Package ‘dynlm’

August 6, 2016

Version 0.3-5
Date 2016-08-04
Title Dynamic Linear Regression
Description Dynamic linear models and time series regression.
Depends R (>= 2.10.0), zoo
Suggests datasets, sandwich, strucchange, TSA
Imports stats, car (>= 2.0-0), lmtest
LazyLoad yes
LazyData yes
License GPL-2 | GPL-3
NeedsCompilation no
Author Achim Zeileis [aut, cre]
Maintainer Achim Zeileis <Achim.Zeileis@R-project.org>
Repository CRAN
Date/Publication 2016-08-06 00:55:50

R topics documented:
dynlm .......................................................... 2
M1Germany ...................................................... 5

Index 7
dynlm

Dynamic Linear Models and Time Series Regression

Description

Interface to `lm.wfit` for fitting dynamic linear models and time series regression relationships.

Usage

dynlm(formula, data, subset, weights, na.action, method = "qr",
       model = TRUE, x = FALSE, y = FALSE, qr = TRUE, singular.ok = TRUE,
       contrasts = NULL, offset, start = NULL, end = NULL, ...)

Arguments

- **formula**: a "formula" describing the linear model to be fit. For details see below and `lm`.
- **data**: an optional "data.frame" or time series object (e.g., "ts" or "zoo"), containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which `lm` is called.
- **subset**: an optional vector specifying a subset of observations to be used in the fitting process.
- **weights**: an optional vector of weights to be used in the fitting process. If specified, weighted least squares is used with weights weights (that is, minimizing \(\sum(w \cdot e^2)\)); otherwise ordinary least squares is used.
- **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 `na.contiguous`, `na.locf`, and `na.approx` are available.
- **method**: the method to be used; for fitting, currently only method = "qr" is supported; method = "model.frame" returns the model frame (the same as with model = TRUE, see below).
- **model, x, y, qr**: logicals. If TRUE the corresponding components of the fit (the model frame, the model matrix, the response, the QR decomposition) are returned.
- **singular.ok**: logical. If FALSE (the default in S but not in R) a singular fit is an error.
- **contrasts**: an optional list. See the contrasts.arg of `model.matrix.default`.
- **offset**: this can be used to specify an `a priori` known component to be included in the linear predictor during fitting. An `offset` term can be included in the formula instead or as well, and if both are specified their sum is used.
- **start**: start of the time period which should be used for fitting the model.
- **end**: end of the time period which should be used for fitting the model.
- **...**: additional arguments to be passed to the low level regression fitting functions.
The package `dynlm` offers three advantages over the direct use of `lm`: 
1. extended formula processing, 
2. preservation of time series attributes, 
3. instrumental variables regression (via two-stage least squares).

For specifying the formula of the model to be fitted, there are additional functions available which allow for convenient specification of dynamics (via `d()` and `L()`) or linear/cyclical patterns (via `trend()`, `season()`, and `harmon()`). All new formula functions require that their arguments are time series objects (i.e., "ts" or "zoo").

Dynamic models: An example would be `d(y) ~ L(y, 2)`, where `d(x, k)` is `diff(x, lag = k)` and `L(x, k)` is `lag(x, lag = -k)`, note the difference in sign. The default for `k` is in both cases 1. For `L()`, it can also be vector-valued, e.g., `y ~ L(y, 1:4)`.

Trends: `y ~ trend(y)` specifies a linear time trend where `(1:n)/freq` is used by default as the regressor. `n` is the number of observations and `freq` is the frequency of the series (if any, otherwise `freq = 1`). Alternatively, `trend(y, scale = FALSE)` would employ `1:n` and `time(y)` would employ the original time index.

Seasonal/cyclical patterns: Seasonal patterns can be specified via `season(x, ref = NULL)` and harmonic patterns via `harmon(x, order = 1)`. `season(x, ref = NULL)` creates a factor with levels for each cycle of the season. Using the `ref` argument, the reference level can be changed from the default first level to any other. `harmon(x, order = 1)` creates a matrix of regressors corresponding to `cos(2 * o * pi * time(x))` and `sin(2 * o * pi * time(x))` where `o` is chosen from `1:order`.

See below for examples and `M1Germany` for a more elaborate application.

Furthermore, a nuisance when working with `lm` is that it offers only limited support for time series data, hence a major aim of `dynlm` is to preserve time series properties of the data. Explicit support is currently available for "ts" and "zoo" series. Internally, the data is kept as a "zoo" series and coerced back to "ts" if the original dependent variable was of that class (and no internal NAs were created by the `na.action`).

To specify a set of instruments, formulas of type `y ~ x1 + x2 | z1 + z2` can be used where `z1` and `z2` represent the instruments. Again, the extended formula processing described above can be employed for all variables in the model.

### Examples
```r
# multiplicative SARIMA(1,0,0)(1,0,0)_12 model fitted to UK seatbelt data
data("UKDriverDeaths", package = "datasets")
uk <- log10(UKDriverDeaths)
dfm <- dynlm(uk ~ L(uk, 1) + L(uk, 12))
dfm
```
## Explicitly set start and end

dfm <- dynlm(uk ~ L(uk, 1) + L(uk, 12), start = c(1975, 1), end = c(1982, 12))
dfm

## Remove lag 12

dfm0 <- update(dfm, . ~ . - L(uk, 12))
anova(dfm0, dfm)

## Add season term

dfm1 <- dynlm(uk ~ 1, start = c(1975, 1), end = c(1982, 12))
dfm2 <- dynlm(uk ~ season(uk), start = c(1975, 1), end = c(1982, 12))
anova(dfm1, dfm2)

plot(uk)
lines(fitted(dfm0), col = 2)
lines(fitted(dfm2), col = 4)

## Regression on multiple lags in a single call

dfm3 <- dynlm(uk ~ L(uk, c(1, 11, 12)), start = c(1975, 1), end = c(1982, 12))
anova(dfm, dfm3)

## Examples 7.11/7.12 from Greene (1993)

data("USDistLag", package = "lmtest")
dfm1 <- dynlm(consumption ~ gnp + L(consumption), data = USDistLag)
dfm2 <- dynlm(consumption ~ gnp + L(gnp), data = USDistLag)
plot(USDistLag[, "consumption"])
lines(fitted(dfm1), col = 2)
lines(fitted(dfm2), col = 4)
if(require("lmtest")) encomptest(dfm1, dfm2)

### Time Series Decomposition ###

## Airline data

data("AirPassengers", package = "datasets")
ap <- log(AirPassengers)
ap_fm <- dynlm(ap ~ trend(ap) + season(ap))
summary(ap_fm)

## Alternative time trend specifications:

**# time(ap)** 1949 + (0, 1, ..., 143)/12

**# trend(ap)** (1, 2, ..., 144)/12

**# trend(ap, scale = FALSE)** (1, 2, ..., 144)

## Exhibit 3.5/3.6 from Cryer & Chan (2008)
if(require("TSA")) {
data("tempdub", package = "TSA")
td_lm <- dynlm(tempdub ~ harmon(tempdub))
summary(td_lm)
plot(tempdub, type = "p")
lines(fitted(td_lm), col = 2)
German M1 Money Demand

Description

German M1 money demand.

Usage

data(M1Germany)

Format

M1Germany is a "zoo" series containing 4 quarterly time series from 1960(1) to 1996(3).

logm1 logarithm of real M1 per capita,

logprice logarithm of a price index,

loggnp logarithm of real per capita gross national product,

interest long-run interest rate.

Details

This is essentially the same data set as GermanM1, the important difference is that it is stored as a zoo series and not as a data frame. It does not contain differenced and lagged versions of the variables (as GermanM1) does, because these do not have to be computed explicitly before applying dynlm.

The (short) story behind the data is the following (for more detailed information see GermanM1): Lütkepohl et al. (1999) investigate the linearity and stability of German M1 money demand: they find a stable regression relation for the time before the monetary union on 1990-06-01 but a clear structural instability afterwards. Zeileis et al. (2005) re-analyze this data set in a monitoring situation.

Source

The data is provided by the German central bank and is available online in the data archive of the Journal of Applied Econometrics http://qed.econ.queensu.ca/jae/1999-v14.5/lutkepohl-terasvirta-wolters/.

References


See Also

GermanM1

Examples

data("M1Germany")
## fit the model of Luetkepohl et al. (1999) on the history period
## before the monetary unification
histfm <- dynlm(d(logm1) ~ d(loggnp, 2) + d(interest) + d(L(interest)) + d(logprice) +
    L(logm1) + L(loggnp) + L(interest) +
    season(logm1, ref = 4),
data = M1Germany, start = c(1961, 1), end = c(1990, 2))

## fit on extended sample period
fm <- update(histfm, end = c(1995, 4))

if(require("strucchange")) {
    scus <- gefp(fm, fit = NULL)
    plot(scus, functional = supLM(0.1))
}
Index

*Topic datasets
  M1Germany, 5
*Topic regression
  dynlm, 2

dynlm, 2
end.dynlm (dynlm), 2
GermanM1, 5, 6
index.dynlm (dynlm), 2
lm, 2, 3
lm.wfit, 2
M1Germany, 3, 5
merge.zoo, 3

na.approx, 2
na.contiguous, 2
na.fail, 2
na.locf, 2
na.omit, 2

offset, 2
options, 2

print.dynlm (dynlm), 2
print.summary.dynlm (dynlm), 2

recresid.dynlm (dynlm), 2
start.dynlm (dynlm), 2
summary.dynlm (dynlm), 2

time.dynlm (dynlm), 2

zoo, 3, 5