Package ‘dynmix’

May 16, 2025

Type Package

Title Estimation of Dynamic Finite Mixtures

Version 2.2

Imports graphics, MASS, Rcpp, stats, utils, zoo

LinkingTo Rcpp, RcppArmadillo

Date 2025-05-15

Author Krzysztof Drachal [aut, cre] (Faculty of Economic Sciences, University of Warsaw, Poland)

Maintainer Krzysztof Drachal <kdrachal@wne.uw.edu.pl>

Description Allows to perform the dynamic mixture estimation with state-space components and normal regression components, and clustering with normal mixture. Quasi-Bayesian estimation, as well as, that based on the Kerridge inaccuracy approximation are implemented. Main references: Nagy and Suzdaleva (2013) <doi:10.1016/j.apm.2013.05.038>; Nagy et al. (2011) <doi:10.1002/acs.1239>.

License GPL-3

LazyData TRUE

URL `https://CRAN.R-project.org/package=dynmix`

Note Ver. 1.0: The author obtained financial resources as part of financing the doctoral scholarship from the Polish National Science Centre (DEC-2018/28/T/HS4/00095). Ver. 2.0: The research was supported by the program "Excellence Initiative - Research University (IDUB)" through the project BOB-IDUB-622-89/2021 - Nowe Idee POB III IDUB, 01/IDUB/2019/94 at the University of Warsaw, Poland.

NeedsCompilation yes

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2025-05-16 08:50:20 UTC

Contents

`cauimp` *Computes Causal Inference through Counterfactual Predictions from a Mixture Estimation with State-Space Components.*

Description

This function estimates causal inference through counterfactual predictions from a mixture estimation with state-space components. Multi-step ahead predictions are generated by the Monte Carlo method.

Usage

```
cauimp(object,x.post,y.post,alpha=0.05,n.sim=100)
```

Arguments

`object`	object of class `mixest` obtained from `mixest1` for the pre-intervention period
`x.post`	`matrix` of independent time-series (predictors) for the post-intervention period, observations inserted rowwise
`y.post`	one column `matrix` of the post-intervention period observed dependent time-series, observations inserted rowwise
`alpha`	optional, `numeric` between 0 and 1, the desired tail area probability for posterior intervals, by default `alpha=0.05` is taken
`n.sim`	optional, `numeric`, number of the post-intervention period simulations, by default `n.sim=100` is taken

Value

list of

$statistics	matrix of summary statistics for the post-intervention period
$significance	logical indicating whether the posterior interval excludes zero
$p	numeric of Bayesian one-sided tail area probability that the observed effect was obtained by chance
$y.hat	vector of the dependent variable predicted for the post-intervention period
$alpha	numeric of the desired tail area probability for posterior intervals, as above
$n.sim	numeric of the number of the post-intervention period simulations, as above
$y.sim	matrix of the simulated dependent variable predictions for the post-intervention period

References

Brodersen, K. H., Gallusser, F., Koehler, J., Remy, N., Scott, S. L., 2015, Inferring causal impact using Bayesian structural time-series models. *Annals of Applied Statistics* **9**, 247–274.

Morgan, S. L., Winship, C., 2007, *Counterfactuals and Causal Inference*, Cambridge University Press.

See Also

mixest1, CausalImpact

Examples

```
data(oil)
m1 <- mixest1(y=oil[1:300,1,drop=FALSE],x=oil[1:300,-1,drop=FALSE],ftype=0,V=1,W=1,kappa=0.97)
x.1 <- oil[301:321,-1,drop=FALSE]
y.1 <- oil[301:321,1,drop=FALSE]
ci <- cauimp(object=m1,x.post=x.1,y.post=y.1,alpha=0.05,n.sim=100)
```

---

convts — *Renames Selected Outcomes of* mixest *and* tvpreg *Objects.*

Description

This function renames rows of selected outcomes stored in mixest and tvpreg objects. It can be useful in generating better looking plots.

Usage

```
convts(x,ind=NULL, ...)
```

Arguments

x	object of class mixest or tvpreg
ind	optional, character consisting of names of time points, should have the same length as the forecasted time-series
...	optional, alternatively, instead of providing ind, arguments of seq.Date can be specified

Value

object of the same class as x but with renamed rownames of selected outcomes

Examples

```
data(oil)
t1 <- tvp.reg(y=oil[,1,drop=FALSE],x=oil[,-1,drop=FALSE],lambda=0.99,V=100,W=100)
plot(t1)

t1a <- convts(x=t1,from=as.Date("1998-02-01"),by="month",length.out=nrow(oil[,1,drop=FALSE]))
plot(t1a)

m1 <- mixest1(y=oil[,1,drop=FALSE],x=oil[,-1,drop=FALSE],ftype=1,V=100,W=100)
plot(m1)

ind <- as.character(seq(from=as.Date("1998-02-01"),by="month",length.out=nrow(oil[,1,drop=FALSE])))
m1a <- convts(x=m1,ind=ind)
plot(m1a)
```

---

ldlt *Computes LDL' Matrix Decomposition.*

Description

This function decomposes matrix V into $V = LDL'$, where L is a lower triangular matrix with unit diagonal and D is a diagonal matrix with non-negative terms.

Usage

```
ldlt(A)
```

Arguments

A	symmetric positive-definite matrix

Value

list of

$L	matrix L
$D	matrix D

References

Zhuang, X., 2020, *Lecture Notes in Numerical Analysis (MATH 381)*, University of Alberta.

Examples

```
A <- matrix(c(5,1,1,3),2,2)
V <- ldlt(A)
V$L
V$D
V$L %*% V$D %*% t(V$L)
A
```

`ltdl` *Computes L'DL Matrix Decomposition.*

Description

This function decomposes matrix V into $V = L'DL$, where L is a lower triangular matrix with unit diagonal and D is a diagonal matrix with non-negative terms.

Usage

```
ltdl(A)
```

Arguments

`A`	symmetric positive-definite `matrix`

Value

`list` of

`$L`	`matrix` L
`$D`	`matrix` D

References

de Jonge, P., Tiberius, C., 1996, *The LAMBDA Method for Integer Ambiguity Estimation: Implementation Aspects*, Universiteitsdrukkerij TU Delft.

Examples

```
A <- matrix(c(5,1,1,3),2,2)
V <- ltdl(A)
V$L
V$D
t(V$L) %*% V$D %*% V$L
A
```

mixest1 *Computes Mixture Estimation with State-Space Components.*

Description

This function estimates recursively mixtures with state-space components with a dynamic model of switching. The components are normal linear models. Suppose there are available k potentially important predictors of y, i.e., $x_1, \ldots, x_k$. Then up to 2^k linear models including constant term can be created by inclding or not including each of these predictors in the individual model, i.e., component of the mixture.

Usage

```
mixest1(y,x,mods=NULL,ftype=NULL,lambda=NULL,kappa=NULL,V=NULL,W=NULL,atype=NULL)
```

Arguments

y	one column matrix of forecasted time-series, observations inserted rowwise
x	matrix of independent time-series (predictors), observations inserted rowwise
mods	optional, matrix indicating which models should be used as components, the first column indicates inclusion of a constant in a component model, by default all possible models with a constant are used, inclusion of a variable is indicated by 1, omitting by 0, component models are indexed by rows, variables (time-series) are indexed by columns
ftype	optional, numeric indicating type of forecasting, 0 represents forecasting based on coefficients derived from the estimated mixture, 1 represents averaging forecasts from all components by the estimated weights, 2 represents selecting the forecast given by the model with the highest weight, 3 represents selecting the forecast from the so-called median probability model (Barbieri and Berger, 2004), by default ftype=0 is taken
lambda	optional, numeric between 0 and 1, a forgetting factor in covariance estimation method described by Raftery et al. (2010), by default the method of Nagy and Suzdaleva (2013) is used
kappa	optional, numeric between 0 and 1, a parameter for the exponentially weighted moving average estimation of components variances, described for example by Koop and Korobilis (2012), if lambda is specified but kappa is not, then the method of recursive moments described by Raftery et al. (2010) is used, by default the method of Nagy and Suzdaleva (2013) is used
V	optional, numeric initial variance for all components (output equation), by default V=1 is taken
W	optional, numeric initial value to be put in the diagonal matrix representing the covariance matrices (state equation), by default W=1 is taken
atype	optional, numeric indicating approximation of pdf, 0 represents quasi-Bayesian approach, 1 represents minimization of the Kerridge inaccuracy (where suitable optimization is done with the Gauss-Newton method, still this increases the computation time greatly), by default atype=0 is taken

Value

object of class mixest, i.e., list of

$y.hat	vector of predictions
$rvi	matrix of relative variable importances
$coef	matrix of regression coefficients corresponding to ftype method chosen
$weights	matrix of estimated weights of component models
$V	vector of updated variances from the selected models, consistent with ftype chosen
$R	matrix of updated diagonal of covariances corresponding to independent variables in regressions, consistent with ftype chosen
$components	matrix of mods
$parameters	character of parameters used in the model
$data.last	list of selected parameters obtained in the last iteration, necessary for the internal use by cauimp

Source

Nagy, I., Suzdaleva, E., 2013, Mixture estimation with state-space components and Markov model of switching. *Applied Mathematical Modelling* **37**, 9970–9984.

References

Barbieri, M. M., Berger, J. O., 2004, Optimal predictive model selection. *The Annals of Statistics* **32**, 870–897.

Burnham, K. P., Anderson, D. R., 2002, *Model Selection and Multimodel Inference*, Springer.

Karny, M. (ed.), 2006, *Optimized Bayesian Dynamic Advising*, Springer.

Koop, G., Korobilis, D., 2012, Forecasting inflation using Dynamic Model Averaging. *International Economic Review* **53**, 867–886.

Nagy, I., Suzdaleva, E., 2017, *Algorithms and Programs of Dynamic Mixture Estimation*, Springer.

Quarteroni, A., Sacco, R., Saleri, F., 2007, *Numerical Mathematics*, Springer.

Raftery, A. E., Karny, M., Ettler, P., 2010, Online prediction under model uncertainty via Dynamic Model Averaging: Application to a cold rolling mill. *Technometrics* **52**, 52–66.

See Also

mixest2

Examples

```
data(oil)
m1 <- mixest1(y=oil[,1,drop=FALSE],x=oil[,-1,drop=FALSE],ftype=1,V=100,W=100)

# Models with only one variable
mods <- diag(1,nrow=ncol(oil[,-1,drop=FALSE]),ncol=ncol(oil[,-1,drop=FALSE]))
```

```
mods <- cbind(1,mods)
m2 <- mixest1(y=oil[,1,drop=FALSE],x=oil[,-1,drop=FALSE],mods=mods,ftype=1,V=100,W=100)
```

`mixest2` *Computes Mixture Estimation with Normal Regression Components.*

Description

This function estimates recursively mixtures with normal regression components with a dynamic model of switching.

Usage

```
mixest2(y,x,mods=NULL,ftype=NULL,V=NULL,W=NULL,atype=NULL,Tvar=NULL)
```

Arguments

`y`	one column `matrix` of forecasted time-series, observations inserted rowwise
`x`	`matrix` of independent time-series (predictors), observations inserted rowwise
`mods`	see `mixest1`
`ftype`	optional, `numeric` indicating type of forecasting, 1 represents averaging forecasts from all components by the estimated weights (i.e., forecasting based on coefficients derived from the estimated mixture), 2 represents selecting the forecast given by the model with the highest weight, 3 represents selecting the forecast from the so-called median probability model (Barbieri and Berger, 2004), by default `ftype=1` is taken
`V`	optional, `numeric` initial variance, the same for all components, by default `V=1` is taken
`W`	optional, `numeric` initial value to be put in the diagonal matrix representing the covariance matrices for regression coefficients, the same for all components, by default `W=1` is taken
`atype`	optional, `numeric` indicating approximation of pdfs, 0 represents quasi-Bayesian approach, 1 represents minimization of the Kerridge inaccuracy, by default `atype=0` is taken
`Tvar`	optional, `numeric` indicating the number of first observations, when variance and covariance updating will not be performed, i.e., the initial values will be kept fixed, by default `Tvar=30` is taken

Value

object of class `mixest`, i.e., `list` of

`$y.hat`	`vector` of predictions
`$rvi`	`matrix` of relative variable importances

$coef	matrix of regression coefficients corresponding to ftype method chosen
$weights	matrix of estimated weights of component models
$V	vector of updated variances from the selected models, consistent with ftype chosen
$R	matrix of updated diagonal of covariances corresponding to independent variables in regressions, consistent with ftype chosen
$components	matrix of mods
$parameters	character of parameters used in the model

Source

Nagy, I., Suzdaleva, E., Karny, M., Mlynarova, T., 2011, Bayesian estimation of dynamic finite mixtures. *International Journal of Adaptive Control and Signal Processing* **25**, 765–787.

References

Barbieri, M. M., Berger, J. O., 2004, Optimal predictive model selection. *The Annals of Statistics* **32**, 870–897.

Burnham, K. P., Anderson, D. R., 2002, *Model Selection and Multimodel Inference*, Springer.

Dedecius, K., 2010, *Partial Forgetting in Bayesian Estimation*, Czech Technical University in Prague.

Karny, M. (ed.), 2006, *Optimized Bayesian Dynamic Advising*, Springer.

Nagy, I., 2015, *Mixture Models and Their Applications*, Czech Technical University in Prague.

Nagy, I., Suzdaleva, E., 2017, *Algorithms and Programs of Dynamic Mixture Estimation*, Springer.

Quarteroni, A., Sacco, R., Saleri, F., 2007, *Numerical Mathematics*, Springer.

See Also

mixest1

Examples

```
data(oil)
m1 <- mixest2(y=oil[,1,drop=FALSE],x=oil[,-1,drop=FALSE],ftype=1,V=100,W=100)
```

oil *Crude Oil Data.*

Description

Selected data from oil market.

Usage

```
data(oil)
```

Format

oil is matrix object such that columnwise are

- crudeoil$p_oil – average spot price of crude oil (Brent, Dubai and WTI) in USD per barrel
- crudeoil$cons – U.S. product supplied of crude oil and petroleum products in thousand barrels
- crudeoil$r – U.S. 3-month treasury bill secondary market rate in %
- crudeoil$stocks – U.S. share prices index, 2015=100
- crudeoil$ex_rate – U.S. real effective exchange rate index (broad basket), 2020=100

Details

The data are in monthly frequency. They cover the period between Feb, 1998 and Oct, 2024. cons, r, stocks and ex_rate are lagged one period back.

Source

The data are provided by Bank for International Settlements, Board of Governors of the Federal Reserve System, OECD, U.S. Energy Information Administration and World Bank.

https://www.bis.org

https://www.eia.gov

https://www.federalreserve.gov

https://www.oecd.org/en.html

https://www.worldbank.org/ext/en/home

References

Bank for International Settlements, 2025. Effective exchange rates, BIS WS_EER 1.0 (data set). https://data.bis.org/topics/EER/BIS%2CWS_EER%2C1.0/M.R.B.US

Board of Governors of the Federal Reserve System, 2025. Selected interest rates. https://www.federalreserve.gov/releases/h15/

OECD, 2025. Share prices. https://www.oecd.org/en/data/indicators/share-prices.html

U.S. Energy Information Administration, 2025. Petroleum /& other liquids. https://www.eia.gov/petroleum/data.php

World Bank, 2025. Commodity markets. https://www.worldbank.org/en/research/commodity-markets

Examples

```
data(oil)
```

---

plot.mixest *Plots Selected Outcomes from* mixest *Object.*

Description

The function plots selected outcomes from `mixest` object.

Usage

```
## S3 method for class 'mixest'
plot(x, ...)
```

Arguments

x	an object of `mixest` class
...	not used

Details

The function plots a few outcomes from `mixest` object. First, the estimated regression coefficients are plotted separately for each variable. Credible intervals of 90% are added. Next, if averaging was chosen for forecasting, then relative variable importances are plotted, i.e., sum of weights of models containing the given variable. If selection procedure was chosen for forecasting, it is plotted whether the given variable is included in the selected model at the given time. Finally weights from all component models are presented in one plot.

See Also

convts

Examples

```
data(oil)
m1 <- mixest1(y=oil[,1,drop=FALSE],x=oil[,-1,drop=FALSE],ftype=2,V=100,W=100)
plot(m1)
```

plot.qbnmix *Plots Selected Outcomes from* qbnmix *Object.*

Description

The function plots selected outcomes from qbnmix object.

Usage

```
## S3 method for class 'qbnmix'
plot(x, ...)
```

Arguments

x	an object of qbnmix class
...	not used

Details

The function plots a few outcomes from qbnmix object. First, it plots means for each cluster. Then, it plots posterior probabilities for each cluster. Finally, estimates of mixing weights for each cluster.

Examples

```
R <- list(matrix(c(1,0.3,0,
                   0.3,0.3,0,
                   0,0,0.15),3,3),
          matrix(c(1,0,0,
                   0,0.5,0,
                   0,0,0.2),3,3))
data <- rbind(MASS::mvrnorm(n=180,c(5,2,3),R[[1]]),
              MASS::mvrnorm(n=20,c(1,2,3),R[[2]]))
data <- data[sample(nrow(data)),]

mu0 <- list(matrix(c(4.8689,1.9417,3.0175),nrow=1,ncol=3),
            matrix(c(1.0182,1.9903,2.8847),nrow=1,ncol=3))
est <- qbnmix(y=data,mu0=mu0)
plot(est)
```

plot.tvpreg — *Plots Selected Outcomes from* `tvpreg` *Object.*

Description

The function plots selected outcomes from `tvpreg` object.

Usage

```
## S3 method for class 'tvpreg'
plot(x, ...)
```

Arguments

`x`	an object of `tvpreg` class
`...`	not used

Details

The function plots the estimated regression coefficients, separately for each variable. 90% credible intervals are added.

See Also

`convts`

Examples

```
data(oil)
t1<- tvp.reg(y=oil[,1,drop=FALSE],x=oil[,-1,drop=FALSE],lambda=0.99,V=100,W=100)
plot(t1)
```

print.mixest — *Prints* `mixest` *Object.*

Description

The function prints selected outcomes obtained from object `mixest`.

Usage

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

Arguments

`x`	an object of `mixest` class
`...`	not used

Details

The function prints the general structure of the model, i.e., names of predictors. It also prints the number of observations (length of time-series) and the number of component models used in estimations (mixing). Additionally it prints the model's parameters (i.e., forecasting method, values of the initial parameters, etc.).

Examples

```
data(oil)
m1 <- mixest1(y=oil[,1,drop=FALSE],x=oil[,-1,drop=FALSE],ftype=2,V=100,W=100)
print(m1)
```

`print.qbnmix` *Prints* `qbnmix` *Object.*

Description

The function prints selected outcomes obtained from `qbnmix`.

Usage

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

Arguments

`x`	an object of `qbnmix` class
`...`	not used

Details

The function prints estimated means and covariance matrices from the last step.

Examples

```
R <- list(matrix(c(1,0.3,0,
                   0.3,0.3,0,
                   0,0,0.15),3,3),
          matrix(c(1,0,0,
                   0,0.5,0,
                   0,0,0.2),3,3))
data <- rbind(MASS::mvrnorm(n=180,c(5,2,3),R[[1]]),
```

```
              MASS::mvrnorm(n=20,c(1,2,3),R[[2]]))
data <- data[sample(nrow(data)),]

mu0 <- list(matrix(c(4.8689,1.9417,3.0175),nrow=1,ncol=3),
            matrix(c(1.0182,1.9903,2.8847),nrow=1,ncol=3))
est <- qbnmix(y=data,mu0=mu0)
print(est)
```

`print.tvpreg` *Prints* `tvpreg` *Object.*

Description

The function prints selected outcomes obtained from object `tvpreg`.

Usage

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

Arguments

`x`	an object of `tvpreg` class
`...`	not used

Details

The function prints the general structure of the model, i.e., names of predictors. It also prints the number of observations (length of time-series) and the regression coefficients as estimated in the last period.

Examples

```
data(oil)
t1<- tvp.reg(y=oil[,1,drop=FALSE],x=oil[,-1,drop=FALSE],lambda=0.99,V=100,W=100)
print(t1)
```

qbnmix *Estimates Normal Mixtures.*

Description

This function performs a recursive clustering for normal mixtures. Quasi-Bayesian approximation is performed.

Usage

```
qbnmix(y,m=2,mu0=NULL,R0=NULL)
```

Arguments

y	matrix of observations, rows correspond to observations, columns correspond to tuples
m	numeric specifying the number of components (clusters), by default m=2 is taken
mu0	optional, initial means, should be a list of m matrices, each of them having one row and ncol(y) columns, if not specified random values are taken
R0	optional, initial covariance matrices, should be a list of m matrices, each of them having ncol(y) rows and ncol(y) columns, if not specified identity matrices are taken

Value

object of class qbnmix, i.e., list of

$mu	list of estimated means
$R	list of estimated covariance matrices (from last step only)
$alpha	matrix of estimates of mixing weights (components columnwise)
$w	matrix of posterior probabilities (components columnwise)
$mu0	list of initial means matrices
$R0	list of initial covaraince matrices

Source

Karny, M., Kadlec, J., Sutanto, E.L., 1998, Quasi-Bayes estimation applied to normal mixture, *Preprints of The 3rd European IEEE Workshop on Computer-Intensive Methods in Control and Data Processing*, Rojicek, J., Valeckova, M., Karny, M., Warwick K. (eds.), UTIA AV CR, 77–82.

Examples

```
R <- list(matrix(c(1,0.3,0,
                   0.3,0.3,0,
                   0,0,0.15),3,3),
          matrix(c(1,0,0,
                   0,0.5,0,
                   0,0,0.2),3,3))
data <- rbind(MASS::mvrnorm(n=180,c(5,2,3),R[[1]]),
              MASS::mvrnorm(n=20,c(1,2,3),R[[2]]))
data <- data[sample(nrow(data)),]

mu0 <- list(matrix(c(4.8689,1.9417,3.0175),nrow=1,ncol=3),
            matrix(c(1.0182,1.9903,2.8847),nrow=1,ncol=3))
est <- qbnmix(y=data,mu0=mu0)
```

---

sqrtmat *Computes the Square Root of a Matrix.*

Description

This function computes the square root of a matrix.

Usage

```
sqrtmat(A)
```

Arguments

A	symmetric positive-definite matrix

Value

matrix B such that $BB' = A$

References

https://en.wikipedia.org/wiki/Square_root_of_a_matrix

Examples

```
A <- matrix(c(5,1,1,3),2,2)
B <- sqrtmat(A)
B %*% t(B)
A
```

tvp.reg *Computes Time-Varying Parameters Regression.*

Description

This function estimates Time-Varying Parameters regression.

Usage

```
tvp.reg(y,x,lambda=NULL,kappa=NULL,V=NULL,W=NULL)
```

Arguments

y	one column matrix of forecasted time-series, observations inserted rowwise
x	matrix of independent time-series (predictors), observations inserted rowwise
lambda	optional, see mixest1
kappa	optional, see mixest1
V	optional, numeric initial variance, by default V=1 is taken
W	optional, numeric initial value to be put on diagonal of covariance matrix, by default W=1 is taken

Details

If lambda is specified, then the method described by Raftery et al. (2010) is used, with possible extentsion to the one described by Koop and Korobilis (2012). Otherwise, the Kalman filter described as by Nagy and Suzdaleva (2013) is used.

Value

object of class tvpreg, i.e., list of

$y.hat	vector of predictions
$coef	matrix of regression coefficients
$R	matrix of diagonals of covariances corresponding to independent variables in regressions
$V	vector of outcome variances

References

Koop, G., Korobilis, D., 2012, Forecasting inflation using Dynamic Model Averaging. *International Economic Review* **53**, 867–886.

Nagy, I., Suzdaleva, E., 2017, *Algorithms and Programs of Dynamic Mixture Estimation*, Springer.

Raftery, A. E., Karny, M., Ettler, P., 2010, Online prediction under model uncertainty via Dynamic Model Averaging: Application to a cold rolling mill. *Technometrics* **52**, 52–66.

Examples

```
data(oil)
t1 <- tvp.reg(y=oil[,1,drop=FALSE],x=oil[,-1,drop=FALSE],lambda=0.99,V=100,W=100)
t2 <- tvp.reg(y=oil[,1,drop=FALSE],x=oil[,-1,drop=FALSE],V=100,W=100)
```

Index