simulated_paths".
quant	A numeric vector of probabilities determining quantiles for method = "simulated_paths".
y	A time series. For univariate time series, a numeric vector or a matrix with a single column. For multivariate times series, a numeric matrix with observations in rows.

x   Optional exogenous variables. For a single variable common for all time-varying parameters, a numeric vector. For multiple variables common for all time-varying parameters, a numeric matrix with observations in rows. For individual variables for each time-varying parameter, a list of numeric vectors or matrices in the above form. The number of observation must be equal to the number of observations of y.

distr   A conditional distribution. See distr() for available distributions.

param   A parametrization of the conditional distribution. If NULL, default parametrization is used. See distr() for available parametrizations.

scaling   A scaling function for the score. The supported scaling functions are the unit scaling (scaling = "unit"), the inverse of the Fisher information matrix scaling (scaling = "fisher_inv"), and the inverse square root of the Fisher information matrix scaling (scaling = "fisher_inv_sqrt"). The latter two scalings use the Fisher information for the time-varying parameters only. For the full Fisher information matrix for both time-varying and static parameters, there are the "full_fisher_inv" and "full_fisher_inv_sqrt" scalings. For the individual Fisher information for each parameter, there are the "diag_fisher_inv" and "diag_fisher_inv_sqrt" scalings. Note that when the parametrization is orthogonal (see distr()), there are no differences between these scaling variants.

regress   A specification of the regression and dynamic equation with regard to exogenous variables. The supported specifications are exogenous variables and dynamics within the same equation (regress = "joint") and separate equations for exogenous variables and dynamics in the fashion of regression models with dynamic errors (regress = "sep"). In a stationary model without exogenous variables, the two specifications are equivalent, although with differently parametrized intercept.

p   A score order. For order common for all parameters, a numeric vector of length 1. For individual order for each parameter, a numeric vector of length equal to the number of parameters. Defaults to 1L.

q   An autoregressive order. For order common for all parameters, a numeric vector of length 1. For individual order for each parameter, a numeric vector of length equal to the number of parameters. Defaults to 1L.

par_static   An optional logical vector indicating static parameters. Overrides x, p, and q.

par_link   An optional logical vector indicating whether the logarithmic/logistic link should be applied to restricted parameters in order to obtain unrestricted values. Defaults to applying the logarithmic/logistic link for time-varying parameters and keeping the original link for constant parameters.

par_init   An optional numeric vector of initial values of time-varying parameters. For NA values or when NULL, set initial values to unconditional values of time-varying parameters. For example, in the case of GAS(1,1) model with regress = "joint", to omega / (1 - phi1). Not to be confused with starting values for the optimization coef_start.

coef_est   A numeric vector of estimated coefficients.

Value

A `list` of S3 class `gas_forecast` with components:

`data$y`	The time series.
`data$x`	The exogenous variables.
`data$x_ahead`	The out-of-sample exogenous variables.
`model$distr`	The conditional distribution.
`model$param`	The parametrization of the conditional distribution.
`model$scaling`	The scaling function.
`model$regress`	The specification of the regression and dynamic equation.
`model$t`	The length of the time series.
`model$t_ahead`	The length of the out-of-sample time series.
`model$n`	The dimension of the model.
`model$m`	The number of exogenous variables.
`model$p`	The score order.
`model$q`	The autoregressive order.
`model$par_static`	
	The static parameters.
`model$par_link`	The parameters with the logarithmic/logistic links.
`model$par_init`	The initial values of the time-varying parameters.
`model$coef_est`	The estimated coefficients.
`forecast$method`	
	The method used for forecasting.
`forecast$y_ahead_mean`	
	The mean of the forecasted time series.
`forecast$y_ahead_sd`	
	The standard deviation of the forecasted time series. Only for `method = "simulated_paths"`.
`forecast$y_ahead_quant`	
	The quantiles of the forecasted time series. Only for `method = "simulated_paths"`.
`forecast$par_tv_ahead_mean`	
	The mean of the forecasted time-varying parameters.
`forecast$par_tv_ahead_sd`	
	The standard deviation of the forecasted time-varying parameters. Only for `method = "simulated_paths"`.
`forecast$par_tv_ahead_quant`	
	The quantiles of the forecasted time-varying parameters. Only for `method = "simulated_paths"`.
`forecast$score_tv_ahead_mean`	
	The mean of the forecasted scores.
`forecast$score_tv_ahead_sd`	
	The standard deviation of the forecasted scores. Only for `method = "simulated_paths"`.
`forecast$score_tv_ahead_quant`	
	The quantiles of the forecasted scores. Only for `method = "simulated_paths"`.

Note

Supported generic functions for S3 class gas_forecast include summary() ans plot().

References

Blasques, F., Koopman, S. J., Łasak, K., and Lucas, A. (2016). In-Sample Confidence Bands and Out-of-Sample Forecast Bands for Time-Varying Parameters in Observation-Driven Models. *International Journal of Forecasting*, **32**(3), 875–887. doi:10.1016/j.ijforecast.2015.11.018.

Creal, D., Koopman, S. J., and Lucas, A. (2013). Generalized Autoregressive Score Models with Applications. *Journal of Applied Econometrics*, **28**(5), 777–795. doi:10.1002/jae.1279.

Harvey, A. C. (2013). *Dynamic Models for Volatility and Heavy Tails: With Applications to Financial and Economic Time Series*. Cambridge University Press. doi:10.1017/cbo9781139540933.

See Also

gas()

Examples

```
# Load the Daily Toilet Paper Sales dataset
data("toilet_paper_sales")
y <- toilet_paper_sales$quantity
x <- as.matrix(toilet_paper_sales[3:9])

# Estimate GAS model based on the negative binomial distribution
est_negbin <- gas(y = y, x = x, distr = "negbin", regress = "sep")
est_negbin

# Forecast the model by the "mean_paths" method
x_ahead <- cbind(kronecker(matrix(1, 53, 1), diag(7)), 1)[3:367, -1]
fcst_negbin <- gas_forecast(est_negbin, t_ahead = 365, x_ahead = x_ahead)
fcst_negbin

# Plot the forecasted expected value
plot(fcst_negbin)
```

---

gas_simulate   *Simulate GAS Model*

Description

A function for simulation of generalized autoregressive score (GAS) models of Creal et al. (2013) and Harvey (2013). Instead of supplying arguments about the model, the function can be applied to the gas object obtained by the gas() function.

Usage

```
gas_simulate(gas_object = NULL, t_sim = 1L, x_sim = NULL, distr = NULL,
  param = NULL, scaling = "unit", regress = "joint", n = NULL,
  p = 1L, q = 1L, par_static = NULL, par_link = NULL,
  par_init = NULL, coef_est = NULL)
```

Arguments

gas_object	An optional GAS estimate, i.e. a list of S3 class gas returned by function [gas()](#).
t_sim	A number of observations to simulate.
x_sim	Exogenous variables used for simulations. For a single variable common for all time-varying parameters, a numeric vector. For multiple variables common for all time-varying parameters, a numeric matrix with observations in rows. For individual variables for each time-varying parameter, a list of numeric vectors or matrices in the above form. The number of observation must be equal to t_sim.
distr	A conditional distribution. See [distr()](#) for available distributions.
param	A parametrization of the conditional distribution. If NULL, default parametrization is used. See [distr()](#) for available parametrizations.
scaling	A scaling function for the score. The supported scaling functions are the unit scaling (scaling = "unit"), the inverse of the Fisher information matrix scaling (scaling = "fisher_inv"), and the inverse square root of the Fisher information matrix scaling (scaling = "fisher_inv_sqrt"). The latter two scalings use the Fisher information for the time-varying parameters only. For the full Fisher information matrix for both time-varying and static parameters, there are the "full_fisher_inv" and "full_fisher_inv_sqrt" scalings. For the individual Fisher information for each parameter, there are the "diag_fisher_inv" and "diag_fisher_inv_sqrt" scalings. Note that when the parametrization is orthogonal (see [distr()](#)), there are no differences between these scaling variants.
regress	A specification of the regression and dynamic equation with regard to exogenous variables. The supported specifications are exogenous variables and dynamics within the same equation (regress = "joint") and separate equations for exogenous variables and dynamics in the fashion of regression models with dynamic errors (regress = "sep"). In a stationary model without exogenous variables, the two specifications are equivalent, although with differently parametrized intercept.
n	A dimension of the model. Required only for multivariate models.
p	A score order. For order common for all parameters, a numeric vector of length 1. For individual order for each parameter, a numeric vector of length equal to the number of parameters. Defaults to 1L.
q	An autoregressive order. For order common for all parameters, a numeric vector of length 1. For individual order for each parameter, a numeric vector of length equal to the number of parameters. Defaults to 1L.
par_static	An optional logical vector indicating static parameters. Overrides x, p, and q.

par_link   An optional logical vector indicating whether the logarithmic/logistic link should
                  be applied to restricted parameters in order to obtain unrestricted values. De-
                  faults to applying the logarithmic/logistic link for time-varying parameters and
                  keeping the original link for constant parameters.

par_init   An optional numeric vector of initial values of time-varying parameters. For NA
                  values or when NULL, set initial values to unconditional values of time-varying
                  parameters. For example, in the case of GAS(1,1) model with regress = "joint",
                  to omega / (1 - phi1). Not to be confused with starting values for the optimiza-
                  tion coef_start.

coef_est   A numeric vector of estimated coefficients.

Value

A list of S3 class gas_simulate with components:

data$x_sim   The exogenous variables used in simulation.

model$distr   The conditional distribution.

model$param   The parametrization of the conditional distribution.

model$scaling   The scaling function.

model$regress   The specification of the regression and dynamic equation.

model$t_sim   The length of the simulated time series.

model$n   The dimension of the model.

model$m   The number of exogenous variables.

model$p   The score order.

model$q   The autoregressive order.

model$par_static

                       The static parameters.

model$par_link   The parameters with the logarithmic/logistic links.

model$par_init   The initial values of the time-varying parameters.

model$coef_est   The estimated coefficients.

simulation$y_sim

                       The simulated time series.

simulation$par_tv_sim

                       The simulated time-varying parameters.

simulation$score_tv_sim

                       The simulated scores.

Note

Supported generic functions for S3 class gas_simulate include [summary()]() ans [plot()]().

References

Creal, D., Koopman, S. J., and Lucas, A. (2013). Generalized Autoregressive Score Models with
Applications. *Journal of Applied Econometrics*, **28**(5), 777–795. doi:10.1002/jae.1279.

Harvey, A. C. (2013). *Dynamic Models for Volatility and Heavy Tails: With Applications to Finan-
cial and Economic Time Series*. Cambridge University Press. doi:10.1017/cbo9781139540933.

See Also

[gas()](gas())

Examples

```
# Simulate GAS model based on the negative binomial distribution
sim_negbin <- gas_simulate(t_sim = 50, distr = "negbin", reg = "sep",
  coef_est = c(2.60, 0.02, 0.95, 0.03))
sim_negbin

# Plot the simulated time series
plot(sim_negbin)
```

---

ice_hockey_championships

Results of the Ice Hockey World Championships

Description

The dataset contains the results of the annual men's Ice Hockey World Championships from 1998 to 2023. In 1998, the International Ice Hockey Federation set the number of teams participating in the championships at 16. Since 1998, a total of 24 teams have qualified for the championship division. This dataset is analyzed in Holý and Zouhar (2022).

Usage

```
ice_hockey_championships
```

Format

A list with components:

rankings A matrix of final rankings. Rows correspond to years, columns to teams. Value Inf means that the team did not advance to the championship. Value NA means that the championship did not take place.

hosts A matrix of dummy variables indicating whether the team hosted the championship. Rows correspond to years, columns to teams. Multiple hosts of one championship is possible. Value NA means that the championship did not take place.

Source

International Ice Hockey Federation (www.iihf.com).

References

Holý, V. and Zouhar, J. (2022). Modelling Time-Varying Rankings with Autoregressive and Score-Driven Dynamics. Journal of the Royal Statistical Society: Series C (Applied Statistics), **71**(5). doi:10.1111/rssc.12584.

`toilet_paper_sales` *Daily Toilet Paper Sales*

Description

The daily number of toilet paper packs sold in a European store in 2001 and 2002. The `promo` variable indicates whether the product was promoted in a campaign. Missing values correspond to the days when the store was closed.

Usage

```
toilet_paper_sales
```

Format

A data frame with columns:

date Date.

monday Dummy variable indicating whether it is Monday.

tuesday Dummy variable indicating whether it is Tuesday.

wednesday Dummy variable indicating whether it is Wednesday.

thursday Dummy variable indicating whether it is Thursday.

friday Dummy variable indicating whether it is Friday.

saturday Dummy variable indicating whether it is Saturday.

sunday Dummy variable indicating whether it is Sunday.

promo Dummy variable indicating whether the product is promoted.

quantity Number of packs sold.

`wrappers_hessian` *Wrappers for Hessian Functions*

Description

Wrappers of common R Hessian functions. Their purpose is to be passed as the `hessian_function` argument in the [`gas()`](#) function.

Usage

```
wrapper_hessian_stats(obj_fun, theta_optim, est_details, ...)

wrapper_hessian_pracma(obj_fun, theta_optim, est_details, ...)

wrapper_hessian_numderiv(obj_fun, theta_optim, est_details, ...)
```

Arguments

`obj_fun`	An objective function.
`theta_optim`	A numeric vector of the optimal values of the variables.
`est_details`	A list of variables used for estimation.
`...`	Additional arguments to be passed to the Hessian function.

Value

A list with components:

`status_hessian`	The status of the Hessian computation.
`theta_hessian`	The Hessian matrix.

Functions

- `wrapper_hessian_stats()`: Wrapper for Hessian function `stats::optimHess()`.

- `wrapper_hessian_pracma()`: Wrapper for Hessian function `pracma::hessian()`.

- `wrapper_hessian_numderiv()`: Wrapper for Hessian function `numDeriv::hessian()`.

See Also

`gas()` `wrappers_optim` `wrappers_parallel`

---

wrappers_optim	*Wrappers for Optimization Functions*

Description

Wrappers of common R optimization functions. Their purpose is to be passed as the `optim_function` argument in the `gas()` function.

Usage

```
wrapper_optim_stats(obj_fun, theta_start, theta_bound_lower, theta_bound_upper,
  est_details, ...)

wrapper_optim_nloptr(obj_fun, theta_start, theta_bound_lower,
  theta_bound_upper, est_details, ...)
```

Arguments

`obj_fun`	An objective function.
`theta_start`	A numeric vector of starting values of the variables.
`theta_bound_lower`	
	A numeric vector of lower bounds on the variables.
`theta_bound_upper`	
	A numeric vector of upper bounds on the variables.
`est_details`	A list of variables used for estimation.
`...`	Additional arguments to be passed to the optimization function.

Value

A list with components:

`status_optim`	The status of the optimization computation.
`theta_optim`	The optimal solution.

Functions

- `wrapper_optim_stats()`: Wrapper for optimization function `stats::optim()`.

- `wrapper_optim_nloptr()`: Wrapper for optimization function `nloptr::nloptr()`.

See Also

`gas()` `wrappers_hessian` `wrappers_parallel`

---

wrappers_parallel	*Wrappers for Parallelization Functions*

Description

Wrappers of common R parallelization functions. Their purpose is to be passed as the `parallel_function` argument in the `gas_bootstrap()` function.

Usage

```
wrapper_parallel_none(run_num, run_fun, run_details, ...)

wrapper_parallel_multicore(run_num, run_fun, run_details, ...)

wrapper_parallel_snow(run_num, run_fun, run_details, ...)
```

Arguments

run_num	A number of iterations.
run_fun	A function to be computed.
run_details	A list of variables used for computation.
...	Additional arguments to be passed to the parallelization function.

Value

A list containing computed values.

Functions

- `wrapper_parallel_none()`: Wrapper for function `base::lapply()`.

- `wrapper_parallel_multicore()`: Wrapper for parallelization function `parallel::mclapply()`.

- `wrapper_parallel_snow()`: Wrapper for parallelization function `parallel::parLapply()`.

See Also

`gas_bootstrap()` `wrappers_optim` `wrappers_hessian`

Index