Package 'GCEstim'

June 7, 2026

Type Package

Title Regression Coefficients Estimation Using the Generalized Cross Entropy

Version 1.1.0

Date 2026-06-07

Description Estimation and inference using the Generalized Maximum Entropy (GME) and Generalized Cross Entropy (GCE) framework, a flexible method for solving ill-posed inverse problems and parameter estimation under uncertainty (Golan, Judge, and Miller (1996, ISBN:978-0471145925) ``Maximum Entropy Econometrics: Robust Estimation with Limited Data''). The package includes routines for generalized cross entropy estimation of linear models including the implementation of a GME-GCE two steps approach. Diagnostic tools, and options to incorporate prior information through support and prior distributions are available (Macedo, Cabral, Afreixo, Macedo and Angelelli (2025) <doi:10.1007/978-3-031-97589-9_21>). In particular, support spaces can be defined by the user or be internally computed based on the ridge trace or on the distribution of standardized regression coefficients. Different optimization methods for the objective function can be used. An adaptation of the normalized entropy aggregation (Macedo and Costa (2019) <doi:10.1007/978-3-030-26036-1_2> ``Normalized entropy aggregation for inhomogeneous large-scale data'') and a two-stage maximum entropy approach for time series regression (Macedo (2022) <doi:10.1080/03610918.2022.2057540>) are also available. Suitable for applications in econometrics, health, signal processing, and other fields requiring robust estimation under data constraints.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0), zoo

Imports downlit, data.table, rlang, lbfgs, lbfgsb3c, meboot, optimParallel, optimx, rstudioapi, stats, clusterGeneration, simstudy, pracma, pathviewr, Rsolnp, bayestestR, ggplot2, ggpubr, ggdist, latex2exp, plotly, viridis, hdrcde, shiny, miniUI, shinyWidgets, shinydashboardPlus, readxl, DT, magrittr

Suggests knitr, rmarkdown, kableExtra, NlcOptim

VignetteBuilder knitr

URL https://github.com/jorgevazcabral/GCEstim

BugReports https://github.com/jorgevazcabral/GCEstim/issues

Config/spelling wordlist: inst/WORDLIST

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Cabral Jorge [aut, cre] (ORCID:
<https://orcid.org/0000-0001-5721-4550>),
Macedo Pedro [ths],
Afreixo Vera [ths]

Maintainer Cabral Jorge <jorgecabral@ua.pt>

Repository CRAN

Date/Publication 2026-06-07 10:40:02 UTC

Contents

`accmeasure` *Accuracy measures*

Description

Function that allows to calculate different types of errors for point predictions:

1. MAE - Mean Absolute Error,
2. MAD - Mean Absolute Deviation,
3. MSE - Mean Squared Error,
4. RMSE - Root Mean Squared Error,
5. MAPE - Mean Absolute Percentage Error,
6. sMAPE - symmetric Mean Absolute Percentage Error,
7. MASE - Mean Absolute Scaled Error (Hyndman & Koehler, 2006)

Usage

```
accmeasure(
  y_pred,
  y_true,
  which = c("RMSE", "MSE", "MAPE", "sMAPE", "MAE", "MAD", "MASE")
)
```

Arguments

y_pred	fitted values.
y_true	observed values.
which	one of c("RMSE", "MAPE", "sMAPE", "MAE", "MAD", "MASE")

Value

The value of the chosen error is returned.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

References

Hyndman, R. J., & Koehler, A. B. (2006) *Another look at measures of forecast accuracy.* International Journal of Forecasting, 22(4), 679–688. doi:10.1016/j.ijforecast.2006.03.001

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)
```

```
accmeasure(fitted(res_gce_package), dataThesis$y, which = "MSE")
accmeasure(coef(res_gce_package), coef.dataThesis, which = "MSE")
```

`case.names.lmgce` *Case Names of* `lmgce` *Fitted Models*

Description

Simple utility returning case names.

Usage

```
## S3 method for class 'lmgce'
case.names(object, ...)
```

Arguments

`object`	Fitted `lmgce` model object.
`...`	Additional arguments (not used).

Value

A character vector containing the names or labels of the cases (observations) in the `lmgce` model object.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

case.names(res_gce_package)
```

changestep *Change the step from* lmgce *object*

Description

Changes the number of GCE reestimations of a lmgce object

Usage

```
changestep(object, twosteps.n, verbose = 0)
```

Arguments

object	fitted lmgce object.
twosteps.n	An integer that defines the number of GCE reestimations to be used.
verbose	An integer to control how verbose the output is. For a value of 0 no messages or output are shown and for a value of 3 all messages are shown. The default is verbose = 0.

Value

An lmgce object with the specified number of GCE reestimations

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

res_gce_package_change_step <- changestep(res_gce_package,0)

summary(res_gce_package)

summary(res_gce_package_change_step)
```

changesupport *Change the support from* lmgce *object*

Description

Changes the support spaces of a lmgce object

Usage

```
changesupport(object, support, verbose = 0)
```

Arguments

object	fitted lmgce object.
support	One of c("min", "1se", "elbow") or a chosen support from object$support.ok.
verbose	An integer to control how verbose the output is. For a value of 0 no messages or output are shown and for a value of 3 all messages are shown. The default is verbose = 0.

Value

An lmgce object with the specified support spaces

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

res_gce_package_change <- changesupport(res_gce_package, "min")

summary(res_gce_package)

summary(res_gce_package_change)
```

coef.cv.lmgce *Extract* cv.lmgce *Coefficients*

Description

Extract coefficients from a cv.lmgce object

Usage

```
## S3 method for class 'cv.lmgce'
coef(object, ...)
```

Arguments

object	Fitted cv.lmgce model object.
...	Additional arguments (not used).

Value

Returns the coefficients from a cv.lmgce object. The coefficients are obtained from the lmgce object with best performance. These coefficients are stored in object$best$coefficients.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

coef.cv.tsbootgce *Extract* cv.tsbootgce *Model Coefficients*

Description

Extract coefficients from a cv.tsbootgce object

Usage

```
## S3 method for class 'cv.tsbootgce'
coef(object, which = NULL, OLS = FALSE, seed = object$seed, ...)
```

Arguments

object	Fitted cv.tsbootgce model object.
which	The default is which = NULL and returns the coefficients defined in the argument coef.method from the cv.tsbootgce object. Can be set as "mode" or "median" and the mode and median coefficients will be computed, respectively (see hdr).
OLS	Boolean value. If TRUE, the confidence interval returned is based on OLS estimates. The default is OLS = FALSE.
seed	A single value, interpreted as an integer, for reproducibility or NULL for randomness. The default is seed = object$seed.
...	Additional arguments.

Value

Returns the coefficients from a cv.tsbootgce object

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

---

coef.dataThesis *Coefficients used in simulated data set generated with fngendata*

Description

Coefficients used in simulated data, used to demonstrate the functions of GCEstim.

Usage

```
coef.dataThesis
```

Format

A vector containing the coefficients used to generate dataThesis

Examples

```
coef.dataThesis
```

coef.lmgce *Extract* lmgce *Model Coefficients*

Description

Extract coefficients from a lmgce object

Usage

```
## S3 method for class 'lmgce'
coef(object, ...)
```

Arguments

object	Fitted lmgce model object.
...	Additional arguments (not used).

Value

Returns the coefficients from a lmgce object

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

coef(res_gce_package)
```

coef.neagging *Extract* neagging *Coefficients*

Description

Extract coefficients from a neagging object

Usage

```
## S3 method for class 'neagging'
coef(object, which = which.min(object$error)[[1]], ...)
```

Arguments

object	Fitted neagging model object.
which	Number of aggregated models. The coefficients returned are by default the ones that produced the lowest in sample error.
...	Additional arguments.

Value

Returns the coefficients from a neagging object

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

res_neagging <- neagging(res_gce_package)
coef(res_neagging)
coef(res_neagging, which = ncol(res_neagging$matrix))
```

---

coef.ridgetrace *Extract* ridgetrace *Model Coefficients*

Description

Extract coefficients from a ridgetrace object

Usage

```
## S3 method for class 'ridgetrace'
coef(object, which = "min.error", ...)
```

Arguments

object	Fitted ridgetrace model object.
which	One of c("min.error", "max.abs"). If which = "min.error", the default, the coefficients that produced the lowest error cross-validation error (cv = TRUE),or in sample error are returned (cv = FALSE). If which = "max.abs" then the maximum absolute coefficients are returned.
...	Additional arguments (not used).

Value

Returns the coefficients from a ridgetrace object

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res.ridgetrace <-
  ridgetrace(
    formula = y ~ X001 + X002 + X003 + X004,
    data = dataThesis)

coef(res.ridgetrace)
```

---

coef.tsbootgce *Extract* tsbootgce *Model Coefficients*

Description

Extract coefficients from a tsbootgce object

Usage

```
## S3 method for class 'tsbootgce'
coef(object, which = NULL, OLS = FALSE, seed = object$seed, ...)
```

Arguments

object	Fitted tsbootgce model object.
which	The default is which = NULL and returns the coefficients defined in the argument coef.method from the tsbootgce object. Can be set as "mode" or "median" and the mode and median coefficients will be computed, respectively (see hdr).
OLS	Boolean value. If TRUE, returns OLS estimates. The default is OLS = FALSE.
seed	A single value, interpreted as an integer, for reproducibility or NULL for randomness. The default is seed = object$seed.
...	Additional arguments.

Value

Returns the coefficients from a tsbootgce object

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res.tsbootgce <-
  tsbootgce(
    formula = CO2 ~ 1 + L(EPC, 1) + L(EUS, 2) + L(GDP, 0),
    data = moz_ts)

coef(res.tsbootgce)
```

---

coefficients.cv.lmgce *Extract* cv.lmgce *Coefficients*

Description

Extract coefficients from a cv.lmgce object

Usage

```
## S3 method for class 'cv.lmgce'
coefficients(object, ...)
```

Arguments

object	Fitted cv.lmgce model object.
...	Additional arguments (not used).

Value

Returns the coefficients from a cv.lmgce object. The coefficients are obtained from the lmgce object with best performance. These coefficients are stored in object$best$coefficients.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

coefficients.cv.tsbootgce

Extract cv.tsbootgce *Model Coefficients*

Description

Extract coefficients from a cv.tsbootgce object

Usage

```
## S3 method for class 'cv.tsbootgce'
coefficients(object, which = NULL, OLS = FALSE, seed = object$seed, ...)
```

Arguments

object	Fitted cv.tsbootgce model object.
which	The default is which = NULL and returns the coefficients defined in the argument coef.method from the cv.tsbootgce object. Can be set as "mode" or "median" and the mode and median coefficients will be computed, respectively (see hdr).
OLS	Boolean value. If TRUE, the confidence interval returned is based on OLS estimates. The default is OLS = FALSE.
seed	A single value, interpreted as an integer, for reproducibility or NULL for randomness. The default is seed = object$seed.
...	Additional arguments.

Value

Returns the coefficients from a cv.tsbootgce object

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

coefficients.lmgce *Extract* lmgce *Model Coefficients*

Description

Extract coefficients from a lmgce object

Usage

```
## S3 method for class 'lmgce'
coefficients(object, ...)
```

Arguments

object	Fitted lmgce model object.
...	Additional arguments (not used).

Value

Returns the coefficients from a lmgce object

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

coefficients(res_gce_package)
```

---

coefficients.neagging *Extract* neagging *Coefficients*

Description

Extract coefficients from a neagging object

Usage

```
## S3 method for class 'neagging'
coefficients(object, which = which.min(object$error)[[1]], ...)
```

Arguments

object	Fitted neagging model object.
which	Number of aggregated models. The coefficients returned are by default the ones that produced the lowest in sample error.
...	Additional arguments.

Value

Returns the coefficients from a neagging object

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

res_neagging <- neagging(res_gce_package)
coefficients(res_neagging)
coefficients(res_neagging, which = ncol(res_neagging$matrix))
```

`coefficients.ridgetrace`

Extract `ridgetrace` *Model Coefficients*

Description

Extract coefficients from a `ridgetrace` object

Usage

```
## S3 method for class 'ridgetrace'
coefficients(object, which = "min.error", ...)
```

Arguments

`object`	Fitted `ridgetrace` model object.
`which`	One of `c("min.error", "max.abs")`. If `which = "min.error"`, the default, the coefficients that produced the lowest error cross-validation error (`cv = TRUE`),or in sample error are returned (`cv = FALSE`). If `which = "max.abs"` then the maximum absolute coefficients are returned.
`...`	Additional arguments.

Value

Returns the coefficients from a `ridgetrace` object

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res.ridgetrace <-
  ridgetrace(
    formula = y ~ X001 + X002 + X003 + X004,
    data = dataThesis)

coefficients(res.ridgetrace)
```

`coefficients.tsbootgce` *Extract* `tsbootgce` *Model Coefficients*

Description

Extract coefficients from a `tsbootgce` object

Usage

```
## S3 method for class 'tsbootgce'
coefficients(object, which = NULL, OLS = FALSE, seed = object$seed, ...)
```

Arguments

`object`	Fitted `tsbootgce` model object.
`which`	The default is `which = NULL` and returns the coefficients defined in the argument `coef.method` from the `tsbootgce` object. Can be set as "mode" or "median" and the mode and median coefficients will be computed, respectively (see `hdr`).
`OLS`	Boolean value. If `TRUE`, returns OLS estimates. The default is `OLS = FALSE`.
`seed`	A single value, interpreted as an integer, for reproducibility or `NULL` for randomness. The default is `seed = object$seed`.
`...`	Additional arguments.

Value

Returns the coefficients from a `tsbootgce` object

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res.tsbootgce <-
  tsbootgce(
    formula = CO2 ~ 1 + L(EPC, 1) + L(EUS, 2) + L(GDP, 0),
    data = moz_ts)

coefficients(res.tsbootgce)
```

`confint.cv.tsbootgce` *Confidence Intervals for* `cv.tsbootgce` *Model Parameters and Normalized Entropy*

Description

Computes confidence intervals for one or more parameters or Normalized Entropy in a `cv.tsbootgce` fitted model.

Usage

```
## S3 method for class 'cv.tsbootgce'
confint(
  object,
  parm,
  level = 0.95,
  which = c("estimates", "NormEnt"),
  method = c("hdr", "percentile", "basic"),
  seed = object$seed,
  OLS = FALSE,
  ...
)
```

Arguments

`object`	Fitted `cv.tsbootgce` model object.
`parm`	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	the confidence level required. The default is `level = 0.95`.
`which`	One of `c("estimates", "NormEnt")`. The default is `which = "estimates"`.
`method`	method used to compute the interval. One of c("hdr", "percentile", "basic"). The default is `method = "hdr"` (see `hdr`).
`seed`	A single value, interpreted as an integer, for reproducibility or `NULL` for randomness. The default is `seed = object$seed`.

OLS	Boolean value. If TRUE, the confidence interval returned is based on OLS estimates. The default is OLS = FALSE.
...	additional arguments.

Value

A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. Generally, these will be labelled as (1-level)/2 and 1 - (1-level)/2 in percentage (by default 2.5 percent and 97.5 percent).

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

---

confint.lmgce *Confidence Intervals for* lmgce *Model Parameters and Normalized Entropy*

Description

Computes confidence intervals for one or more parameters or Normalized Entropy in a lmgce fitted model.

Usage

```
## S3 method for class 'lmgce'
confint(
  object,
  parm,
  level = 0.95,
  which = c("estimates", "NormEnt"),
  method = {
     if (which == "estimates") {
         c("z", "percentile", "basic")

    }
     else {
         c("percentile", "basic")
     }
 },
  boot.B = ifelse(object$boot.B == 0, 100, object$boot.B),
  boot.method = object$boot.method,
  ...
)
```

Arguments

object	Fitted lmgce model object.
parm	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
level	the confidence level required. The default is level = 0.95.
which	One of c("estimates", "NormEnt"). The default is which = "estimates".
method	method used to compute the interval. One of c("z","percentile", "basic"). The default is method = "z" and is only valid for the parameters.
boot.B	A single positive integer greater or equal to 10 for the number of bootstrap replicates for the computation of the bootstrap confidence interval(s), to be used when method = c("percentile", "basic") and when object was created with boot.B = 0. The default is boot.B = 100 when the object has no previous sampling information and boot.B = object$boot.B otherwise, which corresponds to the boot.B given to lmgce when the object was created.
boot.method	Method used for bootstrapping. One of c("residuals", "cases", "wild") which corresponds to resampling on residuals, on individual cases or on residuals multiplied by a N(0,1) variable, respectively. The default is boot.method = object$boot.method.
...	additional arguments.

Value

A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2 in percentage (by default 2.5 percent and 97.5 percent).

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)


confint(res_gce_package, method = "percentile")

confint(res_gce_package, which = "NormEnt", level = 0.99)

confint(res_gce_package, parm = c("X004"), level = 0.99)
```

confint.tsbootgce *Confidence Intervals for* tsbootgce *Model Parameters and Normalized Entropy*

Description

Computes confidence intervals for one or more parameters or Normalized Entropy in a tsbootgce fitted model.

Usage

```
## S3 method for class 'tsbootgce'
confint(
  object,
  parm,
  level = 0.95,
  which = c("estimates", "NormEnt"),
  method = c("hdr", "percentile", "basic"),
  seed = object$seed,
  OLS = FALSE,
  ...
)
```

Arguments

object	Fitted tsbootgce model object.
parm	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
level	the confidence level required. The default is level = 0.95.
which	One of c("estimates", "NormEnt"). The default is which = "estimates".
method	method used to compute the interval. One of c("hdr", "percentile", "basic"). The default is method = "hdr" (see hdr).
seed	A single value, interpreted as an integer, for reproducibility or NULL for randomness. The default is seed = object$seed.
OLS	Boolean value. If TRUE, the confidence interval returned is based on OLS estimates. The default is OLS = FALSE.
...	additional arguments.

Value

A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. Generally, these will be labelled as (1-level)/2 and 1 - (1-level)/2 in percentage (by default 2.5 percent and 97.5 percent).

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res.tsbootgce <-
  tsbootgce(
    formula = CO2 ~ 1 + L(EPC, 1) + L(EUS, 2) + L(GDP, 0),
    data = moz_ts)

confint(res.tsbootgce, method = "percentile")

confint(res.tsbootgce, which = "NormEnt", level = 0.99)

confint(res.tsbootgce, parm = c("L(GDP, 0)"), level = 0.99)
```

---

cv.dynlmgce *Cross-validation for* dynlmgce

Description

Performs k-fold cross-validation for some of the dynlmgce parameters.

Usage

```
cv.dynlmgce(
  formula,
  data,
  subset,
  na.action,
  offset,
  contrasts = NULL,
  start = NULL,
  end = NULL,
  cv = TRUE,
  cv.nfolds = 5,
  errormeasure = c("RMSE", "MSE", "MAE", "MAPE", "sMAPE", "MASE"),
  errormeasure.which = {
     if (isTRUE(cv))
         c("1se", "min", "elbow")

    else c("min", "elbow")
 },
  support.method = c("standardized", "ridge"),
  support.method.ridge.lambda = NULL,
  support.method.ridge.lambda.min = 10^-3,
```

```
  support.method.ridge.lambda.max = 10^3,
  support.method.ridge.lambda.n = 100,
  support.method.ridge.standardize = TRUE,
  support.method.ridge.penalize.intercept = TRUE,
  support.method.ridge.symm = TRUE,
  support.method.ridge.maxresid = TRUE,
  support.signal = NULL,
  support.signal.vector = NULL,
  support.signal.vector.min = 0.3,
  support.signal.vector.max = 20,
  support.signal.vector.n = 20,
  support.signal.points = c(3, 5, 7, 9),
  support.noise = NULL,
  support.noise.points = c(3, 5, 7, 9),
  weight = c(0.1, 0.3, 0.5, 0.7, 0.9),
  twosteps.n = 1,
 method = c("dual.BFGS", "dual.lbfgsb3c", "primal.solnl", "primal.solnp", "dual.CG",
   "dual.L-BFGS-B", "dual.Rcgmin", "dual.bobyqa", "dual.newuoa", "dual.nlminb",
    "dual.nlm", "dual.lbfgs", "dual.optimParallel"),
  caseGLM = c("D", "M", "NM"),
  boot.B = 0,
  boot.method = c("residuals", "cases", "wild"),
  seed = 230676,
  OLS = TRUE,
  verbose = 0,
  coef = NULL
)
```

Arguments

formula	a "formula" describing the linear model to be fit. For details see lm and dynlm.
data	A data.frame (or object coercible by as.data.frame to a data frame) or time series object (e.g., ts or zoo), containing the variables in the model.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
na.action	a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
offset	this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector or matrix of extents matching those of the response. One or more offset terms can be included in the formula instead or as well, and if more than one are specified their sum is used. See model.offset.
contrasts	An optional list. See the contrasts.arg of model.matrix.default.
start	The time of the first observation. Either a single number or a vector of two numbers (the second of which is an integer), which specify a natural time unit and a (1-based) number of samples into the time unit (see ts).

`end` The time of the last observation, specified in the same way as `start` (see `ts`).

`cv` Boolean value. If `TRUE` the error, `errormeasure`, will be computed using cross-validation. If `FALSE` the error will be computed in sample. The default is `cv = TRUE`.

`cv.nfolds` number of folds used for cross-validation when `cv = TRUE`. The default is `cv.nfolds = 5` and the smallest value allowable is `cv.nfolds = 3`.

`errormeasure` Loss function (error) to be used for the selection of the support spaces. One of c("RMSE","MSE", "MAE", "MAPE", "sMAPE", "MASE"). The default is `errormeasure = "RMSE"`.

`errormeasure.which`
Which value of `errormeasure` to be used for selecting a support space upper limit from `support.signal.vector`. One of `c("min", "1se", "elbow")` where `"min"` corresponds to the support spaces that produced the lowest error, `"1se"` corresponds to the support spaces such that error is within 1 standard error of the CV error for `"min"` and `"elbow"` corresponds to the elbow point of the error curve (the point that maximizes the distance between each observation, i.e, the pair composed by the upper limit of the support space and the error, and the line between the first and last observations, i.e., the lowest and the highest upper limits of the support space respectively. See `find_curve_elbow`). The default is `errormeasure.which = "1se"`.

`support.method` One of c("standardized", "ridge"). If `support.method = "standardized}, the default, standardized coefficients are used to define the signal support spaces. If \code{support.method = "ridge` the signal support spaces are define by the ridge trace.

`support.method.ridge.lambda`
Ridge parameter. The default is `support.method.ridge.lambda = NULL` and a lambda logarithmic sequence will be computed based on `support.method.ridge.lambda.n`, `support.method.ridge.lambda.min` and `support.method.ridge.lambda.max`. Supplying a lambda sequence overrides this. To be used when `support.method = "ridge"`.

`support.method.ridge.lambda.min`
Minimum value for the `support.method.ridge.lambda` sequence. The default is `support.method.ridge.lambda.min = 10^-3`. To be used when `support.method = "ridge"` and `support.method.ridge.lambda = NULL`.

`support.method.ridge.lambda.max`
Maximum value for the `support.method.ridge.lambda` sequence. The default is `support.method.ridge.lambda.max = 10^3`. To be used when `support.method = "ridge"` and `support.method.ridge.lambda = NULL`.

`support.method.ridge.lambda.n`
The number of ridge parameters values. The default is `support.method.ridge.lambda.n = 100`. To be used when `support.method = "ridge"` and `support.method.ridge.lambda = NULL`.

`support.method.ridge.standardize`
Boolean value. If `TRUE`, the default, then: i) centering is done by subtracting the column means of x and y from their corresponding columns; ii) scaling is done by dividing the (centered) columns of x and y by their standard deviations. To be used when `support.method = "ridge"`.

support.method.ridge.penalize.intercept
: Boolean value. if TRUE, the default, the intercept will be penalized. To be used when support.method = "ridge" and support.method.ridge.standardize = FALSE.

support.method.ridge.symm
: Boolean value. If TRUE, the default, signal supports will be symmetrical and the upper limit will be the maximum absolute values of the estimated ridge coefficients for support.method.ridge.lambda. If FALSE, the lower and upper limits will be, respectively, the minimum and maximum values of the estimated ridge coefficients.

support.method.ridge.maxresid
: Boolean value. if TRUE, the default, noise supports will symmetrical and the upper limit will be the maximum absolute value of the residuals of ridge estimation for support.method.ridge.lambda. If FALSE limits are computed using the empirical three-sigma rule (Pukelsheim (1994)).

support.signal
: NULL or fixed positive upper limit (L) for the support spaces (-L,L) on standardized data (when support.method = "standardized"); NULL or fixed positive factor to be multiplied by the maximum absolute value of the ridge trace for each coefficient (when support.method = "ridge"); a pair (LL,UL) or a matrix ((k+1) x 2) for the support spaces on original data. The default is support.signal = NULL.

support.signal.vector
: NULL or a vector of positive values when support.signal = NULL. If support.signal.vector = NULL, the default, a vector c(support.signal.vector.min,...,support.signal.vector.max) of dimension support.signal.vector.n and logarithmically equally spaced will be generated. Each value represents the upper limits for the standardized support spaces, when support.method = "standardized" or the factor to be multiplied by the maximum absolute value of the ridge trace for each coefficient, when support.method = "ridge".

support.signal.vector.min
: A positive value for the lowest limit of the support.signal.vector when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.min = 0.3.

support.signal.vector.max
: A positive value for the highest limit of the support.signal.vector when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.max = 20.

support.signal.vector.n
: A positive integer for the number of support spaces to be used when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.n = 20.

support.signal.points
: A vector of positive integers defining the number of points for the signal support to be tested .The default is support.signal.points = c(3, 5, 7, 9).

support.noise
: An interval, preferably centered around zero, given in the form c(LL,UL). If support.noise = NULL, the default, then a vector c(-L,L) is computed using the empirical three-sigma rule Pukelsheim (1994).

`support.noise.points`
: A vector of positive integers defining the number of points for the noise support to be tested. The default is `support.noise.points = c(3, 5, 7, 9)`.

`weight`
: a vector of values between zero and one representing the prediction-precision loss trade-off. The default is `weight = c(0.1, 0.3, 0.5, 0.7, 0.9)`.

`twosteps.n`
: Number of GCE reestimations using a previously estimated vector of signal probabilities.

`method`
: Use `"primal.solnl"` (GCE using Sequential Quadratic Programming (SQP) method; see `solnl`) or `"primal.solnp"` (GCE using the augmented Lagrange multiplier method with an SQP interior algorithm; see `solnp`) for primal form of the optimization problem and `"dual.CG"` (GCE using a conjugate gradients method; see `optim`), `"dual.BFGS"` (GCE using Broyden-Fletcher-Goldfarb-Shanno quasi-Newton method; see `optim`), `"dual.L-BFGS-B"` (GCE using a box-constrained optimization with limited-memory modification of the BFGS quasi-Newton method; see `optim`), `dual.Rcgmin` (GCE using an update of the conjugate gradient algorithm; see `optimx`), `dual.bobyqa` (GCE using a derivative-free optimization by quadratic approximation; see `optimx` and `bobyqa`), `dual.newuoa` (GCE using a derivative-free optimization by quadratic approximation; see `optimx` and `newuoa`), `dual.nlminb` (GCE; see `optimx` and `nlminb`), `dual.nlm` (GCE; see `optimx` and `nlm`), `dual.lbfgs` (GCE using the Limited-memory Broyden-Fletcher-Goldfarb-Shanno; see `lbfgs`), `dual.lbfgsb3c` (GCE using L-BFSC-B implemented in Fortran code and with an Rcpp interface; see `lbfgsb3c`) or `dual.optimParallel` (GCE using parallel version of the L-BFGS-B; see `optimParallel`) for dual form. The default is `method = "dual.BFGS"`.

`caseGLM`
: special cases of the generic general linear model. One of `c("D", "M", "NM")`, where "D" stands for data, "M" for moment and "NM" for normed-moment The default is `caseGLM = "D"`.

`boot.B`
: A single positive integer greater or equal to 10 for the number of bootstrap replicates to be used for the computation of the bootstrap confidence interval(s). Zero value will generate no replicate. The default is `boot.B = 0`.

`boot.method`
: Method to be use for bootstrapping. One of `c("residuals", "cases", "wild")` which corresponds to resampling on residuals, on individual cases or on residuals multiplied by a N(0,1) variable, respectively. The default is `boot.method = "residuals"`.

`seed`
: A single value, interpreted as an integer, for reproducibility or `NULL` for randomness. The default is `seed = 230676`.

`OLS`
: Boolean value. if `TRUE`, the default, OLS estimation is performed.

`verbose`
: An integer to control how verbose the output is. For a value of 0 no messages or output are shown and for a value of 3 all messages are shown. The default is `verbose = 0`.

`coef`
: A vector of the true coefficients, when available.

Details

The `cv.dynlmgce` function fits several dynamic linear regression models via generalized cross according to the defined arguments. In particular, `support.signal.points`, `support.noise.points` and `weight` can be defined as vectors.

Value

`cv.dynlmgce` returns an object of `class` `cv.lmgce` containing at least the following components:

`results` a $C \times 8$ `data.frame`, where C is the number of combinations of the arguments `support.signal.points`, `support.noise.points` and `weight`. Contains information about the arguments, error, convergence of the optimization method and time of computation.

`best` a `lmgce` object obtained with the combination of arguments that produced the lowest cross-validation error.

`support.signal.points`
a vector of the `support.signal.points` tested.

`support.signal.points.best`
the value of `support.signal.points` that produced the lowest cross-validation error.

`support.noise.points`
a vector of the `support.noise.points` tested.

`support.noise.points.best`
the value of `support.noise.points` that produced the lowest cross-validation error.

`weight` a vector of the `weight` tested.

`weight.best` the value of `weight` that produced the lowest cross-validation error.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

References

Golan, A., Judge, G. G. and Miller, D. (1996) *Maximum entropy econometrics : robust estimation with limited data.* Wiley.

Golan, A. (2008). *Information and Entropy Econometrics — A Review and Synthesis.* Foundations and Trends® in Econometrics, 2(1–2), 1–145. doi:10.1561/0800000004

Golan, A. (2017) *Foundations of Info-Metrics: Modeling, Inference, and Imperfect Information (Vol. 1).* Oxford University Press. doi:10.1093/oso/9780199349524.001.0001

Pukelsheim, F. (1994) *The Three Sigma Rule.* The American Statistician, 48(2), 88–91. doi:10.2307/2684253

See Also

See the generic functions `plot.cv.lmgce`, `print.cv.lmgce` and `coef.cv.lmgce`.

Examples

```
res.cv.dynlmgce <-
  cv.dynlmgce(
    formula = CO2 ~ 1 + L(EPC, 1) + L(EUS, 2) + L(GDP, 0),
    data = moz_ts)
```

```
res.cv.dynlmgce
```

`cv.lmgce` *Cross-validation for* `lmgce`

Description

Performs k-fold cross-validation for some of the `lmgce` parameters.

Usage

```
cv.lmgce(
  formula,
  data,
  subset,
  na.action,
  offset,
  contrasts = NULL,
  model = TRUE,
  x = FALSE,
  y = FALSE,
  cv = TRUE,
  cv.nfolds = 5,
  errormeasure = c("RMSE", "MSE", "MAE", "MAPE", "sMAPE", "MASE"),
  errormeasure.which = {
     if (isTRUE(cv))
         c("1se", "min", "elbow")

    else c("min", "elbow")
 },
  support.method = c("standardized", "ridge"),
  support.method.ridge.lambda = NULL,
  support.method.ridge.lambda.min = 10^-3,
  support.method.ridge.lambda.max = 10^3,
  support.method.ridge.lambda.n = 100,
  support.method.ridge.standardize = TRUE,
  support.method.ridge.penalize.intercept = TRUE,
  support.method.ridge.symm = TRUE,
  support.method.ridge.maxresid = TRUE,
  support.signal = NULL,
  support.signal.vector = NULL,
  support.signal.vector.min = 0.3,
  support.signal.vector.max = 20,
  support.signal.vector.n = 20,
  support.signal.points = c(3, 5, 7, 9),
```

```
  support.noise = NULL,
  support.noise.points = c(3, 5, 7, 9),
  weight = c(0.1, 0.3, 0.5, 0.7, 0.9),
  twosteps.n = 1,
 method = c("dual.BFGS", "dual.lbfgsb3c", "primal.solnl", "primal.solnp", "dual.CG",
   "dual.L-BFGS-B", "dual.Rcgmin", "dual.bobyqa", "dual.newuoa", "dual.nlminb",
    "dual.nlm", "dual.lbfgs", "dual.optimParallel"),
  caseGLM = c("D", "M", "NM"),
  boot.B = 0,
  boot.method = c("residuals", "cases", "wild"),
  seed = 230676,
  OLS = TRUE,
  verbose = 0,
  coef = NULL
)
```

Arguments

formula	An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted.
data	A data frame (or object coercible by as.data.frame to a data frame) containing the variables in the model.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
na.action	a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
offset	this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector or matrix of extents matching those of the response. One or more offset terms can be included in the formula instead or as well, and if more than one are specified their sum is used. See model.offset.
contrasts	An optional list. See the contrasts.arg of model.matrix.default.
model	Boolean value. if TRUE, the model frame used is returned. The default is model = TRUE.
x	Boolean value. if TRUE, the model matrix used is returned. The default is x = FALSE.
y	Boolean value. if TRUE, the response used is returned. The default is y = FALSE.
cv	Boolean value. If TRUE the error, errormeasure, will be computed using cross-validation. If FALSE the error will be computed in sample. The default is cv = TRUE.
cv.nfolds	number of folds used for cross-validation when cv = TRUE. The default is cv.nfolds = 5 and the smallest value allowable is cv.nfolds = 3.

errormeasure Loss function (error) to be used for the selection of the support spaces. One of c("RMSE","MSE", "MAE", "MAPE", "sMAPE", "MASE"). The default is errormeasure = "RMSE".

errormeasure.which
Which value of errormeasure to be used for selecting a support space upper limit from support.signal.vector. One of c("min", "1se", "elbow") where "min" corresponds to the support spaces that produced the lowest error, "1se" corresponds to the support spaces such that error is within 1 standard error of the CV error for "min" and "elbow" corresponds to the elbow point of the error curve (the point that maximizes the distance between each observation, i.e, the pair composed by the upper limit of the support space and the error, and the line between the first and last observations, i.e., the lowest and the highest upper limits of the support space respectively. See find_curve_elbow). The default is errormeasure.which = "1se".

support.method One of c("standardized", "ridge"). If support.method = "standardized}, the default, standardized coefficients are used to define the signal support spaces. If \code{support.method = "ridge the signal support spaces are define by the ridge trace.

support.method.ridge.lambda
Ridge parameter. The default is support.method.ridge.lambda = NULL and a lambda logarithmic sequence will be computed based on support.method.ridge.lambda.n, support.method.ridge.lambda.min and support.method.ridge.lambda.max. Supplying a lambda sequence overrides this. To be used when support.method = "ridge".

support.method.ridge.lambda.min
Minimum value for the support.method.ridge.lambda sequence. The default is support.method.ridge.lambda.min = 10^-3. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.lambda.max
Maximum value for the support.method.ridge.lambda sequence. The default is support.method.ridge.lambda.max = 10^3. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.lambda.n
The number of ridge parameters values. The default is support.method.ridge.lambda.n = 100. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.standardize
Boolean value. If TRUE, the default, then: i) centering is done by subtracting the column means of x and y from their corresponding columns; ii) scaling is done by dividing the (centered) columns of x and y by their standard deviations. To be used when support.method = "ridge".

support.method.ridge.penalize.intercept
Boolean value. if TRUE, the default, the intercept will be penalized. To be used when support.method = "ridge" and support.method.ridge.standardize = FALSE.

support.method.ridge.symm
Boolean value. If TRUE, the default, signal supports will be symmetrical and the upper limit will be the maximum absolute values of the estimated ridge co-

efficients for support.method.ridge.lambda. If FALSE, the lower and upper limits will be, respectively, the minimum and maximum values of the estimated ridge coefficients.

support.method.ridge.maxresid
: Boolean value. if TRUE, the default, noise supports will symmetrical and the upper limit will be the maximum absolute value of the residuals of ridge estimation for support.method.ridge.lambda. If FALSE limits are computed using the empirical three-sigma rule (Pukelsheim (1994)).

support.signal
: NULL or fixed positive upper limit (L) for the support spaces (-L,L) on standardized data (when support.method = "standardized"); NULL or fixed positive factor to be multiplied by the maximum absolute value of the ridge trace for each coefficient (when support.method = "ridge"); a pair (LL,UL) or a matrix ((k+1) x 2) for the support spaces on original data. The default is support.signal = NULL.

support.signal.vector
: NULL or a vector of positive values when support.signal = NULL. If support.signal.vector = NULL, the default, a vector c(support.signal.vector.min,...,support.signal.vector.max) of dimension support.signal.vector.n and logarithmically equally spaced will be generated. Each value represents the upper limits for the standardized support spaces, when support.method = "standardized" or the factor to be multiplied by the maximum absolute value of the ridge trace for each coefficient, when support.method = "ridge".

support.signal.vector.min
: A positive value for the lowest limit of the support.signal.vector when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.min = 0.3.

support.signal.vector.max
: A positive value for the highest limit of the support.signal.vector when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.max = 20.

support.signal.vector.n
: A positive integer for the number of support spaces to be used when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.n = 20.

support.signal.points
: A vector of positive integers defining the number of points for the signal support to be tested .The default is support.signal.points = c(3, 5, 7, 9).

support.noise
: An interval, preferably centered around zero, given in the form c(LL,UL). If support.noise = NULL, the default, then a vector c(-L,L) is computed using the empirical three-sigma rule Pukelsheim (1994).

support.noise.points
: A vector of positive integers defining the number of points for the noise support to be tested .The default is support.noise.points = c(3, 5, 7, 9).

weight
: a vector of values between zero and one representing the prediction-precision loss trade-off. The default is weight = c(0.1, 0.3, 0.5, 0.7, 0.9).

twosteps.n
: Number of GCE reestimations using a previously estimated vector of signal probabilities.

method	Use "primal.solnl" (GCE using Sequential Quadratic Programming (SQP) method; see solnl) or "primal.solnp" (GCE using the augmented Lagrange multiplier method with an SQP interior algorithm; see solnp) for primal form of the optimization problem and "dual.CG" (GCE using a conjugate gradients method; see optim), "dual.BFGS" (GCE using Broyden-Fletcher-Goldfarb-Shanno quasi-Newton method; see optim), "dual.L-BFGS-B" (GCE using a box-constrained optimization with limited-memory modification of the BFGS quasi-Newton method; see optim), dual.Rcgmin (GCE using an update of the conjugate gradient algorithm; see optimx), dual.bobyqa (GCE using a derivative-free optimization by quadratic approximation; see optimx and bobyqa), dual.newuoa (GCE using a derivative-free optimization by quadratic approximation; see optimx and newuoa), dual.nlminb (GCE; see optimx and nlminb), dual.nlm (GCE; see optimx and nlm), dual.lbfgs (GCE using the Limited-memory Broyden-Fletcher-Goldfarb-Shanno; see lbfgs), dual.lbfgsb3c (GCE using L-BFSC-B implemented in Fortran code and with an Rcpp interface; see lbfgsb3c) or dual.optimParallel (GCE using parallel version of the L-BFGS-B; see optimParallel) for dual form. The default is method = "dual.BFGS".
caseGLM	special cases of the generic general linear model. One of c("D", "M", "NM"), where "D" stands for data, "M" for moment and "NM" for normed-moment The default is caseGLM = "D".
boot.B	A single positive integer greater or equal to 10 for the number of bootstrap replicates to be used for the computation of the bootstrap confidence interval(s). Zero value will generate no replicate. The default is boot.B = 0.
boot.method	Method to be use for bootstrapping. One of c("residuals", "cases", "wild") which corresponds to resampling on residuals, on individual cases or on residuals multiplied by a N(0,1) variable, respectively. The default is boot.method = "residuals".
seed	A single value, interpreted as an integer, for reproducibility or NULL for randomness. The default is seed = 230676.
OLS	Boolean value. if TRUE, the default, OLS estimation is performed.
verbose	An integer to control how verbose the output is. For a value of 0 no messages or output are shown and for a value of 3 all messages are shown. The default is verbose = 0.
coef	A vector of the true coefficients, when available.

Details

The cv.lmgce function fits several linear regression models via generalized cross according to the defined arguments. In particular, support.signal.points, support.noise.points and weight can be defined as vectors.

Value

cv.lmgce returns an object of class cv.lmgce.

An object of class cv.lmgce is a list containing at least the following components:

`results` a $C \times 8$ `data.frame`, where C is the number of combinations of the arguments `support.signal.points`, `support.noise.points` and `weight`. Contains information about the arguments, error, convergence of the optimization method and time of computation.

`best` a `lmgce` object obtained with the combination of arguments that produced the lowest cross-validation error.

`support.signal.points`
a vector of the `support.signal.points` tested.

`support.signal.points.best`
the value of `support.signal.points` that produced the lowest cross-validation error.

`support.noise.points`
a vector of the `support.noise.points` tested.

`support.noise.points.best`
the value of `support.noise.points` that produced the lowest cross-validation error.

`weight` a vector of the `weight` tested.

`weight.best` the value of `weight` that produced the lowest cross-validation error.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

References

Golan, A., Judge, G. G. and Miller, D. (1996) *Maximum entropy econometrics : robust estimation with limited data.* Wiley.
Golan, A. (2008). *Information and Entropy Econometrics — A Review and Synthesis.* Foundations and Trends® in Econometrics, 2(1–2), 1–145. doi:10.1561/0800000004
Golan, A. (2017) *Foundations of Info-Metrics: Modeling, Inference, and Imperfect Information (Vol. 1).* Oxford University Press. doi:10.1093/oso/9780199349524.001.0001
Pukelsheim, F. (1994) *The Three Sigma Rule.* The American Statistician, 48(2), 88–91. doi:10.2307/2684253

See Also

See the generic functions `plot.cv.lmgce`, `print.cv.lmgce` and `coef.cv.lmgce`.

`cv.tsbootgce` *Time series bootstrap Cross entropy estimation*

Description

This generic function fits a linear regression model using bootstrapped time series via generalized cross entropy.

Usage

```
cv.tsbootgce(
  formula,
  data,
  subset,
  na.action,
  offset,
  contrasts = NULL,
  trim = 0.05,
  reps = 1000,
  start = NULL,
  end = NULL,
  coef.method = c("median", "mode"),
  cv = TRUE,
  cv.nfolds = 5,
  errormeasure = c("RMSE", "MSE", "MAE", "MAPE", "sMAPE", "MASE"),
  errormeasure.which = {
     if (isTRUE(cv))
         c("1se", "min", "elbow")

    else c("min", "elbow")
 },
  support.method = c("standardized", "ridge"),
  support.method.ridge.lambda = NULL,
  support.method.ridge.lambda.min = 10^-3,
  support.method.ridge.lambda.max = 10^3,
  support.method.ridge.lambda.n = 100,
  support.method.ridge.standardize = TRUE,
  support.method.ridge.penalize.intercept = TRUE,
  support.method.ridge.symm = TRUE,
  support.method.ridge.maxresid = TRUE,
  support.signal = NULL,
  support.signal.vector = NULL,
  support.signal.vector.min = 0.3,
  support.signal.vector.max = 20,
  support.signal.vector.n = 20,
  support.signal.points = c(3, 5, 7, 9),
  support.noise = NULL,
  support.noise.points = c(3, 5, 7, 9),
  weight = c(0.1, 0.3, 0.5, 0.7, 0.9),
  twosteps.n = 1,
 method = c("dual.BFGS", "dual.lbfgsb3c", "primal.solnl", "primal.solnp", "dual.CG",
    "dual.L-BFGS-B", "dual.Rcgmin", "dual.bobyqa", "dual.newuoa", "dual.nlminb",
    "dual.nlm", "dual.lbfgs", "dual.optimParallel"),
  caseGLM = c("D", "M", "NM"),
  boot.B = 0,
  boot.method = c("residuals", "cases", "wild"),
  seed = 230676,
```

```
  OLS = TRUE,
  verbose = 0
)
```

Arguments

formula	a "formula" describing the linear model to be fit. For details see lm and dynlm.
data	A data.frame (or object coercible by as.data.frame to a data frame) or time series object (e.g., ts or zoo), containing the variables in the model.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
na.action	a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
offset	this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector or matrix of extents matching those of the response. One or more offset terms can be included in the formula instead or as well, and if more than one are specified their sum is used. See model.offset.
contrasts	An optional list. See the contrasts.arg of model.matrix.default.
trim	The trimming proportion (see meboot). The default is trim = 0.05.
reps	The number of replicates to generate (see meboot). The default is reps = 1000.
start	The time of the first observation. Either a single number or a vector of two numbers (the second of which is an integer), which specify a natural time unit and a (1-based) number of samples into the time unit (see ts).
end	The time of the last observation, specified in the same way as start (see ts).
coef.method	Method used to estimate the coefficients. One of c("median", "mode"). The default is coef.method = "median"
cv	Boolean value. If TRUE the error, errormeasure, will be computed using cross-validation. If FALSE the error will be computed in sample. The default is cv = TRUE.
cv.nfolds	number of folds used for cross-validation when cv = TRUE. The default is cv.nfolds = 5 and the smallest value allowable is cv.nfolds = 3.
errormeasure	Loss function (error) to be used for the selection of the support spaces. One of c("RMSE","MSE", "MAE", "MAPE", "sMAPE", "MASE"). The default is errormeasure = "RMSE".
errormeasure.which	Which value of errormeasure to be used for selecting a support space upper limit from support.signal.vector. One of c("min", "1se", "elbow") where "min" corresponds to the support spaces that produced the lowest error, "1se" corresponds to the support spaces such that error is within 1 standard error of the CV error for "min" and "elbow" corresponds to the elbow point of the error curve (the point that maximizes the distance between each observation, i.e, the pair composed by the upper limit of the support space and the error, and the

line between the first and last observations, i.e., the lowest and the highest upper limits of the support space respectively. See find_curve_elbow). The default is errormeasure.which = "1se".

support.method One of c("standardized", "ridge"). If support.method = "standardized}, the default, standardized coefficients are used to define the signal support spaces. If \code{support.method = "ridge the signal support spaces are define by the ridge trace.

support.method.ridge.lambda
Ridge parameter. The default is support.method.ridge.lambda = NULL and a lambda logarithmic sequence will be computed based on support.method.ridge.lambda.n, support.method.ridge.lambda.min and support.method.ridge.lambda.max. Supplying a lambda sequence overrides this. To be used when support.method = "ridge".

support.method.ridge.lambda.min
Minimum value for the support.method.ridge.lambda sequence. The default is support.method.ridge.lambda.min = 10^-3. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.lambda.max
Maximum value for the support.method.ridge.lambda sequence. The default is support.method.ridge.lambda.max = 10^3. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.lambda.n
The number of ridge parameters values. The default is support.method.ridge.lambda.n = 100. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.standardize
Boolean value. If TRUE, the default, then: i) centering is done by subtracting the column means of x and y from their corresponding columns; ii) scaling is done by dividing the (centered) columns of x and y by their standard deviations. To be used when support.method = "ridge".

support.method.ridge.penalize.intercept
Boolean value. if TRUE, the default, the intercept will be penalized. To be used when support.method = "ridge" and support.method.ridge.standardize = FALSE.

support.method.ridge.symm
Boolean value. If TRUE, the default, signal supports will be symmetrical and the upper limit will be the maximum absolute values of the estimated ridge coefficients for support.method.ridge.lambda. If FALSE, the lower and upper limits will be, respectively, the minimum and maximum values of the estimated ridge coefficients.

support.method.ridge.maxresid
Boolean value. if TRUE, the default, noise supports will symmetrical and the upper limit will be the maximum absolute value of the residuals of ridge estimation for support.method.ridge.lambda. If FALSE limits are computed using the empirical three-sigma rule (Pukelsheim (1994)).

support.signal NULL or fixed positive upper limit (L) for the support spaces (-L,L) on standardized data (when support.method = "standardized"); NULL or fixed positive

factor to be multiplied by the maximum absolute value of the ridge trace for each coefficient (when support.method = "ridge"); a pair (LL,UL) or a matrix ((k+1) x 2) for the support spaces on original data. The default is support.signal = NULL.

support.signal.vector
: NULL or a vector of positive values when support.signal = NULL. If support.signal.vector = NULL, the default, a vector c(support.signal.vector.min,...,support.signal.vector.max) of dimension support.signal.vector.n and logarithmically equally spaced will be generated. Each value represents the upper limits for the standardized support spaces, when support.method = "standardized" or the factor to be multiplied by the maximum absolute value of the ridge trace for each coefficient, when support.method = "ridge".

support.signal.vector.min
: A positive value for the lowest limit of the support.signal.vector when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.min = 0.3.

support.signal.vector.max
: A positive value for the highest limit of the support.signal.vector when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.max = 20.

support.signal.vector.n
: A positive integer for the number of support spaces to be used when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.n = 20.

support.signal.points
: A vector of positive integers defining the number of points for the signal support to be tested .The default is support.signal.points = c(3, 5, 7, 9).

support.noise
: An interval, preferably centered around zero, given in the form c(LL,UL). If support.noise = NULL, the default, then a vector c(-L,L) is computed using the empirical three-sigma rule (Pukelsheim (1994)).

support.noise.points
: A vector of positive integers defining the number of points for the noise support to be tested .The default is support.noise.points = c(3, 5, 7, 9).

weight
: a vector of values between zero and one representing the prediction-precision loss trade-off. The default is weight = c(0.1, 0.3, 0.5, 0.7, 0.9).

twosteps.n
: Number of GCE reestimations using a previously estimated vector of signal probabilities.

method
: Use "primal.solnl" (GCE using Sequential Quadratic Programming (SQP) method; see solnl) or "primal.solnp" (GCE using the augmented Lagrange multiplier method with an SQP interior algorithm; see solnp) for primal form of the optimization problem and "dual.CG" (GCE using a conjugate gradients method; see optim), "dual.BFGS" (GCE using Broyden-Fletcher-Goldfarb-Shanno quasi-Newton method; see optim), "dual.L-BFGS-B" (GCE using a box-constrained optimization with limited-memory modification of the BFGS quasi-Newton method; see optim), dual.Rcgmin (GCE using an update of the conjugate gradient algorithm; see optimx), dual.bobyqa (GCE using a derivative-free optimization by quadratic approximation; see optimx and bobyqa), dual.newuoa (GCE

using a derivative-free optimization by quadratic approximation; see optimx and newuoa), dual.nlminb (GCE; see optimx and nlminb), dual.nlm (GCE; see optimx and nlm), dual.lbfgs (GCE using the Limited-memory Broyden-Fletcher-Goldfarb-Shanno; see lbfgs), dual.lbfgsb3c (GCE using L-BFSC-B implemented in Fortran code and with an Rcpp interface; see lbfgsb3c) or dual.optimParallel (GCE using parallel version of the L-BFGS-B; see optimParallel) for dual form. The default is method = "dual.BFGS".

caseGLM — special cases of the generic general linear model. One of c("D", "M", "NM"), where "D" stands for data, "M" for moment and "NM" for normed-moment The default is caseGLM = "D".

boot.B — A single positive integer greater or equal to 10 for the number of bootstrap replicates to be used for the computation of the bootstrap confidence interval(s). Zero value will generate no replicate. The default is boot.B = 0.

boot.method — Method to be use for bootstrapping. One of c("residuals", "cases", "wild") which corresponds to resampling on residuals, on individual cases or on residuals multiplied by a N(0,1) variable, respectively. The default is boot.method = "residuals".

seed — A single value, interpreted as an integer, for reproducibility or NULL for randomness. The default is seed = 230676.

OLS — Boolean value. if TRUE, the default, OLS estimation is performed.

verbose — An integer to control how verbose the output is. For a value of 0 no messages or output are shown and for a value of 3 all messages are shown. The default is verbose = 0.

Details

The cv.tsbootgce function fits several linear regression models via generalized cross entropy in replicas of time series obtained using meboot. Models for cv.tsbootgce are specified symbolically (see lm and dynlm).

Value

cv.tsbootgce returns an object of class cv.tsbootgce. The generic accessory functions coef.cv.tsbootgce, confint.tsbootgce and plot.tsbootgce extract various useful features of the value returned by object of class tsbootgce.

An object of class tsbootgce is a list containing at least the following components:

call — the matched call.

coefficients — a named data frame of coefficients determined by coef.method.

data.ts — ts object.

error — loss function (error) used for the selection of the support spaces.

error.measure — in sample error for the selected support space.

fitted.values — the fitted mean values.

frequency — see link[zoo]{zoo}.

index — see link[zoo]{zoo}.

lmgce	lmgce object.
meboot	meboot replicates.
model	the model frame used.
nep	normalized entropy of the signal of the model.
nepk	normalized entropy of the signal of each coefficient.
residuals	the residuals, that is response minus fitted values.
results	a list containing the bootstrap results: "coef.matrix", a named data frame of all the coefficients; "nepk.matrix", a named data frame of all the normalized entropy values of each parameter; "nep.vector", a vector of all the normalized entropy values of the model.
seed	the seed used.
terms	the terms object used.
x	if requested (the default), the model matrix used.
xlevels	(only where relevant) a record of the levels of the factors used in fitting.
y	if requested (the default), the response used.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

References

Golan, A., Judge, G. G. and Miller, D. (1996) *Maximum entropy econometrics : robust estimation with limited data.* Wiley.

Golan, A. (2008) *Information and Entropy Econometrics — A Review and Synthesis.* Foundations and Trends® in Econometrics, 2(1–2), 1–145. doi:10.1561/0800000004

Golan, A. (2017) *Foundations of Info-Metrics: Modeling, Inference, and Imperfect Information (Vol. 1).* Oxford University Press. doi:10.1093/oso/9780199349524.001.0001

Hyndman, R.J. (1996) *Computing and graphing highest density regions.* American Statistician, 50, 120-126. doi:10.2307/2684423

Pukelsheim, F. (1994) *The Three Sigma Rule.* The American Statistician, 48(2), 88–91. doi:10.2307/2684253

Vinod, H. D., & Lopez-de-Lacalle, J. (2009). *Maximum Entropy Bootstrap for Time Series: The meboot R Package.* Journal of Statistical Software, 29(5), 1–19. doi:10.18637/jss.v029.i05

See Also

The generic functions plot.tsbootgce, print.tsbootgce, and coef.tsbootgce.

dataExample *Simulated data set generated with fngendata*

Description

Simulated data, used to demonstrate the functions of GCEstim.

Usage

```
dataExample
```

Format

A `data.frame` containing:

X001 A N(0,1) independent variable.

X002 A N(0,1) independent variable.

y A Dependent variable: y = 1 - 6 * X001 + 9 * X002 + error; the error follows a normal distribution with mean equal to zero and variance such that the signal to noise ratio is equal to 1; N = 8.

Examples

```
data(dataExample)

plot(dataExample)
```

dataGCE *Simulated data set generated with fngendata*

Description

Simulated data, used to demonstrate the functions of GCEstim. The seed used is the different from the one used to generate `dataGCE.test` but the remaining parameters are the same.

Usage

```
dataGCE
```

Format

A `data.frame` containing:

X001 A N(0,1) independent variable.

X002 A N(0,1) independent variable.

X003 A N(0,1) independent variable.

X004 A N(0,1) independent variable.

X005 A N(0,1) independent variable.

y A Dependent variable: y = 1 + 3 * X003 + 6 * X004 + 9 * X005 + error; the error follows a normal distribution with mean equal to zero and variance such that the signal to noise ratio is equal to 5.

Examples

```
data(dataGCE)

plot(dataGCE)
```

---

dataGCE.test *Simulated data set generated with fngendata*

Description

Simulated data, used to demonstrate the functions of GCEstim. The seed used is the different from the one used to generate `dataGCE` but the remaining parameters are the same.

Usage

```
dataGCE.test
```

Format

A `data.frame` containing:

X001 A N(0,1) independent variable.

X002 A N(0,1) independent variable.

X003 A N(0,1) independent variable.

X004 A N(0,1) independent variable.

X005 A N(0,1) independent variable.

y A Dependent variable: y = 1 + 3 * X003 + 6 * X004 + 9 * X005 + error; the error follows a normal distribution with mean equal to zero and variance such that the signal to noise ratio is equal to 5.

Examples

```
data(dataGCE.test)

plot(dataGCE.test)
```

dataincRidGME *Simulated data set generated with fngendata*

Description

Simulated data, used to demonstrate the functions of GCEstim.

Usage

```
dataincRidGME
```

Format

A `data.frame` containing:

X001 A N(0,1) independent variable.

X002 A N(0,1) independent variable.

X003 A N(0,1) independent variable.

X004 A N(0,1) independent variable.

X005 A N(0,1) independent variable.

X006 A N(0,1) independent variable.

y A Dependent variable: y = 2.5 - 8 * X004 + 19 * X005 - 13 * X006 + error; the error follows a normal distribution with mean equal to zero and variance such that the signal to noise ratio is equal to 1.

Examples

```
data(dataincRidGME)

plot(dataincRidGME)
```

dataincRidGME.test *Simulated data set generated with fngendata*

Description

Simulated data, used to demonstrate the functions of GCEstim.

Usage

```
dataincRidGME.test
```

Format

A `data.frame` containing:

X001 A N(0,1) independent variable.

X002 A N(0,1) independent variable.

X003 A N(0,1) independent variable.

X004 A N(0,1) independent variable.

X005 A N(0,1) independent variable.

X006 A N(0,1) independent variable.

y A Dependent variable: y = 2.5 - 8 * X004 + 19 * X005 - 13 * X006 + error; the error follows a normal distribution with mean equal to zero and variance such that the signal to noise ratio is equal to 1.

Examples

```
data(dataincRidGME.test)

plot(dataincRidGME.test)
```

---

dataThesis *Simulated data set generated with fngendata*

Description

Simulated data, used to demonstrate the functions of GCEstim.

Usage

```
dataThesis
```

Format

A `data.frame` containing:

X001 A N(0,1) independent variable.

X002 A N(0,1) independent variable.

X003 A N(0,1) independent variable.

X004 A N(0,1) independent variable.

y A Dependent variable: y = -4 - 16 * X002 - 12 * X003 - 5 * X004 + error; the error follows a normal distribution with mean equal to zero and variance such that the signal to noise ratio is equal to 5; N = 75.

Examples

```
data(dataThesis)

plot(dataThesis)
```

df.residual.lmgce *Residual Degrees-of-Freedom*

Description

Returns the residual degrees-of-freedom extracted from a fitted model lmgce object.

Usage

```
## S3 method for class 'lmgce'
df.residual(object, ...)
```

Arguments

object	Fitted lmgce model object.
...	additional arguments.

Value

The value of the residual degrees-of-freedom extracted from a lmgce object.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

df.residual(res_gce_package)
```

dynlmgce *Dynamic Linear Models and Time Series Regression using Cross entropy estimation*

Description

This generic function fits dynamic linear models using time series via generalized cross entropy.

Usage

```
dynlmgce(
  formula,
  data,
  subset,
  na.action,
  offset,
  contrasts = NULL,
  start = NULL,
  end = NULL,
  cv = TRUE,
  cv.nfolds = 5,
  errormeasure = c("RMSE", "MSE", "MAE", "MAPE", "sMAPE", "MASE"),
  errormeasure.which = {
     if (isTRUE(cv))
         c("1se", "min", "elbow")

    else c("min", "elbow")
 },
  support.method = c("standardized", "ridge"),
  support.method.ridge.lambda = NULL,
  support.method.ridge.lambda.min = 10^-3,
  support.method.ridge.lambda.max = 10^3,
  support.method.ridge.lambda.n = 100,
  support.method.ridge.standardize = TRUE,
  support.method.ridge.penalize.intercept = TRUE,
  support.method.ridge.symm = TRUE,
  support.method.ridge.maxresid = TRUE,
  support.signal = NULL,
  support.signal.vector = NULL,
  support.signal.vector.min = 0.3,
  support.signal.vector.max = 20,
  support.signal.vector.n = 20,
  support.signal.points = c(1/5, 1/5, 1/5, 1/5, 1/5),
  support.noise = NULL,
  support.noise.points = c(1/3, 1/3, 1/3),
  weight = 0.5,
  twosteps.n = 1,
 method = c("dual.BFGS", "dual.lbfgsb3c", "primal.solnl", "primal.solnp", "dual.CG",
    "dual.L-BFGS-B", "dual.Rcgmin", "dual.bobyqa", "dual.newuoa", "dual.nlminb",
     "dual.nlm", "dual.lbfgs", "dual.optimParallel"),
  caseGLM = c("D", "M", "NM"),
  boot.B = 0,
  boot.method = c("residuals", "cases", "wild"),
  seed = 230676,
  OLS = TRUE,
  verbose = 0
)
```

Arguments

formula	a "formula" describing the linear model to be fit. For details see lm and dynlm.
data	A data.frame (or object coercible by as.data.frame to a data frame) or time series object (e.g., ts or zoo), containing the variables in the model.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
na.action	a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
offset	this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector or matrix of extents matching those of the response. One or more offset terms can be included in the formula instead or as well, and if more than one are specified their sum is used. See model.offset.
contrasts	An optional list. See the contrasts.arg of model.matrix.default.
start	The time of the first observation. Either a single number or a vector of two numbers (the second of which is an integer), which specify a natural time unit and a (1-based) number of samples into the time unit (see ts).
end	The time of the last observation, specified in the same way as start (see ts).
cv	Boolean value. If TRUE the error, errormeasure, will be computed using cross-validation. If FALSE the error will be computed in sample. The default is cv = TRUE.
cv.nfolds	number of folds used for cross-validation when cv = TRUE. The default is cv.nfolds = 5 and the smallest value allowable is cv.nfolds = 3.
errormeasure	Loss function (error) to be used for the selection of the support spaces. One of c("RMSE","MSE", "MAE", "MAPE", "sMAPE", "MASE"). The default is errormeasure = "RMSE".
errormeasure.which	Which value of errormeasure to be used for selecting a support space upper limit from support.signal.vector. One of c("min", "1se", "elbow") where "min" corresponds to the support spaces that produced the lowest error, "1se" corresponds to the support spaces such that error is within 1 standard error of the CV error for "min" and "elbow" corresponds to the elbow point of the error curve (the point that maximizes the distance between each observation, i.e, the pair composed by the upper limit of the support space and the error, and the line between the first and last observations, i.e., the lowest and the highest upper limits of the support space respectively. See find_curve_elbow). The default is errormeasure.which = "1se".
support.method	One of c("standardized", "ridge"). If support.method = "standardized}, the default, standardized coefficients are used to define the signal support spaces. If \code{support.method = "ridge the signal support spaces are define by the ridge trace.

support.method.ridge.lambda
: Ridge parameter. The default is support.method.ridge.lambda = NULL and a lambda logarithmic sequence will be computed based on support.method.ridge.lambda.n, support.method.ridge.lambda.min and support.method.ridge.lambda.max. Supplying a lambda sequence overrides this. To be used when support.method = "ridge".

support.method.ridge.lambda.min
: Minimum value for the support.method.ridge.lambda sequence. The default is support.method.ridge.lambda.min = 10^-3. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.lambda.max
: Maximum value for the support.method.ridge.lambda sequence. The default is support.method.ridge.lambda.max = 10^3. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.lambda.n
: The number of ridge parameters values. The default is support.method.ridge.lambda.n = 100. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.standardize
: Boolean value. If TRUE, the default, then: i) centering is done by subtracting the column means of x and y from their corresponding columns; ii) scaling is done by dividing the (centered) columns of x and y by their standard deviations. To be used when support.method = "ridge".

support.method.ridge.penalize.intercept
: Boolean value. if TRUE, the default, the intercept will be penalized. To be used when support.method = "ridge" and support.method.ridge.standardize = FALSE.

support.method.ridge.symm
: Boolean value. If TRUE, the default, signal supports will be symmetrical and the upper limit will be the maximum absolute values of the estimated ridge coefficients for support.method.ridge.lambda. If FALSE, the lower and upper limits will be, respectively, the minimum and maximum values of the estimated ridge coefficients.

support.method.ridge.maxresid
: Boolean value. if TRUE, the default, noise supports will symmetrical and the upper limit will be the maximum absolute value of the residuals of ridge estimation for support.method.ridge.lambda. If FALSE limits are computed using the empirical three-sigma rule (Pukelsheim (1994)).

support.signal
: NULL or fixed positive upper limit (L) for the support spaces (-L,L) on standardized data (when support.method = "standardized"); NULL or fixed positive factor to be multiplied by the maximum absolute value of the ridge trace for each coefficient (when support.method = "ridge"); a pair (LL,UL) or a matrix ((k+1) x 2) for the support spaces on original data. The default is support.signal = NULL.

support.signal.vector
: NULL or a vector of positive values when support.signal = NULL. If support.signal.vector = NULL, the default, a vector c(support.signal.vector.min,...,support.signal.vector.max) of dimension support.signal.vector.n and logarithmically equally spaced

will be generated. Each value represents the upper limits for the standardized support spaces, when support.method = "standardized" or the factor to be multiplied by the maximum absolute value of the ridge trace for each coefficient, when support.method = "ridge".

support.signal.vector.min
: A positive value for the lowest limit of the support.signal.vector when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.min = 0.3.

support.signal.vector.max
: A positive value for the highest limit of the support.signal.vector when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.max = 20.

support.signal.vector.n
: A positive integer for the number of support spaces to be used when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.n = 20.

support.signal.points
: A positive integer, a vector or a matrix. Prior weights for the signal. If not a positive integer then the sum of weights by row must be equal to 1. The default is support.signal.points = c(1 / 5, 1 / 5, 1 / 5, 1 / 5, 1 / 5).

support.noise
: An interval, preferably centered around zero, given in the form c(LL,UL). If support.noise = NULL, the default, then a vector c(-L,L) is computed using the empirical three-sigma rule (Pukelsheim (1994)).

support.noise.points
: A positive integer, a vector or a matrix. Prior weights for the noise. If not a positive integer then the sum of weights by row must be equal to 1. The default is support.noise.points = c(1 / 3, 1 / 3, 1 / 3).

weight
: a value between zero and one representing the prediction-precision loss trade-off. If weight = 0.5, the default, equal weight is placed on the signal and noise entropies. A higher than 0.5 value places more weight on the noise entropy whereas a lower than 0.5 value places more weight on the signal entropy.

twosteps.n
: Number of GCE reestimations using a previously estimated vector of signal probabilities.

method
: Use "primal.solnl" (GCE using Sequential Quadratic Programming (SQP) method; see solnl) or "primal.solnp" (GCE using the augmented Lagrange multiplier method with an SQP interior algorithm; see solnp) for primal form of the optimization problem and "dual.CG" (GCE using a conjugate gradients method; see optim), "dual.BFGS" (GCE using Broyden-Fletcher-Goldfarb-Shanno quasi-Newton method; see optim), "dual.L-BFGS-B" (GCE using a box-constrained optimization with limited-memory modification of the BFGS quasi-Newton method; see optim), dual.Rcgmin (GCE using an update of the conjugate gradient algorithm; see optimx), dual.bobyqa (GCE using a derivative-free optimization by quadratic approximation; see optimx and bobyqa), dual.newuoa (GCE using a derivative-free optimization by quadratic approximation; see optimx and newuoa), dual.nlminb (GCE; see optimx and nlminb), dual.nlm (GCE; see optimx and nlm), dual.lbfgs (GCE using the Limited-memory Broyden-Fletcher-Goldfarb-Shanno; see lbfgs), dual.lbfgsb3c (GCE using L-BFSC-B implemented in Fortran code and with an Rcpp interface; see lbfgsb3c)

	or dual.optimParallel (GCE using parallel version of the L-BFGS-B; see optimParallel) for dual form. The default is method = "dual.BFGS".
caseGLM	special cases of the generic general linear model. One of c("D", "M", "NM"), where "D" stands for data, "M" for moment and "NM" for normed-moment The default is caseGLM = "D".
boot.B	A single positive integer greater or equal to 10 for the number of bootstrap replicates to be used for the computation of the bootstrap confidence interval(s). Zero value will generate no replicate. The default is boot.B = 0.
boot.method	Method to be use for bootstrapping. One of c("residuals", "cases", "wild") which corresponds to resampling on residuals, on individual cases or on residuals multiplied by a N(0,1) variable, respectively. The default is boot.method = "residuals".
seed	A single value, interpreted as an integer, for reproducibility or NULL for randomness. The default is seed = 230676.
OLS	Boolean value. if TRUE, the default, OLS estimation is performed.
verbose	An integer to control how verbose the output is. For a value of 0 no messages or output are shown and for a value of 3 all messages are shown. The default is verbose = 0.

Details

The dynlmgce function fits linear regression models via generalized cross entropy time series. Models for dynlmgce are specified symbolically (see lm and dynlm).

Value

dynlmgce returns an object of class lmgce. The generic accessory functions coef.lmgce, confint.lmgce and plot.lmgce extract various useful features of the value returned by object of class lmgce.

The dynlmgce function returns a list containing at least the following components:

call	the matched call.
coefficients	a named data frame of coefficients.
data.ts	ts object.
error	loss function (error) used for the selection of the support spaces.
error.measure	in sample error for the selected support space.
fitted.values	the fitted mean values.
frequency	see link[zoo]{zoo}.
index	see link[zoo]{zoo}.
lmgce	lmgce object.
model	the model frame used.
nep	normalized entropy of the signal of the model.
nepk	normalized entropy of the signal of each coefficient.
residuals	the residuals, that is response minus fitted values.

results	a list containing the bootstrap results: "coef.matrix", a named data frame of all the coefficients; "nepk.matrix", a named data frame of all the normalized entropy values of each parameter; "nep.vector", a vector of all the normalized entropy values of the model.
seed	the seed used.
terms	the terms object used.
x	if requested (the default), the model matrix used.
xlevels	(only where relevant) a record of the levels of the factors used in fitting.
y	if requested (the default), the response used.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

References

Golan, A., Judge, G. G. and Miller, D. (1996) *Maximum entropy econometrics : robust estimation with limited data.* Wiley.

Golan, A. (2008) *Information and Entropy Econometrics — A Review and Synthesis.* Foundations and Trends® in Econometrics, 2(1–2), 1–145. doi:10.1561/0800000004

Golan, A. (2017) *Foundations of Info-Metrics: Modeling, Inference, and Imperfect Information (Vol. 1).* Oxford University Press. doi:10.1093/oso/9780199349524.001.0001

Pukelsheim, F. (1994) *The Three Sigma Rule.* The American Statistician, 48(2), 88–91. doi:10.2307/2684253

See Also

The generic functions plot.lmgce, print.lmgce, and coef.lmgce.

Examples

```
res.dynlmgce <-
  dynlmgce(
    formula = CO2 ~ 1 + L(EPC, 1) + L(EUS, 2) + L(GDP, 0),
    data = moz_ts)

res.dynlmgce
```

ER.test *Entropy Ratio test*

Description

The Entropy Ratio test - which corresponds to the likelihood ratio, or empirical ratio, test - measures the entropy discrepancy between the constrained and the unconstrained models.

Usage

```
ER.test(object)
```

Arguments

object	fitted lmgce object.

Value

A matrix with the X-squared statistics, degrees of freedom and p-value for each parameter.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)


ER.test(res_gce_package)
```

fitted.lmgce *Calculate lmgce Fitted Values*

Description

The fitted values for the linear model represented by a lmgce object are extracted.

Usage

```
## S3 method for class 'lmgce'
fitted(object, ...)
```

Arguments

object	Fitted lmgce model object.
...	additional arguments.

Value

Returns a vector with the fitted values for the linear model represented by a lmgce object.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

fitted(res_gce_package)
```

---

fitted.values.lmgce *Calculate* lmgce *Fitted Values*

Description

The fitted values for the linear model represented by a lmgce object are extracted.

Usage

```
## S3 method for class 'lmgce'
fitted.values(object, ...)
```

Arguments

object	Fitted lmgce model object.
...	additional arguments.

Value

Returns a vector with the fitted values for the linear model represented by a lmgce object.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

fitted.values(res_gce_package)
```

---

fngendata *Data generating function*

Description

Generates data

Usage

```
fngendata(
  n,
  bin.k = 0,
  bin.prob = NULL,
  cont.k = 5,
  y.gen.bin.k = 0,
  y.gen.bin.beta = NULL,
  y.gen.bin.prob = NULL,
  y.gen.cont.beta = c(2, 4, 6, 8, 10),
  y.gen.cont.mod.k = 0,
  y.gen.cont.mod.beta = matrix(c(-2, 2), 1, 2, byrow = TRUE),
  y.gen.bin.mod.prob = c(0.5),
  y.gen.cont.sp.k = 0,
  y.gen.cont.sp.groups = 2,
  y.gen.cont.sp.rho = 0.2,
  y.gen.cont.sp.dif = 1,
  intercept.beta = 0,
  Xgenerator.method = "simstudy",
  corMatrix = 100,
  rho = NULL,
  corstr = NULL,
  condnumber = 1,
  mu = 0,
  muvect = NULL,
  sd = 1,
  sdvect = NULL,
  error.dist = "normal",
```

```
  error.dist.mean = 0,
  error.dist.sd = 1,
  error.dist.snr = NULL,
  error.dist.df = 2,
  dataframe = TRUE,
  seed = NULL
)
```

Arguments

n	Number of individuals.
bin.k	Number of binary variables not used for generating y.
bin.prob	A vector of probabilities with length equal to bin.k.
cont.k	Number of continuous variables not used for generating y.
y.gen.bin.k	Number of binary variables used for generating y.
y.gen.bin.beta	A vector of coefficients with length equal to bin.kused to generate y.
y.gen.bin.prob	A vector of probabilities with length equal to y.gen.bin.k.
y.gen.cont.beta	A vector of coefficients with length equal to cont.k used to generate y.
y.gen.cont.mod.k	Experimental
y.gen.cont.mod.beta	Experimental
y.gen.bin.mod.prob	Experimental
y.gen.cont.sp.k	Experimental
y.gen.cont.sp.groups	Experimental
y.gen.cont.sp.rho	Experimental
y.gen.cont.sp.dif	Experimental
intercept.beta	Value for the constant used to generate y.
Xgenerator.method	Method used to generate X data ("simstudy" or "svd").
corMatrix	A positive number for alphad (see rcorrmatrix), NULL or a correlation matrix to be used when Xgenerator is "simstudy".
rho	Correlation coefficient, -1 <= rho <= 1. Use when Xgenerator is "simstudy" and corMatrix is NULL.
corstr	correlation structure ("ind", "cs" or "ar1") (see genCorData) to be used when Xgenerator is "simstudy" and corMatrix is NULL.
condnumber	A value for the condition number of the X matrix to be used when Xgenerator is "svd".

mu	The mean of the variables. To be used when all variables have the same mean.
muvect	A vector of means. To be used when variables have different means. The length of muvect must be k.
sd	Standard deviation of the variables. To be used when all variables have the same standard deviation.
sdvect	A vector of standard deviations. To be used when variables have different standard deviations. The length of sdvect must be k.
error.dist	Distribution of the error. "normal" for normal distribution or "t" for t-student distribution.
error.dist.mean	Mean value used when error.dist is "normal".
error.dist.sd	Standard deviation value used when error.dist is "normal".
error.dist.snr	Signal to noise ratio. If not NULL, the value of error.dist.sd will be ignored and it will be determined accordingly.
error.dist.df	Degrees of freedom used when error.dist is "t".
dataframe	Logical. If TRUE, the default, returns a data.frame else returns a list.
seed	A seed for reproducibility.

Value

A data.frame or a list composed of a matrix of independent variables values (X), a vector of the dependent variable values (y), a vector of coefficient values (coefficients), a vector of non-zero coefficients (y.coefficients), and a vector of the error values (epsilon).

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
dataThesis <-
  fngendata(
    n = 75,
    cont.k = 1,
    y.gen.cont.beta = c(-16, -12, -5),
    intercept.beta = -4,
    Xgenerator.method = "svd", condnumber = 200,
    mu = 0, sd = 1,
    error.dist = "normal", error.dist.mean = 0, error.dist.snr = 5,
    dataframe = TRUE, seed = 230687)

summary(dataThesis)
```

formula.lmgce *Extract Model Formula from* lmgce *object*

Description

Returns the model used to fit lmgce object.

Usage

```
## S3 method for class 'lmgce'
formula(x, ...)
```

Arguments

x	fitted lmgce object.
...	additional arguments.

Value

An object of class formula representing the model formula.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

formula(res_gce_package)
```

lmgce *Generalized Cross entropy estimation*

Description

This generic function fits a linear regression model via generalized cross entropy. Initial support spaces can be provided or computed.

Usage

```
lmgce(
  formula,
  data,
  subset,
  na.action,
  offset,
  contrasts = NULL,
  model = TRUE,
  x = FALSE,
  y = FALSE,
  cv = TRUE,
  cv.nfolds = 5,
  errormeasure = c("RMSE", "MSE", "MAE", "MAPE", "sMAPE", "MASE"),
  errormeasure.which = {
     if (isTRUE(cv))
         c("1se", "min", "elbow")

    else c("min", "elbow")
 },
  support.method = c("standardized", "ridge"),
  support.method.ridge.lambda = NULL,
  support.method.ridge.lambda.min = 10^-3,
  support.method.ridge.lambda.max = 10^3,
  support.method.ridge.lambda.n = 100,
  support.method.ridge.standardize = TRUE,
  support.method.ridge.penalize.intercept = TRUE,
  support.method.ridge.symm = TRUE,
  support.method.ridge.maxresid = TRUE,
  support.signal = NULL,
  support.signal.vector = NULL,
  support.signal.vector.min = 0.3,
  support.signal.vector.max = 20,
  support.signal.vector.n = 20,
  support.signal.points = c(1/5, 1/5, 1/5, 1/5, 1/5),
  support.noise = NULL,
  support.noise.points = c(1/3, 1/3, 1/3),
  weight = 0.5,
```

```
  twosteps.n = 1,
 method = c("dual.BFGS", "dual.lbfgsb3c", "primal.solnl", "primal.solnp", "dual.CG",
   "dual.L-BFGS-B", "dual.Rcgmin", "dual.bobyqa", "dual.newuoa", "dual.nlminb",
    "dual.nlm", "dual.lbfgs", "dual.optimParallel"),
  caseGLM = c("D", "M", "NM"),
  boot.B = 0,
  boot.method = c("residuals", "cases", "wild"),
  seed = 230676,
  OLS = TRUE,
  verbose = 0
)
```

Arguments

formula	An object of class formula (or one that can be coerced to that class): a symbolic description of the model to be fitted.
data	A data frame (or object coercible by as.data.frame to a data frame) containing the variables in the model.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
na.action	a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
offset	this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector or matrix of extents matching those of the response. One or more offset terms can be included in the formula instead or as well, and if more than one are specified their sum is used. See model.offset.
contrasts	An optional list. See the contrasts.arg of model.matrix.default.
model	Boolean value. if TRUE, the model frame used is returned. The default is model = TRUE.
x	Boolean value. if TRUE, the model matrix used is returned. The default is x = FALSE.
y	Boolean value. if TRUE, the response used is returned. The default is y = FALSE.
cv	Boolean value. If TRUE the error, errormeasure, will be computed using cross-validation. If FALSE the error will be computed in sample. The default is cv = TRUE.
cv.nfolds	number of folds used for cross-validation when cv = TRUE. The default is cv.nfolds = 5 and the smallest value allowable is cv.nfolds = 3.
errormeasure	Loss function (error) to be used for the selection of the support spaces. One of c("RMSE","MSE", "MAE", "MAPE", "sMAPE", "MASE"). The default is errormeasure = "RMSE".
errormeasure.which	Which value of errormeasure to be used for selecting a support space upper limit from support.signal.vector. One of c("min", "1se", "elbow")

where "min" corresponds to the support spaces that produced the lowest error, "1se" corresponds to the support spaces such that error is within 1 standard error of the CV error for "min" and "elbow" corresponds to the elbow point of the error curve (the point that maximizes the distance between each observation, i.e, the pair composed by the upper limit of the support space and the error, and the line between the first and last observations, i.e., the lowest and the highest upper limits of the support space respectively. See find_curve_elbow). The default is errormeasure.which = "1se".

support.method One of c("standardized", "ridge"). If support.method = "standardized}, the default, standardized coefficients are used to define the signal support spaces. If \code{support.method = "ridge the signal support spaces are define by the ridge trace.

support.method.ridge.lambda
Ridge parameter. The default is support.method.ridge.lambda = NULL and a lambda logarithmic sequence will be computed based on support.method.ridge.lambda.n, support.method.ridge.lambda.min and support.method.ridge.lambda.max. Supplying a lambda sequence overrides this. To be used when support.method = "ridge".

support.method.ridge.lambda.min
Minimum value for the support.method.ridge.lambda sequence. The default is support.method.ridge.lambda.min = 10^-3. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.lambda.max
Maximum value for the support.method.ridge.lambda sequence. The default is support.method.ridge.lambda.max = 10^3. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.lambda.n
The number of ridge parameters values. The default is support.method.ridge.lambda.n = 100. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.standardize
Boolean value. If TRUE, the default, then: i) centering is done by subtracting the column means of x and y from their corresponding columns; ii) scaling is done by dividing the (centered) columns of x and y by their standard deviations. To be used when support.method = "ridge".

support.method.ridge.penalize.intercept
Boolean value. if TRUE, the default, the intercept will be penalized. To be used when support.method = "ridge" and support.method.ridge.standardize = FALSE.

support.method.ridge.symm
Boolean value. If TRUE, the default, signal supports will be symmetrical and the upper limit will be the maximum absolute values of the estimated ridge coefficients for support.method.ridge.lambda. If FALSE, the lower and upper limits will be, respectively, the minimum and maximum values of the estimated ridge coefficients.

support.method.ridge.maxresid
Boolean value. if TRUE, the default, noise supports will symmetrical and the upper limit will be the maximum absolute value of the residuals of ridge estima-

tion for support.method.ridge.lambda. If FALSE limits are computed using the empirical three-sigma rule (Pukelsheim (1994)).

support.signal NULL or fixed positive upper limit (L) for the support spaces (-L,L) on standardized data (when support.method = "standardized"); NULL or fixed positive factor to be multiplied by the maximum absolute value of the ridge trace for each coefficient (when support.method = "ridge"); a pair (LL,UL) or a matrix ((k+1) x 2) for the support spaces on original data. The default is support.signal = NULL.

support.signal.vector
NULL or a vector of positive values when support.signal = NULL. If support.signal.vector = NULL, the default, a vector c(support.signal.vector.min,...,support.signal.vector.max) of dimension support.signal.vector.n and logarithmically equally spaced will be generated. Each value represents the upper limits for the standardized support spaces, when support.method = "standardized" or the factor to be multiplied by the maximum absolute value of the ridge trace for each coefficient, when support.method = "ridge".

support.signal.vector.min
A positive value for the lowest limit of the support.signal.vector when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.min = 0.3.

support.signal.vector.max
A positive value for the highest limit of the support.signal.vector when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.max = 20.

support.signal.vector.n
A positive integer for the number of support spaces to be used when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.n = 20.

support.signal.points
A positive integer, a vector or a matrix. Prior weights for the signal. If not a positive integer then the sum of weights by row must be equal to 1. The default is support.signal.points = c(1 / 5, 1 / 5, 1 / 5, 1 / 5, 1 / 5).

support.noise An interval, preferably centered around zero, given in the form c(LL,UL). If support.noise = NULL, the default, then a vector c(-L,L) is computed using the empirical three-sigma rule (Pukelsheim (1994)).

support.noise.points
A positive integer, a vector or a matrix. Prior weights for the noise. If not a positive integer then the sum of weights by row must be equal to 1. The default is support.noise.points = c(1 / 3, 1 / 3, 1 / 3).

weight a value between zero and one representing the prediction-precision loss trade-off. If weight = 0.5, the default, equal weight is placed on the signal and noise entropies. A higher than 0.5 value places more weight on the noise entropy whereas a lower than 0.5 value places more weight on the signal entropy.

twosteps.n Number of GCE reestimations using a previously estimated vector of signal probabilities.

method Use "primal.solnl" (GCE using Sequential Quadratic Programming (SQP) method; see solnl) or "primal.solnp" (GCE using the augmented Lagrange

multiplier method with an SQP interior algorithm; see solnp) for primal form of the optimization problem and "dual.CG" (GCE using a conjugate gradients method; see optim), "dual.BFGS" (GCE using Broyden-Fletcher-Goldfarb-Shanno quasi-Newton method; see optim), "dual.L-BFGS-B" (GCE using a box-constrained optimization with limited-memory modification of the BFGS quasi-Newton method; see optim), dual.Rcgmin (GCE using an update of the conjugate gradient algorithm; see optimx), dual.bobyqa (GCE using a derivative-free optimization by quadratic approximation; see optimx and bobyqa), dual.newuoa (GCE using a derivative-free optimization by quadratic approximation; see optimx and newuoa), dual.nlminb (GCE; see optimx and nlminb), dual.nlm (GCE; see optimx and nlm), dual.lbfgs (GCE using the Limited-memory Broyden-Fletcher-Goldfarb-Shanno; see lbfgs), dual.lbfgsb3c (GCE using L-BFSC-B implemented in Fortran code and with an Rcpp interface; see lbfgsb3c) or dual.optimParallel (GCE using parallel version of the L-BFGS-B; see optimParallel) for dual form. The default is method = "dual.BFGS".

caseGLM — special cases of the generic general linear model. One of c("D", "M", "NM"), where "D" stands for data, "M" for moment and "NM" for normed-moment The default is caseGLM = "D".

boot.B — A single positive integer greater or equal to 10 for the number of bootstrap replicates to be used for the computation of the bootstrap confidence interval(s). Zero value will generate no replicate. The default is boot.B = 0.

boot.method — Method to be use for bootstrapping. One of c("residuals", "cases", "wild") which corresponds to resampling on residuals, on individual cases or on residuals multiplied by a N(0,1) variable, respectively. The default is boot.method = "residuals".

seed — A single value, interpreted as an integer, for reproducibility or NULL for randomness. The default is seed = 230676.

OLS — Boolean value. if TRUE, the default, OLS estimation is performed.

verbose — An integer to control how verbose the output is. For a value of 0 no messages or output are shown and for a value of 3 all messages are shown. The default is verbose = 0.

Details

The lmgce function fits a linear regression model via generalized cross entropy. Models for lmgce are specified symbolically. A typical model has the form response ~ terms where response is the (numeric) response vector and terms is a series of terms which specifies a linear predictor for response. lmgce calls the lower level functions lmgce.validate, lmgce.assign.ci, lmgce.assign.noci, lmgce.sscv, lmgce.ss, lmgce.cv and lmgce.fit.

Value

lmgce returns an object of class lmgce. The function summary.lmgce is used to obtain and print a summary of the results. The generic accessory functions coef.lmgce, fitted.values.lmgce, residuals.lmgce and df.residual.lmgce, extract various useful features of the value returned by object of class lmgce.

An object of class lmgce is a list containing at least the following components:

coefficients	a named vector of coefficients.
residuals	the residuals, that is response minus fitted values.
fitted.values	the fitted mean values.
df.residual	the residual degrees of freedom.
call	the matched call.
terms	the terms object used.
contrast	(only where relevant) the contrasts used.
xlevels	(only where relevant) a record of the levels of the factors used in fitting.
offset	the offset used (missing if none were used).
y	if requested (the default), the response used.
x	if requested (the default), the model matrix used.
model	if requested (the default), the model frame used.
na.action	(where relevant) information returned by model.frame on the special handling of NAs.
boot.B	number of bootstrap replicates used.
boot.method	method used for bootstrapping.
caseGLM	case of the generic general linear model used.
convergence	an integer code. 0 indicates successful optimization completion. Other numbers indicate different errors. See optim, optimx, solnl, solnp, lbfgs) and lbfgsb3c).
error	loss function (error) used for the selection of the support spaces.
error.measure	in sample error for the selected support space.
error.measure.cv.mean	cross-validation mean error for the selected support space.
error.measure.cv.sd	standard deviation of the cross-validation error for the selected support space.
error.which	which criterion/standardized/factor support was used
support.signal.1se	upper limit of the standardized support space or factor that produced the error within one standard error from the minimum error.
support.signal.elbow	upper limit of the standardized support space or factor that produced the error correspondent to the elbow of the error curve.
support.signal.min	upper limit of the standardized support space or factor that produced the minimum error.
p0	vector of prior weights used for the signal.
p	estimated probabilities associated with the signal.
w0	vector of prior weights used for the noise.
w	estimated probabilities associated with the noise.

`lambda`	estimated Lagrange multipliers.
`nep`	normalized entropy of the signal of the model.
`nep.cv.mean`	cross-validation normalized entropy of the signal of the model.
`nep.cv.sd`	standard deviation of the cross-validation normalized entropy of the signal of the model.
`nepk`	normalized entropy of the signal of each coefficient.
`results`	results from the different support spaces with or without cross-validation, and from bootstrap replicates, namely number of attempts (if the number of attempts is greater than three times the number of bootstrap replicates the bootstrapping process stops), coefficients and normalized entropies (nep - model, and nepk - coefficients), when applicable; results from OLS estimation if `OLS = TRUE`; results from GCE reestimation if `twosteps.n` is greater than 0.
`support`	vector of given positive upper limits for the support spaces on standardized data or factors, when `support.signal = NULL` or `support.signal = L`, or `"interval"` otherwise.
`support.matrix`	matrix with the support spaces used for estimation on original data.
`support.method`	method chosen for the support's limits
`support.ok`	vector of successful positive upper limits for the support spaces on standardized data (`support.method = "standardized"`) or factors (`support.method = "ridge"`), when `support.signal = NULL` or `support.signal = L`, or `"interval"` otherwise.
`support.stdUL`	when applicable, the upper limit of the standardized support chosen, when `support.method = "standardized"` or the factor used when `support.method = "ridge"`.
`vcov`	variance-covariance matrix of the coefficients.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

References

Golan, A., Judge, G. G. and Miller, D. (1996) *Maximum entropy econometrics : robust estimation with limited data.* Wiley.

Golan, A. (2008). *Information and Entropy Econometrics — A Review and Synthesis.* Foundations and Trends® in Econometrics, 2(1–2), 1–145. doi:10.1561/0800000004

Golan, A. (2017) *Foundations of Info-Metrics: Modeling, Inference, and Imperfect Information (Vol. 1).* Oxford University Press. doi:10.1093/oso/9780199349524.001.0001

Pukelsheim, F. (1994) *The Three Sigma Rule.* The American Statistician, 48(2), 88–91. doi:10.2307/2684253

Macedo, P., Cabral, J., Afreixo, V., Macedo, F., Angelelli, M. (2025) *RidGME estimation and inference in ill-conditioned models.* In: Gervasi O, Murgante B, Garau C, et al., eds. Computational Science and Its Applications – ICCSA 2025 Workshops. Springer Nature Switzerland; 2025:300-313. doi:10.1007/9783031975899_21

See Also

summary.lmgce for more detailed summaries. The generic functions plot.lmgce, print.lmgce, coef.lmgce and confint.lmgce.

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)


res_gce_package
```

---

lmgceAddin *An add-in to easily generate the code for a* lmgce *or* cv.lmgce *analysis*

Description

Select data and choose the arguments to be used. The execution of the code is also possible within the addin.

Usage

```
lmgceAddin()
```

Details

An addin for lmgce and cv.lmgce

Value

The code to be use in the lmgce analysis.

Examples

```
lmgceAddin()
```

lmgceAPP — lmgce *Shiny application*

Description

A Shiny application to execute lmgce

Usage

```
lmgceAPP()
```

Value

NULL. This function is called for its side effect (launching the app).

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

model.matrix.lmgce — *Extract design matrix from* lmgce *object*

Description

Returns the design matrix used to fit lmgce object.

Usage

```
## S3 method for class 'lmgce'
model.matrix(object, ...)
```

Arguments

object	fitted lmgce object.
...	additional arguments.

Value

A numeric matrix with one row for each observation and one column for each parameter in the model.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

model.matrix(res_gce_package)
```

`moz_ts` *Worldbank time series data for Mozambique*

Description

Mozambique's CO2, GDP, EPC e EUS time series (1991-2014) from https://databank.worldbank.org/ (Downloaded in 2024/12/03).

Usage

```
moz_ts
```

Format

A `ts` object containing:

year Year to which data refers

CO2 CO2 emissions (metric tons per capita); Data from: IDA Results Measurement System, Tier I Database – WDI;

EPC Electric power consumption (kWh per capita). Data from database: Jobs;

EUS Energy use (kg of oil equivalent per capita). Data from database: World Development Indicators;

GDP Gross domestic product per capita (current US$); Data from: World Development Indicators.

Examples

```
data(moz_ts)

plot(moz_ts)
```

neagging *Normalized Entropy Aggregation for Inhomogeneous Large-Scale Data - Neagging*

Description

Computes the estimates for the Normalized Entropy Aggregation

Usage

```
neagging(
  object,
  boot.B = ifelse(object$boot.B == 0, 100, object$boot.B),
  boot.method = object$boot.method,
  error = object$error
)
```

Arguments

object	Fitted lmgce or tsbootgce model object.
boot.B	To use with a lmgce object. A single positive integer greater or equal to 10 for the number of bootstrap replicates for the computation of the Normalized Entropy Aggregation estimate(s), to be used when object was created with boot.B = 0. The default is boot.B = 100 when the object has no previous sampling information and boot.B = object$boot.B otherwise, which corresponds to the boot.B given to lmgce when the object was created.
boot.method	To use with a lmgce object. Method used for bootstrapping. One of c("residuals", "cases", "wild") which corresponds to resampling on residuals, on individual cases or on residuals multiplied by a N(0,1) variable, respectively. The default is boot.method = object$boot.method.
error	Loss function (error) to be used for the selection of the support spaces. One of c("RMSE","MSE", "MAE", "MAPE", "sMAPE", "MASE"). The default is boot.method = object$error.

Value

An object of class neagging is a list containing at least the following components:

matrix	a matrix where each column contains sequentially the aggregated estimates.
error	a named vector with the in sample error for each aggregated set of estimates.
error.object	the in sample error of the object.
coefficients	the aggregated coefficients that produced the lowest in sample error.
coefficients.object	the coefficients of the object.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

References

da Conceição Costa, M. and Macedo, P. (2019). *Normalized Entropy Aggregation for Inhomogeneous Large-Scale Data.* In O. Valenzuela, F. Rojas, H. Pomares, & I. Rojas (Eds.), Theory and Applications of Time Series Analysis (pp. 19–29). Springer International Publishing. doi:10.1007/9783030260361_2

See Also

The generic functions `plot.neagging` and `coef.neagging`.

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

res_neagging <- neagging(res_gce_package, boot.method = "cases")

res_neagging

res.tsbootgce <-
  tsbootgce(
    formula = CO2 ~ 1 + L(EPC, 1) + L(EUS, 2) + L(GDP, 0),
    data = moz_ts)

neagging(res.tsbootgce)
```

---

nobs.lmgce *Extract the Number of Observations from a* `lmgce` *model fit*

Description

Extract the number of 'observations' from a `lmgce` model fit.

Usage

```
## S3 method for class 'lmgce'
nobs(object, ...)
```

Arguments

object	Fitted lmgce model object.
...	additional arguments.

Value

An integer scalar representing the number of observations (rows) used in fitting the lmgce model object.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)


nobs(res_gce_package)
```

NormEnt *Normalized Entropy*

Description

Returns the normalized entropy of the model or the normalized entropy of the predictors.

Usage

```
NormEnt(object, model = TRUE, parm)
```

Arguments

object	fitted lmgce or tsbootgce object.
model	Boolean value. If model = TRUE, the model's normalized entropy is returned. If model = FALSE the normalized entropy of each estimate is returned. The default is model = TRUE.
parm	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.

Value

the value of the normalized entropy of the model or parameters.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)


NormEnt(res_gce_package)
```

---

plot.cv.lmgce *Plot Diagnostics for a* cv.lmgce *Object*

Description

Two plots (selectable by which) are currently available to evaluate a cv.lmgce object. The plots depicts the error change with the combination of different arguments of cv.lmgce and also the best combination.

Usage

```
## S3 method for class 'cv.lmgce'
plot(x, which = c(1, 2), ncol = 1, scales = "free", ...)
```

Arguments

x	Fitted cv.lmgce model object.
which	A subset of the numbers 1:2.
ncol	Number of columns of the plot (see facet_wrap); to use when which=1.
scales	One of c("free", "fixed") (see facet_wrap); to use when which=1.
...	additional arguments.

Value

A list of ggplot objects.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

See Also

cv.lmgce

---

plot.cv.tsbootgce *Plot Diagnostics for a* cv.tsbootgce *object*

Description

Four plots (selectable by which) are currently available to evaluate a cv.tsbootgce object.

Usage

```
## S3 method for class 'cv.tsbootgce'
plot(
  x,
  which = c(1, 2),
  group = TRUE,
  group.ncol = NULL,
  group.nrow = NULL,
  ci.levels = c(0.9, 0.95, 0.99),
  ci.method = c("hdr", "basic", "percentile"),
  seed = object$seed,
  lambda = 1,
  col = NULL,
  plot.lines = TRUE,
  legend.position = "bottom",
  ...
)
```

Arguments

x	Fitted cv.tsbootgce object.
which	Integers from 1 to 4. The default is which = c(1,2).
group	Boolean value. If group = TRUE, the default, plots are grouped in one image.
group.ncol	Number of columns (see ggarrange). The default is group.ncol = NULL.
group.nrow	Number of rows. (see ggarrange). The default is group.nrow = NULL.
ci.levels	the confidence levels (maximum of 4) required to compute the confidence interval. The default is ci.levels = c(0.90, 0.95, 0.99).
ci.method	One of c("hdr", "basic", "percentile"). The default is ci.method = "hdr" (see hdr).

seed	A single value, interpreted as an integer, for reproducibility or NULL for randomness. The default is seed = object$seed.
lambda	Box-Cox transformation parameter. Value between 0 and 1. The default is lambda = 1 (see hdr).
col	Vector of colors for regions. The default is col = NULL.
plot.lines	Boolean. The default is plot.lines = TRUE.
legend.position	The default is legend.position = "bottom".
...	additional arguments.

Value

A ggplot object.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

See Also

cv.tsbootgce

---

plot.lmgce *Plot Diagnostics for a* lmgce *Object*

Description

Seven plots (selectable by which) are currently available to evaluate a lmgce object: a plot of the Estimates and confidence intervals; four plots of supports against Prediction Error, Estimates, Normalized Entropy and Precision Error; two plots of GCE reestimation against Prediction and Precision Errors. Note that plots regarding Precision Error are only produced if the argument coef is not NULL.

Usage

```
## S3 method for class 'lmgce'
plot(
  x,
  type = c("ggplot2", "plotly"),
  which = 1:7,
  ci.level = 0.95,
  ci.method = c("z", "percentile", "basic"),
  boot.B = ifelse(x$boot.B == 0, 100, x$boot.B),
  boot.method = x$boot.method,
  coef = NULL,
  OLS = TRUE,
```

```
  NormEnt = TRUE,
 caption = list(paste0("Estimates (", ci.method[1], " ", ci.level * 100, "% CI)"),
    "Prediction Error vs supports", "Estimates vs supports",
    "Normalized Entropy vs supports", "Precision Error vs supports",
   "Prediction Error vs GCE reestimation", "Precision Error vs GCE reestimation"),
  ...
)
```

Arguments

x	Fitted lmgce model object.
type	One of c("ggplot2", "plotly"). "ggplot2" is used by default.
which	A subset of the numbers 1:7.
ci.level	the confidence level (0,1) required to compute the confidence interval.
ci.method	the method used to compute the confidence interval. One of c("z","percentile", "basic"). The default is method = "z".
boot.B	A single positive integer greater or equal to 10 for the number of bootstrap replicates for the computation of the bootstrap confidence interval(s), to be used when method = c("percentile", "basic") and when object was created with boot.B = 0. The default is boot.B = 100 when the object has no previous sampling information and boot.B = object$boot.B otherwise, which corresponds to the boot.B given to lmgce when the object was created.
boot.method	Method used for bootstrapping. One of c("residuals", "cases", "wild") which corresponds to resampling on residuals, on individual cases or on residuals multiplied by a N(0,1) variable, respectively. The default is boot.method = object$boot.method.
coef	A vector of true coefficients to be used when which = c(5,7).
OLS	Boolean value. if TRUE, the default, plots the OLS results.
NormEnt	Boolean value. if TRUE, the default, plots the normalized entropy.
caption	Captions to appear above the plots; character vector or list of valid graphics annotations, see as.graphicsAnnot, of length 7, the j-th entry corresponding to which[j]. Can be set to "" or NA to suppress all captions.
...	additional arguments.

Value

A named list of ggplot or plotly objects, each representing a separate plot.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

See Also

lmgce

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

plot(res_gce_package)
```

`plot.neagging` *Plot Diagnostics for a* `neagging` *Object*

Description

Two plots (selectable by `which`) are currently available to evaluate a `neagging` object: plots of the estimates and in sample error against the number of bootstrap samples aggregated.

Usage

```
## S3 method for class 'neagging'
plot(x, which = 1, ...)
```

Arguments

`x`	Fitted `neagging` model object.
`which`	Numbers 1 or 2.
`...`	additional arguments.

Value

A `ggplot` object.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

See Also

`lmgce`, `tsbootgce` and `neagging`

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

res_neagging <- neagging(res_gce_package)

plot(res_neagging)
```

`plot.ridgetrace` *Plot Diagnostics for a* `ridgetrace` *Object*

Description

Plot Diagnostics for a `ridgetrace` Object

Usage

```
## S3 method for class 'ridgetrace'
plot(x, coef = NULL, log = TRUE, range = TRUE, ...)
```

Arguments

`x`	Fitted `ridgetrace` model object.
`coef`	A vector of true coefficients if available.
`log`	Boolean. If `log = TRUE`, the default, horizontal axis will display the logarithm of lambda.
`range`	Boolean. If `range = TRUE`, the default, minimum and maximum values for each coefficient will be shown.
`...`	additional arguments.

Value

Supports are returned.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

res.ridgetrace <- ridgetrace(formula = y ~ X001 + X002 + X003 + X004, data = dataThesis)

plot(res.ridgetrace)

plot.tsbootgce *Plot Diagnostics for a* tsbootgce *object*

Description

Four plots (selectable by which) are currently available to evaluate a tsbootgce object.

Usage

```
## S3 method for class 'tsbootgce'
plot(
  x,
  which = c(1, 2),
  group = TRUE,
  group.ncol = NULL,
  group.nrow = NULL,
  ci.levels = c(0.9, 0.95, 0.99),
  ci.method = c("hdr", "basic", "percentile"),
  seed = object$seed,
  lambda = 1,
  col = NULL,
  plot.lines = TRUE,
  legend.position = "bottom",
  ...
)
```

Arguments

x	Fitted tsbootgce object.
which	Integers from 1 to 4. The default is which = c(1,2).
group	Boolean value. If group = TRUE, the default, plots are grouped in one image.
group.ncol	Number of columns (see ggarrange). The default is group.ncol = NULL.
group.nrow	Number of rows. (see ggarrange). The default is group.nrow = NULL.
ci.levels	the confidence levels (maximum of 4) required to compute the confidence interval. The default is ci.levels = c(0.90, 0.95, 0.99).
ci.method	One of c("hdr", "basic", "percentile"). The default is ci.method = "hdr" (see hdr).
seed	A single value, interpreted as an integer, for reproducibility or NULL for randomness. The default is seed = object$seed.
lambda	Box-Cox transformation parameter. Value between 0 and 1. The default is lambda = 1 (see hdr).
col	Vector of colors for regions. The default is col = NULL.
plot.lines	Boolean. The default is plot.lines = TRUE.
legend.position	The default is legend.position = "bottom".
...	additional arguments.

Value

A ggplot object.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

See Also

tsbootgce

Examples

```
res.tsbootgce <-
  tsbootgce(
    formula = CO2 ~ 1 + L(EPC, 1) + L(EUS, 2) + L(GDP, 0),
    data = moz_ts)

plot(res.tsbootgce, which = 2, group = TRUE)
```

---

predict.lmgce *Predict method for* lmgce *Linear Model Fits*

Description

Predicted values based on a fitted model lmgce object.

Usage

```
## S3 method for class 'lmgce'
predict(
  object,
  newdata,
  interval = c("none", "confidence"),
  type = c("response", "terms"),
  level = 0.95,
  terms = NULL,
  na.action = na.pass,
  ...
)
```

Arguments

object	Fitted lmgce model object.
newdata	An optional data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
interval	One of c("none", "confidence"). Type of interval calculation. Can be abbreviated.
type	One of c("response", "terms"). Type of prediction (response or model term). Can be abbreviated.
level	Tolerance/confidence level (0,1).
terms	if type = "terms", which terms (default is all terms), a character vector.
na.action	function determining what should be done with missing values in newdata. The default is to predict NA.
...	additional arguments.

Value

predict.lmgce produces a vector of predictions.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)


predict(res_gce_package, dataThesis)
```

---

print.cv.lmgce *Print cv.lmgce object*

Description

Print cv.lmgce object

Usage

```
## S3 method for class 'cv.lmgce'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

x	fitted cv.lmgce object.
digits	significant digits in printout.
...	additional print arguments.

Value

A small summary of a cv.lmgce object is returned.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

---

print.cv.tsbootgce *Print cv.tsbootgce object*

Description

Print cv.tsbootgce object

Usage

```
## S3 method for class 'cv.tsbootgce'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

x	fitted cv.tsbootgce object.
digits	significant digits in printout.
...	additional print arguments.

Value

A small summary of a tsbootgce object is returned.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

print.lmgce *Print a* lmgce *object*

Description

Concise summary of a lmgce object

Usage

```
## S3 method for class 'lmgce'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

x	fitted lmgce object.
digits	significant digits in printout.
...	additional print arguments.

Value

A small summary of a lmgce object is returned.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

res_gce_package
```

print.ridgetrace *Print a* ridgetrace *object*

Description

Concise summary of a ridgetrace object

Usage

```
## S3 method for class 'ridgetrace'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

x	fitted ridgetrace object.
digits	significant digits in printout.
...	additional print arguments.

Value

A small summary of a ridgetrace object is returned.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res.ridgetrace <-
  ridgetrace(
    formula = y ~ X001 + X002 + X003 + X004,
    data = dataThesis)

res.ridgetrace
```

print.summary.lmgce *Print Summary of* lmgce *Model Fits*

Description

print.summary method for class lmgce.

Usage

```
## S3 method for class 'summary.lmgce'
print(
  x,
  digits = max(3L, getOption("digits") - 3L),
  symbolic.cor = x$symbolic.cor,
  signif.stars = getOption("show.signif.stars"),
  ...
)
```

Arguments

x	an object of class summary.lmgce, usually, a result of a call to summary.lmgce.
digits	The number of significant digits to use when printing.
symbolic.cor	Boolean value. if TRUE, print the correlations in a symbolic form (see symnum) rather than as numbers.
signif.stars	Boolean value. if TRUE, 'significance stars' are printed for each coefficient.
...	Further arguments passed to or from other methods.

Value

The function print.summary.lmgce prints the information in a summary.lmgce object.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)


summary(res_gce_package)

summary(res_gce_package, ci.level = 0.90, ci.method = "basic")
```

print.tsbootgce *Print* tsbootgce *object*

Description

Print tsbootgce object

Usage

```
## S3 method for class 'tsbootgce'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

x	fitted tsbootgce object.
digits	significant digits in printout.
...	additional print arguments.

Value

A small summary of a tsbootgce object is returned.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res.tsbootgce <-
  tsbootgce(
    formula = CO2 ~ 1 + L(EPC, 1) + L(EUS, 2) + L(GDP, 0),
    data = moz_ts)

res.tsbootgce
```

resid.lmgce *Extract* lmgce *Model Residuals*

Description

resid is a function which extracts model residuals from lmgce objects.

Usage

```
## S3 method for class 'lmgce'
resid(object, ...)
```

Arguments

object	Fitted lmgce model object.
...	additional arguments.

Value

Returns the residuals from a lmgce object

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

resid(res_gce_package)
```

residuals.lmgce *Extract* lmgce *Model Residuals*

Description

residuals is a function which extracts model residuals from lmgce objects. The abbreviated form resid is an alias for residuals.

Usage

```
## S3 method for class 'lmgce'
residuals(object, ...)
```

Arguments

object	Fitted lmgce model object.
...	additional arguments.

Value

Returns the residuals from a lmgce object

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

residuals(res_gce_package)
```

---

res_gce_package *Example 'lmgce' object*

Description

An example of an object of class 'lmgce' used for demonstration.

Usage

```
res_gce_package
```

Format

An object of class '"lmgce"'.

Source

generated by the package.

`ridgetrace` *Function to obtain the ridge trace and choose the support limits given a formula*

Description

Function to obtain the ridge trace and choose the support limits given a formula

Usage

```
ridgetrace(
  formula,
  data,
  subset,
  na.action,
  offset,
  contrasts = NULL,
  lambda = NULL,
  lambda.min = 10^-3,
  lambda.max = 10^3,
  lambda.n = 100,
  standardize = TRUE,
  penalize.intercept = TRUE,
  errormeasure = c("RMSE", "MSE", "MAE", "MAPE", "sMAPE", "MASE"),
  cv = TRUE,
  cv.nfolds = 5,
  seed = 230676
)
```

Arguments

`formula`	An object of class `formula` (or one that can be coerced to that class): a symbolic description of the model to be fitted.
`data`	A data frame (or object coercible by `as.data.frame` to a data frame) containing the variables in the model.
`subset`	an optional vector specifying a subset of observations to be used in the fitting process.
`na.action`	a function which indicates what should happen when the data contain `NA`s. The default is set by the `na.action` setting of `options`, and is `na.fail` if that is unset. The 'factory-fresh' default is `na.omit`. Another possible value is `NULL`, no action. Value `na.exclude` can be useful.
`offset`	this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be `NULL` or a numeric vector or matrix of extents matching those of the response. One or more `offset` terms can be included in the formula instead or as well, and if more than one are specified their sum is used. See `model.offset`.

contrasts	An optional list. See the contrasts.arg of model.matrix.default.
lambda	Ridge parameter. The default is lambda = NULL and a lambda logarithmic sequence will be computed based on lambda.n, lambda.min and lambda.max. Supplying a lambda sequence overrides this.
lambda.min	Minimum value for the lambda sequence. The default id lambda.min = 10^-3. To be used when lambda = NULL.
lambda.max	Maximum value for the lambda sequence. The default id lambda.max = 10^3. To be used when lambda = NULL.
lambda.n	The number of lambda values. The default is lambda.n = 100. To be used when lambda = NULL.
standardize	Boolean value. If TRUE, the default, then: i) centering is done by subtracting the column means of x and y from their corresponding columns; ii) scaling is done by dividing the (centered) columns of x and y by their standard deviations.
penalize.intercept	Boolean value. If TRUE, the default, the intercept will be penalized when standardize = FALSE.
errormeasure	Loss function (error) to be used for the selection of the support spaces. One of c("RMSE","MSE", "MAE", "MAPE", "sMAPE", "MASE"). The default is errormeasure = "RMSE".
cv	Boolean value. If TRUE the error, errormeasure, will be computed using cross-validation. If FALSE the error will be computed in sample. The default is cv = TRUE.
cv.nfolds	number of folds used for cross-validation when cv = TRUE. The default is cv.nfolds = 5.
seed	A single value, interpreted as an integer, for reproducibility or NULL for randomness. The default is seed = 230676.

Value

An object of class ridgetrace is a list containing at least the following components:

lambda	the lambda sequence used
min.coef	a named vector of coefficients (minimum coefficients)
max.coef	a named vector of coefficients (maximum coefficients)
max.abs.coef	a named vector of coefficients (maximum absolute coefficients)
max.abs.residual	the maximum absolute residual
coef.lambda	a data.frame with the coefficients for each lambda tested
coef.lambda.cv	a list of length cv.nfolds containing data.frames with the coefficients for each lambda tested in each cross-validation fold
error.lambda	a vector with the in sample error
error.lambda.cv	a data.frame with cross-validation errors
call	the matched call

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res.ridgetrace <-
  ridgetrace(
    formula = y ~ X001 + X002 + X003 + X004,
    data = dataThesis)

res.ridgetrace
```

---

scalebackcoef *Scale coefficients back*

Description

Given a vector of scaled (standardized) regression coefficients the function returns the unscaled (in the original scale) regression coefficients

Usage

```
scalebackcoef(X.scaled, y.scaled, betas.scaled, intercept = TRUE)
```

Arguments

X.scaled	A matrix scaled with scale.
y.scaled	A vector scaled with scale.
betas.scaled	A vector of given scaled coefficients.
intercept	logical indicating if intercept is to be calculated

Value

Returns a vector of unscaled numeric regression coefficients.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

summary.lmgce *Summarise a linear regression model via generalized cross entropy fit*

Description

summary method for class lmgce. Function used to produce summary information from a fitted linear regression model via generalized cross entropy as represented by object of class lmgce.

Usage

```
## S3 method for class 'lmgce'
summary(
  object,
  call = TRUE,
  correlation = FALSE,
  symbolic.cor = FALSE,
  ci.level = NULL,
  ci.method = c("z", "percentile", "basic"),
  boot.B = ifelse(object$boot.B == 0, 100, object$boot.B),
  boot.method = object$boot.method,
  ...
)
```

Arguments

object	Fitted lmgce model object.
call	Boolean value. if TRUE, the call used is returned. The default is model = TRUE.
correlation	Boolean value. if TRUE, the correlation matrix of the estimated parameters is returned and printed.
symbolic.cor	Boolean value. if TRUE, print the correlations in a symbolic form (see symnum) rather than as numbers.
ci.level	the confidence level (0,1) required to compute the confidence interval. The default is ci.level = NULL which results in the omission of the confidence interval.
ci.method	method used to compute a confidence interval. One of c("z","percentile", "basic"). The default is ci.method = "z".
boot.B	A single positive integer greater or equal to 10 for the number of bootstrap replicates for the computation of the bootstrap confidence interval(s), to be used when method = c("percentile", "basic") and when object was created with boot.B = 0. The default is boot.B = 100 when the object has no previous sampling information and boot.B = object$boot.B otherwise, which corresponds to the boot.B given to lmgce when the object was created.
boot.method	Method used for bootstrapping. One of c("residuals", "cases", "wild") which corresponds to resampling on residuals, on individual cases or on residuals multiplied by a N(0,1) variable, respectively. The default is boot.method = object$boot.method.

... additional arguments.

Value

The function summary.lmgce computes and returns a list of summary statistics of the fitted lmgce linear model given in object, using the components (list elements) "call" and "terms" from its argument, plus

residuals	the residuals, that is response minus fitted values.
coefficients	a $p \times 4$ matrix, where p is the number of non-aliased coefficients, with columns for the estimated coefficient, its standard error, z-statistic and corresponding (two-sided) p-value. Aliased coefficients are omitted.
support	a $p \times 3$ matrix with columns for the normalized entropy (NormEnt), and lower (LL) and upper (UL) limits for each of the $K + 1$ support spaces.
aliased	named logical vector showing if the original coefficients are aliased.
sigma	the square root of the estimated variance of the random error.
df	degrees of freedom, a 3-vector $(p, n - p)$ the first being the number of non-aliased coefficients, the last being the p minus the number of included individuals n.
r.squared	R^2, 'fraction of variance explained by the model'
adj.r.squared	the above R^2 statistic 'adjusted', penalizing for higher p.
cov.unscaled	a $p \times p$ matrix of covariances of the $\hat{\beta}$
support.stdUL	when applicable, the upper limit of the standardized support chosen, when support.method = "standardized" or the factor used when support.method = "ridge".
support.method	method chosen for the support's limits
nep	the normalized entropy of the model.
nep.cv.mean	the cross-validation normalized entropy of the model.
nep.cv.sd	the standard deviation of the cross-validation normalized entropy of the model.
error	the error measure chosen
error.which	which criterion/standardized/factor support was used
error.measure	the value of the error measure
error.measure.cv.mean	the cross-validation value of the error measure
error.measure.cv.sd	the standard deviation of the cross-validation value of the error measure
correlation	the correlation matrix corresponding to the above cov.unscaled, if correlation = TRUE is specified.
symbolic.cor	(only if correlation = TRUE) The value of the argument symbolic.cor.
na.action	from object, if present there.
ci.method	method used to compute a confidence interval

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)


sm_res_gce_package <- summary(res_gce_package)

str(sm_res_gce_package)

sm_res_gce_package$coefficients
```

---

tsbootgce *Time series bootstrap Cross entropy estimation*

Description

This generic function fits a linear regression model using bootstrapped time series via generalized cross entropy.

Usage

```
tsbootgce(
  formula,
  data,
  subset,
  na.action,
  offset,
  contrasts = NULL,
  trim = 0.05,
  reps = 1000,
  start = NULL,
  end = NULL,
  coef.method = c("median", "mode"),
  cv = TRUE,
  cv.nfolds = 5,
  errormeasure = c("RMSE", "MSE", "MAE", "MAPE", "sMAPE", "MASE"),
  errormeasure.which = {
     if (isTRUE(cv))
         c("1se", "min", "elbow")

    else c("min", "elbow")
 },
  support.method = c("standardized", "ridge"),
```

```
  support.method.ridge.lambda = NULL,
  support.method.ridge.lambda.min = 10^-3,
  support.method.ridge.lambda.max = 10^3,
  support.method.ridge.lambda.n = 100,
  support.method.ridge.standardize = TRUE,
  support.method.ridge.penalize.intercept = TRUE,
  support.method.ridge.symm = TRUE,
  support.method.ridge.maxresid = TRUE,
  support.signal = NULL,
  support.signal.vector = NULL,
  support.signal.vector.min = 0.3,
  support.signal.vector.max = 20,
  support.signal.vector.n = 20,
  support.signal.points = c(1/5, 1/5, 1/5, 1/5, 1/5),
  support.noise = NULL,
  support.noise.points = c(1/3, 1/3, 1/3),
  weight = 0.5,
  twosteps.n = 1,
 method = c("dual.BFGS", "dual.lbfgsb3c", "primal.solnl", "primal.solnp", "dual.CG",
   "dual.L-BFGS-B", "dual.Rcgmin", "dual.bobyqa", "dual.newuoa", "dual.nlminb",
    "dual.nlm", "dual.lbfgs", "dual.optimParallel"),
  caseGLM = c("D", "M", "NM"),
  boot.B = 0,
  boot.method = c("residuals", "cases", "wild"),
  seed = 230676,
  OLS = TRUE,
  verbose = 0
)
```

Arguments

formula	a "formula" describing the linear model to be fit. For details see lm and dynlm.
data	A data.frame (or object coercible by as.data.frame to a data frame) or time series object (e.g., ts or zoo), containing the variables in the model.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
na.action	a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit. Another possible value is NULL, no action. Value na.exclude can be useful.
offset	this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector or matrix of extents matching those of the response. One or more offset terms can be included in the formula instead or as well, and if more than one are specified their sum is used. See model.offset.
contrasts	An optional list. See the contrasts.arg of model.matrix.default.
trim	The trimming proportion (see meboot). The default is trim = 0.05.

reps — The number of replicates to generate (see meboot). The default is reps = 1000.

start — The time of the first observation. Either a single number or a vector of two numbers (the second of which is an integer), which specify a natural time unit and a (1-based) number of samples into the time unit (see ts).

end — The time of the last observation, specified in the same way as start (see ts).

coef.method — Method used to estimate the coefficients. One of c("median", "mode"). The default is coef.method = "median"

cv — Boolean value. If TRUE the error, errormeasure, will be computed using cross-validation. If FALSE the error will be computed in sample. The default is cv = TRUE.

cv.nfolds — number of folds used for cross-validation when cv = TRUE. The default is cv.nfolds = 5 and the smallest value allowable is cv.nfolds = 3.

errormeasure — Loss function (error) to be used for the selection of the support spaces. One of c("RMSE","MSE", "MAE", "MAPE", "sMAPE", "MASE"). The default is errormeasure = "RMSE".

errormeasure.which — Which value of errormeasure to be used for selecting a support space upper limit from support.signal.vector. One of c("min", "1se", "elbow") where "min" corresponds to the support spaces that produced the lowest error, "1se" corresponds to the support spaces such that error is within 1 standard error of the CV error for "min" and "elbow" corresponds to the elbow point of the error curve (the point that maximizes the distance between each observation, i.e, the pair composed by the upper limit of the support space and the error, and the line between the first and last observations, i.e., the lowest and the highest upper limits of the support space respectively. See find_curve_elbow). The default is errormeasure.which = "1se".

support.method — One of c("standardized", "ridge"). If support.method = "standardized}, the default, standardized coefficients are used to define the signal support spaces. If \code{support.method = "ridge the signal support spaces are define by the ridge trace.

support.method.ridge.lambda — Ridge parameter. The default is support.method.ridge.lambda = NULL and a lambda logarithmic sequence will be computed based on support.method.ridge.lambda.n, support.method.ridge.lambda.min and support.method.ridge.lambda.max. Supplying a lambda sequence overrides this. To be used when support.method = "ridge".

support.method.ridge.lambda.min — Minimum value for the support.method.ridge.lambda sequence. The default is support.method.ridge.lambda.min = 10^-3. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.lambda.max — Maximum value for the support.method.ridge.lambda sequence. The default is support.method.ridge.lambda.max = 10^3. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.lambda.n
: The number of ridge parameters values. The default is support.method.ridge.lambda.n = 100. To be used when support.method = "ridge" and support.method.ridge.lambda = NULL.

support.method.ridge.standardize
: Boolean value. If TRUE, the default, then: i) centering is done by subtracting the column means of x and y from their corresponding columns; ii) scaling is done by dividing the (centered) columns of x and y by their standard deviations. To be used when support.method = "ridge".

support.method.ridge.penalize.intercept
: Boolean value. if TRUE, the default, the intercept will be penalized. To be used when support.method = "ridge" and support.method.ridge.standardize = FALSE.

support.method.ridge.symm
: Boolean value. If TRUE, the default, signal supports will be symmetrical and the upper limit will be the maximum absolute values of the estimated ridge coefficients for support.method.ridge.lambda. If FALSE, the lower and upper limits will be, respectively, the minimum and maximum values of the estimated ridge coefficients.

support.method.ridge.maxresid
: Boolean value. if TRUE, the default, noise supports will symmetrical and the upper limit will be the maximum absolute value of the residuals of ridge estimation for support.method.ridge.lambda. If FALSE limits are computed using the empirical three-sigma rule (Pukelsheim (1994)).

support.signal
: NULL or fixed positive upper limit (L) for the support spaces (-L,L) on standardized data (when support.method = "standardized"); NULL or fixed positive factor to be multiplied by the maximum absolute value of the ridge trace for each coefficient (when support.method = "ridge"); a pair (LL,UL) or a matrix ((k+1) x 2) for the support spaces on original data. The default is support.signal = NULL.

support.signal.vector
: NULL or a vector of positive values when support.signal = NULL. If support.signal.vector = NULL, the default, a vector c(support.signal.vector.min,...,support.signal.vector.max) of dimension support.signal.vector.n and logarithmically equally spaced will be generated. Each value represents the upper limits for the standardized support spaces, when support.method = "standardized" or the factor to be multiplied by the maximum absolute value of the ridge trace for each coefficient, when support.method = "ridge".

support.signal.vector.min
: A positive value for the lowest limit of the support.signal.vector when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.min = 0.3.

support.signal.vector.max
: A positive value for the highest limit of the support.signal.vector when support.signal = NULL and support.signal.vector = NULL. The default is support.signal.vector.max = 20.

support.signal.vector.n
: A positive integer for the number of support spaces to be used when support.signal

= NULL and support.signal.vector = NULL. The default is support.signal.vector.n = 20.

support.signal.points
: A positive integer, a vector or a matrix. Prior weights for the signal. If not a positive integer then the sum of weights by row must be equal to 1. The default is support.signal.points = c(1 / 5, 1 / 5, 1 / 5, 1 / 5, 1 / 5).

support.noise
: An interval, preferably centered around zero, given in the form c(LL,UL). If support.noise = NULL, the default, then a vector c(-L,L) is computed using the empirical three-sigma rule (Pukelsheim (1994)).

support.noise.points
: A positive integer, a vector or a matrix. Prior weights for the noise. If not a positive integer then the sum of weights by row must be equal to 1. The default is support.noise.points = c(1 / 3, 1 / 3, 1 / 3).

weight
: a value between zero and one representing the prediction-precision loss trade-off. If weight = 0.5, the default, equal weight is placed on the signal and noise entropies. A higher than 0.5 value places more weight on the noise entropy whereas a lower than 0.5 value places more weight on the signal entropy.

twosteps.n
: Number of GCE reestimations using a previously estimated vector of signal probabilities.

method
: Use "primal.solnl" (GCE using Sequential Quadratic Programming (SQP) method; see solnl) or "primal.solnp" (GCE using the augmented Lagrange multiplier method with an SQP interior algorithm; see solnp) for primal form of the optimization problem and "dual.CG" (GCE using a conjugate gradients method; see optim), "dual.BFGS" (GCE using Broyden-Fletcher-Goldfarb-Shanno quasi-Newton method; see optim), "dual.L-BFGS-B" (GCE using a box-constrained optimization with limited-memory modification of the BFGS quasi-Newton method; see optim), dual.Rcgmin (GCE using an update of the conjugate gradient algorithm; see optimx), dual.bobyqa (GCE using a derivative-free optimization by quadratic approximation; see optimx and bobyqa), dual.newuoa (GCE using a derivative-free optimization by quadratic approximation; see optimx and newuoa), dual.nlminb (GCE; see optimx and nlminb), dual.nlm (GCE; see optimx and nlm), dual.lbfgs (GCE using the Limited-memory Broyden-Fletcher-Goldfarb-Shanno; see lbfgs), dual.lbfgsb3c (GCE using L-BFSC-B implemented in Fortran code and with an Rcpp interface; see lbfgsb3c) or dual.optimParallel (GCE using parallel version of the L-BFGS-B; see optimParallel) for dual form. The default is method = "dual.BFGS".

caseGLM
: special cases of the generic general linear model. One of c("D", "M", "NM"), where "D" stands for data, "M" for moment and "NM" for normed-moment The default is caseGLM = "D".

boot.B
: A single positive integer greater or equal to 10 for the number of bootstrap replicates to be used for the computation of the bootstrap confidence interval(s). Zero value will generate no replicate. The default is boot.B = 0.

boot.method
: Method to be use for bootstrapping. One of c("residuals", "cases", "wild") which corresponds to resampling on residuals, on individual cases or on residuals multiplied by a N(0,1) variable, respectively. The default is boot.method = "residuals".

seed	A single value, interpreted as an integer, for reproducibility or NULL for randomness. The default is seed = 230676.
OLS	Boolean value. if TRUE, the default, OLS estimation is performed.
verbose	An integer to control how verbose the output is. For a value of 0 no messages or output are shown and for a value of 3 all messages are shown. The default is verbose = 0.

Details

The tsbootgce function fits several linear regression models via generalized cross entropy in replicas of time series obtained using meboot. Models for tsbootgce are specified symbolically (see lm and dynlm).

Value

tsbootgce returns an object of class tsbootgce. The generic accessory functions coef.tsbootgce, confint.tsbootgce and plot.tsbootgce extract various useful features of the value returned by object of class tsbootgce.

An object of class tsbootgce is a list containing at least the following components:

call	the matched call.
coefficients	a named data frame of coefficients determined by coef.method.
data.ts	ts object.
error	loss function (error) used for the selection of the support spaces.
error.measure	in sample error for the selected support space.
fitted.values	the fitted mean values.
frequency	see link[zoo]{zoo}.
index	see link[zoo]{zoo}.
lmgce	lmgce object.
meboot	meboot replicates.
model	the model frame used.
nep	normalized entropy of the signal of the model.
nepk	normalized entropy of the signal of each coefficient.
residuals	the residuals, that is response minus fitted values.
results	a list containing the bootstrap results: "coef.matrix", a named data frame of all the coefficients; "nepk.matrix", a named data frame of all the normalized entropy values of each parameter; "nep.vector", a vector of all the normalized entropy values of the model.
seed	the seed used.
terms	the terms object used.
x	if requested (the default), the model matrix used.
xlevels	(only where relevant) a record of the levels of the factors used in fitting.
y	if requested (the default), the response used.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

References

Golan, A., Judge, G. G. and Miller, D. (1996) *Maximum entropy econometrics : robust estimation with limited data.* Wiley.

Golan, A. (2008) *Information and Entropy Econometrics — A Review and Synthesis.* Foundations and Trends® in Econometrics, 2(1–2), 1–145. doi:10.1561/0800000004

Golan, A. (2017) *Foundations of Info-Metrics: Modeling, Inference, and Imperfect Information (Vol. 1).* Oxford University Press. doi:10.1093/oso/9780199349524.001.0001

Hyndman, R.J. (1996) *Computing and graphing highest density regions.* American Statistician, 50, 120-126. doi:10.2307/2684423

Pukelsheim, F. (1994) *The Three Sigma Rule.* The American Statistician, 48(2), 88–91. doi:10.2307/2684253

Vinod, H. D., & Lopez-de-Lacalle, J. (2009). *Maximum Entropy Bootstrap for Time Series: The meboot R Package.* Journal of Statistical Software, 29(5), 1–19. doi:10.18637/jss.v029.i05

See Also

The generic functions `plot.tsbootgce`, `print.tsbootgce`, and `coef.tsbootgce`.

Examples

```
res.tsbootgce <-
  tsbootgce(
    formula = CO2 ~ 1 + L(EPC, 1) + L(EUS, 2) + L(GDP, 0),
    data = moz_ts)

res.tsbootgce
```

`variable.names.lmgce` *Variable Names of* `lmgce` *Fitted Models*

Description

Simple utility returning variable names.

Usage

```
## S3 method for class 'lmgce'
variable.names(object, ...)
```

Arguments

object	Fitted lmgce model object.
...	additional arguments.

Value

A character vector containing the names of the variables in the lmgce model object.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)


variable.names(res_gce_package)
```

---

vcov.lmgce *Extract lmgce Model's Variance-Covariance Matrix*

Description

Returns the variance-covariance matrix of the main parameters of a lmgce object

Usage

```
## S3 method for class 'lmgce'
vcov(object, ...)
```

Arguments

object	Fitted lmgce model object.
...	additional arguments.

Value

A matrix of the estimated covariances between the parameter estimates in the linear predictor of the lmgce model.

Author(s)

Jorge Cabral, <jorgecabral@ua.pt>

Examples

```
res_gce_package <-
  lmgce(y ~ .,
        data = dataThesis,
        twosteps.n = 1,
        boot.B = 100,
        seed = 230676)

vcov(res_gce_package)
```

Index