Package 'dLagM'

September 11, 2017

Type Package

Title Time Series Regression Models with Distributed Lag Models

Version 0.1.0

Date 2017-09-11

Author Haydar Demirhan

Maintainer Haydar Demirhan <haydar.demirhan@rmit.edu.au>

Description Provides time series regression models with one predictor using finite distributed lag models, polynomial (Almon) distributed lag models, geometric distributed lag models with Koyck transformation, and autoregressive distributed lag models. It also consists of functions for computation of h-step ahead forecasts from these models. See Baltagi (2011) <doi:10.1007/978-3-642-20059-5> for more information.

Depends stats, dynlm, wavethresh, AER, Hmisc

License GPL-3

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-09-11 11:00:40 UTC

R topics documented:

dLagM-package 2
ardlDlm 3
ardlDlmForecast 4
dlm 5
dlmForecast 6
finiteDLMauto 7
koyckDlm 8
koyckDlmForecast 9
MASE 10
polyDlm 12
polyDlmForecast 13
sortScore 14
warming 15

Index **16**

dLagM-package *Implementation of Time Series Regression Models with Distributed Lag Models*

Description

Provides time series regression models with one predictor using finite distributed lag models, polynomial (Almon) distributed lag models, geometric distributed lag models with Koyck transformation, and autoregressive distributed lag models. It also consists of functions for computation of h-step ahead forecasts from these models. See Baltagi (2011) for more information.

Details

Package:	dLagM
Type:	Package
Version:	0.1.0
Date:	2017-09-11
License:	GPL-3

To implement time series regression with finite distributed lag models use `dlm` function.

To implement time series regression with polynomial distributed lag models use `polyDlm` function.

To implement time series regression with geometric distributed lag models with Koyck transformation use `koyckDlm` function.

To implement time series regression with autoregressive distributed lag models use `ardlDlm` function.

Author(s)

Haydar Demirhan

Maintainer: Haydar Demirhan <haydar.demirhan@rmit.edu.au>

References

B.H. Baltagi. *Econometrics*, Fifth Ed. Springer, 2011.

R.C. Hill, W.E. Griffiths, G.G. Judge. *Undergraduate Econometrics*. Wiley, 2000.

See Also

`dlm`, `polyDlm`, `koyckDlm`, `ardlDlm`

Examples

```
# --- For examples, please refer to specific functions ---
```

`ardlDlm` *Implement finite autoregressive distributed lag model*

Description

A function that applies autoregressive distributed lag models of order (p , q) with one predictor.

Usage

```
ardlDlm(x , y , p = 1 , q = 1 , show.summary = TRUE)
```

Arguments

`x`	A vector including the observations of predictor time series. This is not restricted to `ts` objects.
`y`	A vector including the observations of dependent time series. This is not restricted to `ts` objects.
`p`	An integer representing the order of autoregressive process.
`q`	An integer representing finite lag length.
`show.summary`	If `TRUE`, prints standard model summary for the model of interest.

Details

The autoregressive DLM is a flexible and parsimonious infinite DLMs. We write the model ARDL(p, q) as

$$Y_t = \mu + \beta_0 X_t + \beta_1 X_{t-1} + \cdots + \beta_p X_{t-p} + \gamma_1 Y_{t-1} + \cdots + \gamma_q Y_{t-q} + e_t.$$

Value

`model`	An object of class `lm`. See the details of `lm` function.
`order`	A vector composed of p and q orders.

Author(s)

Haydar Demirhan

Maintainer: Haydar Demirhan <haydar.demirhan@rmit.edu.au>

References

B.H. Baltagi. *Econometrics*, Fifth Ed. Springer, 2011.

R.C. Hill, W.E. Griffiths, G.G. Judge. *Undergraduate Econometrics*. Wiley, 2000.

Examples

```
data(warming)
model.ardl = ardlDlm(x = warming$NoMotorVehicles,
y = warming$Warming, p = 1 , q = 1 , show.summary = TRUE)
```

`ardlDlmForecast` *Compute forecasts for autoregressive distributed lag models*

Description

A function that computes forecasts for autoregressive distributed lag model with one predictor.

Usage

```
ardlDlmForecast(model , x , h = 1)
```

Arguments

`model`	An object of class `lm` including the fitted model with `ardl.dlm()` function.
`x`	A vector including the new observations of independent time series. This is not restricted to `ts` objects.
`h`	The number of ahead forecasts.

Details

This function directly uses the model formula and estimates of model coefficients to find forecast one-by-one starting from the one-step ahead forecast.

Value

`forecasts`	A vector including forecasts.

Author(s)

Haydar Demirhan

Maintainer: Haydar Demirhan <haydar.demirhan@rmit.edu.au>

Examples

```
data(warming)
model.ardl = ardlDlm(x = warming$NoMotorVehicles,
y = warming$Warming, p = 1 , q = 1 , show.summary = TRUE)
ardlDlmForecast(model = model.ardl , x = c(95, 98) , h = 2)
```

dlm *Implement finite distributed lag model*

Description

A function that applies distributed lag models with one predictor.

Usage

```
dlm(x , y , q , show.summary = TRUE)
```

Arguments

x	A vector including the observations of predictor time series. This is not restricted to ts objects.
y	A vector including the observations of dependent time series. This is not restricted to ts objects.
q	An integer representing finite lag length.
show.summary	If TRUE, prints standard model summary for the model of interest.

Details

When a decision made on a variable, some of the related variables would be effected through time. For example, when income tax rate is increased, this would reduce expenditures of consumers on goods and services, which reduces profits of suppliers, which reduces the demand for productive inputs, which reduces the profits of the input suppliers, and so on (Judge and Griffiths, 2000). These effects occur over the future time periods; hence, they are distributed across the time.

In a distributed-lag model, the effect of an independent variable X on a dependent variable Y occurs over the time. Therefore, DLMs are dynamic models. A linear finite DLM with one independent variable is written as follows:

$$Y_t = \alpha + \sum_{s=0}^{q} \beta_s X_{t-s} + \epsilon_t,$$

where ϵ_t is a stationary error term with $E(\epsilon_t) = 0, Var(\epsilon_t) = \sigma^2, Cov(\epsilon_t, \epsilon_s) = 0$.

Value

model	An object of class lm. See the details of href{https://stat.ethz.ch/R-manual/R-devel/library/st function.
designMatrix	The design matrix composed of transformed z-variables.

Author(s)

Haydar Demirhan

Maintainer: Haydar Demirhan <haydar.demirhan@rmit.edu.au>

References

B.H. Baltagi. *Econometrics*, Fifth Ed. Springer, 2011.

R.C. Hill, W.E. Griffiths, G.G. Judge. *Undergraduate Econometrics*. Wiley, 2000.

Examples

```
data(warming)
model.dlm = dlm(x = warming$NoMotorVehicles ,
y = warming$Warming , q = 2 ,  show.summary = TRUE)
```

`dlmForecast` *Compute forecasts for finite distributed lag model*

Description

A function that computes forecasts for finite distributed lag model with one predictor.

Usage

```
dlmForecast(model , x , h = 1)
```

Arguments

`model`	An object of class `lm` including the fitted model with `dlm()` function.
`x`	A vector including the new observations of independent time series. This is not restricted to `ts` objects.
`h`	The number of ahead forecasts.

Details

This function directly uses the model formula and estimates of model coefficients to find forecast one-by-one starting from the one-step ahead forecast.

Value

`forecasts`	A vector including forecasts.

Author(s)

Haydar Demirhan

Maintainer: Haydar Demirhan <haydar.demirhan@rmit.edu.au>

Examples

```
data(warming)
model.dlm = dlm(x = warming$NoMotorVehicles ,
y = warming$Warming , q = 2 ,  show.summary = TRUE)
dlmForecast(model = model.dlm , x = c(95 ,
98, 101) , h = 3)
```

`finiteDLMauto` *Find the optimal lag lenght for finite DLMs*

Description

A function that fits finite DLMs for a range of lag lenths and orders the fitted models according to a desired measure.

Usage

```
finiteDLMauto(x, y, q.min = NULL, q.max = NULL, k.order = NULL,
model.type = c("dlm","poly"), error.type = c("MASE","AIC","BIC","radj"), trace = FALSE)
```

Arguments

`x`	A vector including the observations of predictor time series. This is not restricted to `ts` objects.
`y`	A vector including the observations of dependent time series. This is not restricted to `ts` objects.
`q.min`	An integer representing the lower limit of the range of lag lengths to be considered. If missing, it will be set to 1.
`q.max`	An integer representing the upper limit of the range of lag lengths to be considered. If missing, it will be set to 10.
`k.order`	An integer representing order of polynomial distributed lags.
`model.type`	The type of model to be fitted. If set to `dlm`, finite distributed lag models are fitted. If set to `poly`, polynomial distributed lag models are fitted.
`error.type`	The type of goodness-of-fit measure to be used for the selection of optimal lag length. If set to `MASE`, the optimal lag lenthg is determined according to MASE. If set to `AIC`, the optimal lag lenthg is determined according to AIC. If set to `BIC`, the optimal lag lenthg is determined according to BIC. If set to `radj`, the optimal lag lenthg is determined according to Adjusted R-square.
`trace`	If `TRUE`, prints all of the goodness-of-fit measures for all fitted models.

Details

If `q.max` is entered too large, its value will be adjusted to have a sample size of 10 for fitting the regression model for finite DLMs.

Value

Retunrs a `data.frame` including the values of goodness-of-fit measures and corresponding lag lenghts.

Author(s)

Agung Andiojaya

Maintainer: Agung Andiojaya <a@gmail.com>

Examples

```
data(warming)
# Run the search over polynomial DLMs according to MASE values
finiteDLMauto(x = warming$NoMotorVehicles , y = warming$Warming ,
q.max = 12, k.order = 3, model.type = "poly",
error.type = "MASE", trace = TRUE)

# Run the search over finite DLMs according to AIC values
finiteDLMauto(x = warming$NoMotorVehicles , y = warming$Warming ,
q.min = 2, q.max = 12, model.type = "dlm" ,error.type = "AIC",
trace = TRUE)
```

---

koyckDlm *Implement finite autoregressive distributed lag model*

Description

A function that applies autoregressive distributed lag models of order (p , q) with one predictor.

Usage

```
koyckDlm(x , y , show.summary = TRUE)
```

Arguments

x	A vector including the observations of predictor time series. This is not restricted to ts objects.
y	A vector including the observations of dependent time series. This is not restricted to ts objects.
show.summary	If TRUE, prints standard model summary for the model of interest.

Details

To deal with infinite DLMs, we can use the Koyck transformation. When we apply Koyck transformation, we get the following:

$$Y_t - \phi Y_{t-1} = \alpha(1-\phi) + \beta X_t + (\epsilon_t - \phi\epsilon_{t-1}).$$

When we solve this equation for Y_t, we obtain Koyck DLM as follows:

$$Y_t = \delta_1 + \delta_2 Y_{t-1} + \delta_3 X_t + \nu_t,$$

where $\delta_1 = \alpha(1-\phi), \delta_2 = \phi, \delta_3 = \beta$ and the random error after the transformation is $\nu_t = (\epsilon_t - \phi\epsilon_{t-1})$ (Judge and Griffiths, 2000).

Then, instrumental variables estimation is employed to fit the model.

Value

`model` An object of class `ivreg`. See the details of `ivreg` function.

`geometric.coefficients`
A vector composed of corresponding geometric distributed lag model coefficients.

Author(s)

Haydar Demirhan

Maintainer: Haydar Demirhan <haydar.demirhan@rmit.edu.au>

References

B.H. Baltagi. *Econometrics*, Fifth Ed. Springer, 2011.

R.C. Hill, W.E. Griffiths, G.G. Judge. *Undergraduate Econometrics*. Wiley, 2000.

Examples

```
data(warming)
model.koyck = koyckDlm(x = warming$NoMotorVehicles ,
y = warming$Warming , show.summary = TRUE)
```

---

koyckDlmForecast *Compute forecasts for Koyck transformation of distributed lag models*

Description

A function that computes forecasts for Koyck transformation of distributed lag model with one predictor.

Usage

```
koyckDlmForecast(model , x , h = 1)
```

Arguments

`model`	An object of class `lm` including the fitted model with `koyck.dlm()` function.
`x`	A vector including the new observations of independent time series. This is not restricted to `ts` objects.
`h`	The number of ahead forecasts.

Details

This function directly uses the model formula and estimates of model coefficients to find forecast one-by-one starting from the one-step ahead forecast.

Value

`forecasts` A vector including forecasts.

Author(s)

Haydar Demirhan

Maintainer: Haydar Demirhan <haydar.demirhan@rmit.edu.au>

Examples

```
data(warming)
model.koyck = koyckDlm(x = warming$NoMotorVehicles ,
y = warming$Warming , show.summary = TRUE)
koyckDlmForecast(model = model.koyck  , x = c(95, 98, 101), h = 3)
```

MASE *Compute mean absolute scaled error (MASE)*

Description

A function that computes mean absolute scaled error for fitted models.

Usage

```
MASE(model, ...)
```

Arguments

`model`	Model object fitted for time series data.
`...`	Optionally more fitted models.

Details

Let $e_t = Y_t - \hat{Y}_t$ be the one-step-ahead forecast error. Then, a scaled error is defined as

$$q_t = \frac{e_t}{\frac{1}{n-1}\sum_{i=2}^{n}|Y_i - Y_{i-1}|},$$

which is independent of the scale of the data. Mean absolute scaled error is defined as

$$MASE = mean(|q_t|)$$

(Hyndman and Koehler, 2006).

Fitted models would be finite, polynomial, Koyck, ARDL DLMs, or linear model fitted with `lm()` function. This function also computes MASE values of different models when fed at the same time.

Value

MASE — Mean absolute scaled error (MASE) for the observed and fitted series sent into the function.

Author(s)

Haydar Demirhan

Maintainer: Haydar Demirhan <haydar.demirhan@rmit.edu.au>

References

Hyndman, R.J. and Koehler, A.B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22, 679-688.

Examples

```
data(warming)
# Fit a bunch of polynomial DLMs
model.poly1 = polyDlm(x = warming$NoMotorVehicles , y = warming$Warming ,
                      q = 2 , k = 2 , show.beta = TRUE ,  show.summary = FALSE)
model.poly2 = polyDlm(x = warming$NoMotorVehicles , y = warming$Warming ,
                       q = 3 , k = 2 , show.beta = TRUE ,  show.summary = FALSE)
model.poly3 = polyDlm(x = warming$NoMotorVehicles , y = warming$Warming ,
                       q = 4 , k = 2 , show.beta = TRUE ,  show.summary = FALSE)
MASE(model.poly1, model.poly2, model.poly3)

# Fit a bunch of finite DLMs
model.dlm1 = dlm(x = warming$NoMotorVehicles , y = warming$Warming, q =2)
model.dlm2 = dlm(x = warming$NoMotorVehicles , y = warming$Warming, q =3)
model.dlm3 = dlm(x = warming$NoMotorVehicles , y = warming$Warming, q =4)
MASE(model.dlm1, model.dlm2, model.dlm3)

# Fit a linear model
model.lm = lm(Warming ~ NoMotorVehicles , data = warming)
MASE(model.lm)

# Fit a Koyck model
model.koyck = koyckDlm(x = warming$NoMotorVehicles , y = warming$Warming )
MASE(model.koyck)

# Fit a bunch of ARDLs
model.ardl1 = ardlDlm(x = warming$NoMotorVehicles , y = warming$Warming, p=1, q=2)
model.ardl2 = ardlDlm(x = warming$NoMotorVehicles , y = warming$Warming, p=2, q=2)
model.ardl3 = ardlDlm(x = warming$NoMotorVehicles , y = warming$Warming, p=3, q=2)
MASE(model.ardl1 , model.ardl2 , model.ardl3)

# Find MASEs of different model objects
MASE(model.ardl1 , model.dlm1 , model.poly1, model.lm)
```

polyDlm *Implement finite polynomial distributed lag model*

Description

A function that applies polynomial distributed lag models with one predictor.

Usage

```
polyDlm(x , y , q , k , show.beta = TRUE , show.summary = TRUE)
```

Arguments

x	A vector including the observations of predictor time series. This is not restricted to ts objects.
y	A vector including the observations of dependent time series. This is not restricted to ts objects.
q	An integer representing finite lag length.
k	An integer representing order of polynomial distributed lags.
show.beta	If TRUE, generates original beta parameters and associate t-tests and prints the results.
show.summary	If TRUE, prints standard model summary for the model of interest.

Details

When a decision made on a variable, some of the related variables would be effected through time. For example, when income tax rate is increased, this would reduce expenditures of consumers on goods and services, which reduces profits of suppliers, which reduces the demand for productive inputs, which reduces the profits of the input suppliers, and so on (Judge and Griffiths, 2000). These effects occur over the future time periods; hence, they are distributed across the time.

In a distributed-lag model, the effect of an independent variable X on a dependent variable Y occurs over the time. Therefore, DLMs are dynamic models. A linear finite DLM with one independent variable is written as follows:

$$Y_t = \alpha + \sum_{s=0}^{q} \beta_s X_{t-s} + \epsilon_t,$$

where ϵ_t is a stationary error term with $E(\epsilon_t) = 0, Var(\epsilon_t) = \sigma^2, Cov(\epsilon_t, \epsilon_s) = 0$.

Value

model	An object of class lm. See the details of href{https://stat.ethz.ch/R-manual/R-devel/library/st function.
designMatrix	The design matrix composed of transformed z-variables.

`designMatrix.x` The design matrix composed of original x-variables.

`beta.coefficients`
Estimates and t-tests of original beta coefficients. This will be generated if `show.beta` is set to `TRUE`.

Author(s)

Haydar Demirhan

Maintainer: Haydar Demirhan <haydar.demirhan@rmit.edu.au>

References

B.H. Baltagi. *Econometrics*, Fifth Ed. Springer, 2011.

R.C. Hill, W.E. Griffiths, G.G. Judge. *Undergraduate Econometrics*. Wiley, 2000.

Examples

```
data(warming)
model.poly = polyDlm(x = warming$NoMotorVehicles , y = warming$Warming ,
q = 2 , k = 2 , show.beta = TRUE ,  show.summary = TRUE)
```

---

polyDlmForecast *Compute forecasts for polynomial distributed lag model*

Description

A function that computes forecasts for polynomial distributed lag model with one predictor.

Usage

```
polyDlmForecast(model , x , h = 1)
```

Arguments

`model`	An object of class `lm` including the fitted model with `poly.dlm()` function.
`x`	A vector including the new observations of independent time series. This is not restricted to `ts` objects.
`h`	The number of ahead forecasts.

Details

This function directly uses the model formula and estimates of model coefficients to find forecast one-by-one starting from the one-step ahead forecast.

Value

`forecasts` A vector including forecasts.

Author(s)

Haydar Demirhan

Maintainer: Haydar Demirhan <haydar.demirhan@rmit.edu.au>

Examples

```
data(warming)
model.poly = polyDlm(x = warming$NoMotorVehicles , y = warming$Warming ,
q = 2 , k = 2 , show.beta = TRUE ,  show.summary = TRUE)
polyDlmForecast(model = model.poly , x = c(95, 98) , h = 1)
```

---

sortScore *Sort ACI, BIC and MASE scores*

Description

A function that displays sorted AIC, BIC, and MASE scores.

Usage

```
sortScore(x, score = c("bic", "aic", "mase"))
```

Arguments

x	A vector of AIC, BIC, or MASE values.
score	The type of scores to be sorted.

Details

This function sorts the AIC, BIC, or MASE scores to display the smallest one at the top of a bunch of AIC, BIC, or MASE scores.

Author(s)

Cameron Doyle

Maintainer: Cameron Doyle <cdoyle305@gmail.com>

Examples

```
data(warming)
model.poly1 = polyDlm(x = warming$NoMotorVehicles , y = warming$Warming ,
                      q = 2 , k = 2 , show.beta = TRUE ,  show.summary = TRUE)
model.poly2 = polyDlm(x = warming$NoMotorVehicles , y = warming$Warming ,
                       q = 3 , k = 2 , show.beta = TRUE ,  show.summary = TRUE)
model.poly3 = polyDlm(x = warming$NoMotorVehicles , y = warming$Warming ,
                       q = 4 , k = 2 , show.beta = TRUE ,  show.summary = TRUE)

aic = AIC(model.poly1$model, model.poly2$model, model.poly3$model)
```

```
bic = BIC(model.poly1$model, model.poly2$model, model.poly3$model)
mase = MASE(model.poly1$model, model.poly2$model, model.poly3$model)

sortScore(aic , score = "aic")
sortScore(bic , score = "bic")
sortScore(mase , score = "mase")
```

`warming` *Global warming and vehicle prediction data*

Description

This data set is composed of annual mean global warming series between 1997 and 2016 showing the change in global surface temperature relative to 1951-1980 average temperatures and the number of vehicles produced within the same time span over the globe.

Usage

```
data(warming)
```

Format

multiple time series

Source

Global Climate Center, NASA Organisation Internationale des Constructeurs d'Automobiles (OICA)

References

https://climate.nasa.gov/vital-signs/global-temperature/

http://www.oica.net/category/production-statistics/

Examples

```
data(warming)
vehicleWarming.ts = ts(warming[,2:3], start = 1997)
plot(vehicleWarming.ts, main="Time series plots
of global warming and the nuber of produced motor
vehciles series.")
```

Index

∗Topic **datasets**
warming, 15
∗Topic **distributed lag model, time series, regression model, polynomial lag, predictor**
dLagM-package, 2

ardlDlm, *2*, 3
ardlDlmForecast, 4

dLagM-package, 2
dlm, *2*, 5
dlmForecast, 6

finiteDLMauto, 7

koyckDlm, *2*, 8
koyckDlmForecast, 9

MASE, 10

polyDlm, *2*, 12
polyDlmForecast, 13

sortScore, 14

warming, 15