Package 'eDMA'

October 13, 2022

Type Package

Title Dynamic Model Averaging with Grid Search

Version 1.5-3

Author Leopoldo Catania [aut,cre], Nima Nonejad [aut]

Maintainer Leopoldo Catania <leopoldo.catania@econ.au.dk>

Description Perform dynamic model averaging with grid search as in Dangl and Halling (2012) <doi:10.1016/j.jfineco.2012.04.003> using parallel computing.

SystemRequirements Requires the OpenMP library for parallel computing. If the OpenMP library is not available, the code is executed sequentially and a warning is printed.

License GPL (>= 2)

LazyData TRUE

Imports Rcpp (>= 0.12.5)

LinkingTo Rcpp,RcppArmadillo

Depends zoo,xts,methods,parallel

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-08-27 09:54:27 UTC

R topics documented:

`eDMA-package` *Dynamic Model Averaging with Modifications*

Description

Perform Dynamic Model Averaging with modifications introduced in Dangl and Halling (2012) using parallel computing.

Details

Package:	eDMA
Type:	Package
Version:	1.5-3
Date:	2018-27-08
License:	GPL (>= 2)

Author(s)

Leopoldo Catania & Nima Nonejad

Maintainer: Leopoldo Catania <leopoldo.catania@uniroma2.it>

References

Raftery, Adrian E., Miroslav Karny, and Pavel Ettler. "Online prediction under model uncertainty via dynamic model averaging: Application to a cold rolling mill." Technometrics 52.1 (2010): 52-66.

Catania, Leopoldo, and Nima Nonejad (2018). "Dynamic Model Averaging for Practitioners in Economics and Finance: The eDMA Package." Journal of Statistical Software, 84(11), 1-39. doi: 10.18637/jss.v084.i11.

Dangl, Thomas, and Michael Halling. "Predictive regressions with time-varying coefficients." Journal of Financial Economics 106.1 (2012): 157-181.

Raftery, Adrian E., David Madigan, and Jennifer A. Hoeting. "Bayesian model averaging for linear regression models." Journal of the American Statistical Association 92.437 (1997): 179-191.

Harrison, Jeff, and Mike West. Bayesian Forecasting & Dynamic Models. Springer, 1999.

See Also

DMA

Examples

```
library(eDMA)

## load data
data("USData")

## do DMA, keep the first three predictors fixed and the intercept
Fit <- DMA(GDPDEF ~ Lag(GDPDEF, 1) + Lag(GDPDEF, 2) + Lag(GDPDEF, 3) +
            Lag(ROUTP, 1) + Lag(UNEMP, 1), data = USData, vDelta = c(0.9,0.95,0.99),
             vKeep = c(1, 2, 3, 4))
```

BacktestDMA — *Backtest measures for Dynamic Model Averaging and comparison with Dynamic Model Selection*

Description

Backtest measures for Dynamic Model Averaging and comparison with Dynamic Model Selection. This function evaluates the out of sample performance of DMA and compare it with DMS.

Usage

```
BacktestDMA(object, iBurnPeriod = NULL)
```

Arguments

`object`	an object of the class DMA-class, created using the function DMA.
`iBurnPeriod`	An integer indicating the length of the burn-in period. By default `iBurnPeriod = NULL`. If `iBurnPeriod = NULL` then one third of the total sample is used as the burn-in in period and a warning is printed.

Details

The function returns a `matrix` with Mean Squared Error (MSE), Mean Absolute Error (MAD) and Predictive Likelihood for DMA and DMS using the predictions during the out-of-sample period.

Value

An object of the class `matrix`.

Author(s)

Leopoldo Catania & Nima Nonejad

Examples

```
library(eDMA)

## load data
data("USData")

## do DMA, keep the first three predictors fixed and the intercept
Fit <- DMA(GDPDEF ~ Lag(GDPDEF, 1) + Lag(GDPDEF, 2) + Lag(GDPDEF, 3) +
           Lag(ROUTP, 1) + Lag(UNEMP, 1), data = USData, vDelta = c(0.9, 0.95, 0.99),
           vKeep = c(1, 2, 3))

BacktestDMA(Fit, iBurnPeriod = 32)
```

DMA *Perform Dynamic Model Averaging*

Description

Implements the Dynamic Model Averaging procedure with the possibility of also performing averaging over a grid of foregetting factor values

Usage

```
DMA(formula, data, vDelta = c(0.9, 0.95, 0.99), dAlpha = 0.99,
    vKeep = NULL, bZellnerPrior = FALSE, dG = 100, bParallelize = TRUE,
    iCores = NULL, dBeta = 1.0)
```

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`	an object of the class data.frame, (or object coercible by as.data.frame to a data frame) containing the variables in the model. It can also be an object of the classes `ts`, `xts` or `zoo`. If this is the case, the time information is used in the graphical representation of the results. The last observation of the dependent variable can be equal to `NA`. This is the case when a series of length T is available but the user wants to have predictions for time $T+1$, see Details.
`vDelta`	D x 1 numeric vector representing the the grid of values of δ. By default `vDelta = c(0.9, 0.95, 0.99)`.
`dAlpha`	numeric variable representing α. By default `dAlpha = 0.99`.
`vKeep`	`numeric` vector of indices representing the predictors that must be always included in the models. The combinations of predictors that do not include the variables declared in `vKeep` are automatically discarded. The indices must be consistent with the model description given in `formula`, i.e., if the first and the

fourth variables always have to be included, then we must set vKeep=c(1, 4). Note that, the intercept (if not removed from `formula`) is always in the first position. It can also be a character vector indicating the names of the predictors if these are consistent with the provided `formula`. If `vKeep = "KS"` the "Kitchen Sink" formulation is adopted, i.e., all the predictors are always included, see, e.g., Paye (2012). By default all the combinations are considered, i.e. `vKeep = NULL`.

`bZellnerPrior` Boolean variable indicating whether the Zellner prior should be used on the coefficients at time t=0. Default = `FALSE`.

`dG` numeric variable equal to 100 by default. If `bZellnerPrior = TRUE` this represent the variable 'g' in Eq. (4) of Dangl Halling (2012). Otherwise, if `bZellnerPrior = FALSE` it represents the scaling factor for the variance covariance matrix of the normal prior for θ_0, i.e. θ_0~N(0,dG*I) where I is the identity matrix.

`bParallelize` Boolean variable indicating whether to use multiple processors to speed up the computations. By default `bParallelize = TRUE`.

`iCores` integer indicating the number of cores to use if `bParallelize = TRUE`. By default all but one cores are used. The number of cores is guessed using the `detectCores()` function from the `parallel` package.

`dBeta` integer indicating the forgetting factor for the measurement variance, see Prado and West (2010) pp. 132 and Beckmann and Schussler (2014).

Details

There might be situations when the practitioner desires to predict $T + 1$ conditional on observation till time T in a true out-of-sample fashion. In such circumstances the user can substitute the future value of the dependent variable with an `NA`. This way, the code treats the last observation as missing and does not perform backtesting or updating of the coefficients. However, the filter provides us with the necessary quantities to perform prediction. The predicted value $\hat{y_{T+1}} = E[y_T + 1|F_T]$ as well as the predicted variance decomposition can then be extracted using the getLastForecast method. The other quantities that can be extracted, for example via the as.data.frame method, will ignore the presence of the last `NA` and report results only for the fist T observations.

See Catania and Nonejad (2016) for further details.

Value

An object of the class `DMA`, see DMA-class.

Author(s)

Leopoldo Catania & Nima Nonejad

References

Beckmann, J., & Schussler, R. (2014). Forecasting Equity Premia using Bayesian Dynamic Model Averaging (No. 2914). Center for Quantitative Economics (CQE), University of Muenster.

Dangl, T., & Halling, M. (2012). Predictive regressions with time–varying coefficients. *Journal of Financial Economics*, **106(1)**, 157–181. doi: 10.1016/j.jfineco.2012.04.003.

Catania, Leopoldo, and Nima Nonejad (2018). "Dynamic Model Averaging for Practitioners in Economics and Finance: The eDMA Package." Journal of Statistical Software, 84(11), 1-39. doi: 10.18637/jss.v084.i11.

Paye, B.S. (2012). 'Deja vol': Predictive Regressions for Aggregate Stock Market Volatility Using Macroeconomic Variables.*Journal of Financial Economics*, **106(3)**, 527-546. ISSN 0304-405X. doi: 10.1016/j.jfineco.2012.06.005. URL `http://www.sciencedirect.com/science/article/pii/S0304405X12001316`.

Prado, R., & West, M. (2010). Time series: modeling, computation, and inference. CRC Press.

Examples

```
#  Code chunk of Catania and Nonejad (2016) Fast Dynamic Model Averaging
#  for Practitioners in Economics and Finance: The eDMA Package
library(eDMA)

## load data
data("USData")

## do DMA, keep the first three predictors fixed and the intercept
Fit <- DMA(GDPDEF ~ Lag(GDPDEF, 1) + Lag(GDPDEF, 2) + Lag(GDPDEF, 3) +
            Lag(ROUTP, 1) + Lag(UNEMP, 1), data = USData, vDelta = c(0.9,0.95,0.99),
             vKeep = c(1, 2, 3, 4))

Fit
```

`DMA-class` *class: Class for the DMA class*

Description

Class for the DMA estimate.

Objects from the Class

A virtual Class: No objects may be created from it.

Slots

`model`**:** Object of class `"list"` Contains information about the DMA specification.

`data`**:** Object of class `"list"` Contains the data given to the `DMA` function.

`Est`**:** Object of class `"list"` Contains the estimated quantities.

Methods

as.data.frame `signature(object = "DMA")`: Extracts estimated quantities, (see note).

plot `signature(x = "DMA", y = "missing")`: Plots estimated quantities.

show `signature(object = "DMA")`.

summary `signature(object = "DMA")`: Print a summary of the estimated model. This method accepts the additional argument `iBurnPeriod` corresponding at the length of the burn-in period. By default `iBurnPeriod = NULL`, i.e. all the sample is used.

coef `signature(object = "DMA")`: Extract the filtered regressor coefficients. This method accepts the additional argument `iBurnPeriod` corresponding at the length of the burn-in period. By default `iBurnPeriod = NULL`, i.e. all the sample is used.

residuals `signature(object = "DMA")`: Extract the residuals of the model. This method accepts the additional argument `iBurnPeriod` corresponding at the length of the burn-in period. By default `iBurnPeriod = NULL`, i.e. all the sample is used. The additional Boolean argument `standardize` controls if standardize residuals should be returned. By default `standardize = FALSE`. The additional argument `type` permits to choose between residuals evaluated using DMA or DMS. By default `type = "DMA"`.

inclusion.prob `signature(object = "DMA")`: Extract the inclusion probabilities of the regressors. This method accepts the additional argument `iBurnPeriod` corresponding at the length of the burn-in period. By default `iBurnPeriod = NULL`, i.e. all the sample is used.

pred.like `signature(object = "DMA")`: Extract the predictive log-likelihood series. This method accepts the additional argument `iBurnPeriod` corresponding at the length of the burn-in period. By default `iBurnPeriod = NULL`, i.e. all the sample is used. The additional argument `type` permits to choose between predictive likelihood evaluated using DMA or DMS. By default `type = "DMA"`.

getLastForecast `signature(object = "DMA")`: If the last observation of the dependent variable was `NA`, i.e. the practitioner desidered to predict Y_{T+1} having a sample of length T (without backtesting the result), this method can be used to extract the predicted value $\hat{y_T + 1} = E[y_{T+1}|F_T]$ as well as the predicted variance decomposition according to Equation (12) of Catania and Nonejad (2016).

Note

The `as.data.frame()` method permits to extract several estimated quantities. It accepts the two additional arguments: `which` with possible values:

- `mincpmt`: Posterior inclusion probabilities of the predictors.
- `vsize`: Expected number of predictors (average size).
- `mtheta`: Filtered estimates of the regression coefficients.
- `mpmt`: Posterior probability of the degree of instability.
- `vyhat`: Point forecasts.
- `vLpdfhat`: Predictive log-likelihood.
- `vdeltahat`: Posterior weighted average of delta.
- `mvdec`: representing the y_t variance decomposition. The function returns a T x 5 matrix whose columns contains the variables.

  - vtotal: total variance.
  - vobs: Observational variance.
  - vcoeff: Variance due to errors in the estimation of the coefficients.
  - vmod: Variance due to model uncertainty.
  - vtvp: Variance due to uncertainty with respect to the choice of the degrees of time–variation in the regression coefficients.
- vhighmp_DMS: Highest posterior model probability.
- vhighmpTop01_DMS: Sum of the 10% highest posterior model probabilities.

and iBurnPeriod which is an integer indicating the length of the burn-in period. For instance, if iBurnPeriod = 50 the first 50 observations are removed from the output. By default iBurnPeriod = NULL meaning that all the observations are returned.

Author(s)

Leopoldo Catania & Nima Nonejad

References

Catania, Leopoldo, and Nima Nonejad (2018). "Dynamic Model Averaging for Practitioners in Economics and Finance: The eDMA Package." Journal of Statistical Software, 84(11), 1-39. doi: 10.18637/jss.v084.i11.

Lag — *Lag a vector or a matrix of observations*

Description

Lag a vector or a matrix of observations by iK periods.

Usage

```
Lag(mY, iK)
```

Arguments

mY	a vector or a matrix of observations.
iK	An integer indicating the number of lag.

Details

The function returns a numeric vector or a matrix depending on the class of mY. The dimension of the object is preserved and NA's are introduced for the missing values.

Value

An object of the class numeric or matrix.

Author(s)

Leopoldo Catania & Nima Nonejad

Examples

```
#  Code chunk of Catania and Nonejad (2016) Dynamic Model Averaging
#  for Practitioners in Economics and Finance: The eDMA Package
library(eDMA)

## load data
data("USData")

UDData_lagged <- Lag(USData, 1)
```

PowerSet *Build the power set of the values {0,1,2,...,*`iK`*}.*

Description

Build the power set of the values {0,1,2,...,`iK`}.

Usage

```
PowerSet(iK)
```

Arguments

`iK`	`numeric` an integer value indicating the end of the series {0,1,2,...,`iK`}.

Details

The function returns a `list` of `numeric` vectors with the indices representing all the 2^iK subsets. The empty subset { } is represented by the `numeric(0)` vector.

Value

, 4 An object of the class `list`.

Author(s)

Leopoldo Catania & Nima Nonejad

Examples

```
library(eDMA)
PowerSet(5)
```

SimData *data: Simulated data from DLM of West and Harrison (1999).*

Description

This is the simulated dataset used in Section 4.1 of Catania and Nonejad (2016).

Usage

```
data(USData)
```

Format

A matrix object containing 500 x 6 simulated observations.

References

Catania, Leopoldo, and Nima Nonejad (2018). "Dynamic Model Averaging for Practitioners in Economics and Finance: The eDMA Package." Journal of Statistical Software, 84(11), 1-39. doi: 10.18637/jss.v084.i11.

West, Mike. Bayesian forecasting. John Wiley & Sons, Inc., 1999.

Examples

```
#the data set has been generated as:

set.seed(7892)

iT <- 500
iK <- 3

dV <- 0.1
mW <- diag(iK + 1) * 0.01
dPhi <- 1

vBeta0 <- rep(0, iK + 1)
mX <- cbind(1, matrix(rnorm(iT * (iK)), iT, iK))

lOut <- SimulateDLM(iT, mX, vBeta0, mW, dV, dPhi)
vY <- lOut$vY
```

```
mX <- mX[, -1]

iK_Add <- 2

mX_add <- matrix(rnorm(iT * iK_Add), iT, iK_Add)

SimData <- cbind(y = vY, mX, mX_add)
colnames(SimData) <- c("y", paste("x", 2:(iK + iK_Add + 1), sep = ""))
```

`SimulateDLM` *Simulate from DLM of West and Harrison (1999).*

Description

Simulate from DLM of West and Harrison (1999), as in Section 2 of Catania and Nonejad (2016).

Usage

```
SimulateDLM(iT, mX, vBeta0, mW, dV, dPhi)
```

Arguments

`iT`	`numeric`, number of observation to simulate.
`mX`	`matrix` of dimension `iT` x N m where N is the number of covariates.
`vBeta0`	`numeric` vector with initial value for the regressor coefficients.
`mW`	`matrix` covariance matrix of the state equation.
`dV`	`numeric` variance of the observation (measurement equation).
`dPhi`	`numeric` value for the autoregressive parameter of the regressors. It imposes that all the regressors have the same autoregressive parameters, if `dPhi = 1`, then the regressors evolve as random-walks.

Details

The function returns a `list` of two elements: `vY` and `mBeta`. `vY` is a `iT` x 1 `numeric` vector of simulated dependent variables. `mBeta` is a `matrix` of dimension `iT` x `ncol(mX)` of regressor coefficients.

Value

An object of the class `list`.

Author(s)

Leopoldo Catania & Nima Nonejad

References

Catania, Leopoldo, and Nima Nonejad (2018). "Dynamic Model Averaging for Practitioners in Economics and Finance: The eDMA Package." Journal of Statistical Software, 84(11), 1-39. doi: 10.18637/jss.v084.i11.

West, Mike. Bayesian forecasting. John Wiley & Sons, Inc., 1999.

Examples

```
set.seed(7892)

iT <- 500
iK <- 3

dV <- 0.1
mW <- diag(iK + 1) * 0.01
dPhi <- 1

vBeta0 <- rep(0, iK + 1)
mX <- cbind(1, matrix(rnorm(iT * (iK)), iT, iK))

lOut <- SimulateDLM(iT, mX, vBeta0, mW, dV, dPhi)
vY <- lOut$vY
```

`USData` *data: Quarterly US inflation and associated predictors*

Description

This is the dataset used in Groen et al. (2013) and is downloadable from `http://www.tandfonline.com/doi/suppl/10.1080/07350015.2012.727718`.
The variables are:

GDPDEF : Quarterly log changes in the Gross Domestic Product implicit price deflator.
ROUTP : Real GDP in volume terms.
RCONS : Real durable personal consumption expenditures in volume terms
RINVR : Real residential investment in volume terms
PIMP : Import deflator
UNEMP : Unemployment ratio
NFPR : Non-farm payrolls data on employment
HSTS : Housing starts
OIL : Real spot price of oil
FOOD : Real food commodities price index
RAW : Real raw material commodities price index
M2 : The M2 monetary aggregate
YL : The level factor

TS : The slope factor
CS : Curvature factor
MS : One-year ahead inflation expectations that come from the Reuters/Michigan Survey of Consumers.

Usage

```
data(USData)
```

Format

A xts object containing 206x16 observations from from 1960-01-01 to 2011-04-01.

References

Groen, Jan JJ, Richard Paap, and Francesco Ravazzolo. (2013) Real–time inflation forecasting in a changing world. *Journal of Business & Economic Statistics*, **31.1** : 29–44.

`USRecessions` *data: Dates of U.S. recessions as inferred by GDP-based recession indicator (JHDUSRGDPBR).*

Description

Dates of U.S. recessions as inferred by GDP-based recession indicator (JHDUSRGDPBR) downloaded from FRED. `https://fred.stlouisfed.org/series/JHDUSRGDPBR#0`.

From the FRED website:

The series assigns dates to U.S. recessions based on a mathematical model of the way that recessions differ from expansions. Whereas the NBER business cycle dates are based on a subjective assessment of a variety of indicators, the dates here are entirely mechanical and are calculated solely from historically reported GDP data. Whenever the GDP-based recession indicator index rises above 67%, the economy is determined to be in a recession. The date that the recession is determined to have begun is the first quarter prior to that date for which the inference from the mathematical model using all data available at that date would have been above 50%. The next time the GDP-based recession indicator index falls below 33%, the recession is determined to be over, and the last quarter of the recession is the first quarter for which the inference from the mathematical model using all available data at that date would have been below 50%.

Usage

```
data(USData)
```

Format

A zoo object containing 206 observations at quarterly frequency from 1960-Q1 to 2011-Q4.

References

Hamilton, James, Dates of U.S. recessions as inferred by GDP-based recession indicator [JHDUSRGDPBR], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/JHDUSRGDPBR, February 4, 2018.

Examples

```
data("USRecessions")
```

Index