Package ‘ZIM’

August 28, 2018

Title Zero-Inflated Models (ZIM) for Count Time Series with Excess Zeros

Version 1.1.0

Description Analyze count time series with excess zeros.

Two types of statistical models are supported: Markov regression by Yang et al. (2013) <doi:10.1016/j.stamet.2013.02.001> and state-space models by Yang et al. (2015) <doi:10.1177/1471082X14535530>. They are also known as observation-driven and parameter-driven models respectively in the time series literature. The functions used for Markov regression or observation-driven models can also be used to fit ordinary regression models with independent data under the zero-inflated Poisson (ZIP) or zero-inflated negative binomial (ZINB) assumption. Besides, the package contains some miscellaneous functions to compute density, distribution, quantile, and generate random numbers from ZIP and ZINB distributions.

License GPL-3

Encoding UTF-8

RoxygenNote 6.1.0

Imports MASS

Suggests pscl, TSA

URL https://github.com/biostatstudio/ZIM

BugReports https://github.com/biostatstudio/ZIM/issues

NeedsCompilation no

Author Ming Yang [aut, cre],
Gideon Zamba [aut],
Joseph Cavanaugh [aut]

Maintainer Ming Yang <mingyang@biostatstudio.com>

Repository CRAN

Date/Publication 2018-08-28 13:04:25 UTC
R topics documented:

ZIM-package ................................................................. 2
bshift ................................................................. 3
dzim ................................................................. 4
dzim.control ................................................................. 4
dzim.filter ................................................................. 5
dzim.fit ................................................................. 6
dzim.plot ................................................................. 6
dzim.sim ................................................................. 7
dzim.smooth ................................................................. 7
injury ................................................................. 8
pvalue ................................................................. 8
syph ................................................................. 9
zim ................................................................. 11
zim.control ................................................................. 12
zim.fit ................................................................. 12
ZINB ................................................................. 13
ZIP ................................................................. 14

Description

Fits observation-driven and parameter-driven models for count time series with excess zeros.

Details

The package ZIM contains functions to fit statistical models for count time series with excess zeros (Yang et al., 2013, 2015). The main function for fitting observation-driven models is zim, and the main function for fitting parameter-driven models is dzim.

Note

The observation-driven models for zero-inflated count time series can also be fit using the function zeroinfl from the pscl package (Zeileis et al., 2008). Fitting parameter-driven models is based on sequential Monte Carlo (SMC) methods, which are computer intensive and could take several hours to estimate the model parameters.
bshift

References


bshift

*Backshift Operator*

**Description**

Apply the backshift operator or lag operator to a time series objective.

**Usage**

```
bshift(x, k = 1)
```

**Arguments**

- `x`: univariate or multivariate time series.
- `k`: number of lags.

**See Also**

`lag`, `zlag`

**Examples**

```
x <- arima.sim(model = list(ar = 0.8, sd = 0.5), n = 120)
bshift(x, k = 12)
```
**dzim**

*Fitting Dynamic Zero-Inflated Models*

**Description**

`dzim` is used to fit dynamic zero-inflated models.

**Usage**

```r
dzim(formula, data, subset, na.action, weights = 1, offset = 0, 
control = dzim.control(...), ...)
```

**Arguments**

- **formula**: an objective of class "formula".
- **data**: an optional dataframe, list or environment 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.
- **weights**: an optional vector of 'prior weights' to be used in the fitting process.
- **offset**: this can be used to specify a priori known component to be included in the linear predictor during fitting.
- **control**: control arguments from `dzim.control`
- **...**: additional arguments

**See Also**

`dzim.fit`, `dzim.filter`, `dzim.smooth`, `dzim.control`, `dzim.sim`, `dzim.plot`

---

**dzim.control**

*Auxiliary for Controlling DZIM Fitting*

**Description**

Auxiliary function for `dzim` fitting. Typically only used internally by `dzim.fit`, but may be used to construct a control argument for either function.

**Usage**

```r
dzim.control(dist = c("poisson", "nb", "zip", "znb"), trace = FALSE, 
start = NULL, order = 1, mu_0 = rep(0, order), Sigma_0 = diag(1, 
order), N = 1000, R = 1000, niter = 500)
```
**Arguments**

- `dist`: count model family.
- `trace`: logical; if TRUE, display iteration history.
- `start`: initial parameter values.
- `order`: autoregressive order.
- `mu0`: mean vector for initial state.
- `Sigma0`: covariance matrix for initial state.
- `N`: number of particles in particle filtering.
- `R`: number of replications in particle smoothing.
- `niter`: number of iterations.

**Note**

The default values of `N`, `R`, and `niter` are chosen based on our experience. In some cases, `N = 500`, `R = 500`, and `niter = 200` might be sufficient. The `dzim.plot` function should always be used for convergence diagnostics.

**See Also**

`dzim`, `dzim.fit`, `dzim.filter`, `dzim.smooth`, `dzim.sim`, `dzim.plot`

---

**dzim.filter**  
*Particle Filtering for DZIM*

**Description**

Function to implement the particle filtering method proposed by Gordsill et al. (1993).

**Usage**

`dzim.filter(y, X, w, para, control)`

**Arguments**

- `y`: response variable.
- `X`: design matrix.
- `w`: `log(w)` is used as an offset variable in the linear predictor.
- `para`: model parameters.
- `control`: control arguments.

**References**

See Also

dzim, dzim.fit, dzim.smooth, dzim.control, dzim.sim, dzim.plot

dzim.fit

Fitter Function for Dynamic Zero-Inflated Models

Description

dzim.fit is the basic computing engine called by dzim used to fit dynamic zero-inflated models. This should usually not be used directly unless by experienced users.

Usage

dzim.fit(y, X, offset = rep(0, n), control = dzim.control(...), ...)

Arguments

y response variable.
X design matrix.
offset offset variable.
control control arguments.
... additional arguments.

See Also

dzim, dzim.control, dzim.filter, dzim.smooth, dzim.sim, dzim.plot

dzim.plot

Trace Plots from DZIM

Description

Function to display trace plots from a dynamic zero-inflated model.

Usage

dzim.plot(object, k.inv = FALSE, sigma.sq = FALSE, ...)

Arguments

object objective from dzim or dzim.fit.
k.inv logical; indicating whether an inverse transformation is needed for the dispersion parameter.
sigma.sq logical; indicating whether a square transformation is needed for the standard deviation parameter.
... additional arguments.
**dzim.sim**  
*Simulate Data from DZIM*

**Description**

Simulate data from a dynamic zero-inflated model.

**Usage**

```r
dzim.sim(x, w, omega, k, beta, phi, sigma, mu0, Sigma0)
```

**Arguments**

- `x`: design matrix.
- `w`: log(w) is used as an offset variable in the linear predictor.
- `omega`: zero-inflation parameter.
- `k`: dispersion parameter.
- `beta`: regression coefficients.
- `phi`: autoregressive coefficients.
- `sigma`: standard deviation.
- `mu0`: mean vector of initial state.
- `Sigma0`: covariance matrix of initial state.

**See Also**

`dzim, dzim.fit, dzim.filter, dzim.smooth, dzim.control, dzim.plot`

**dzim.smooth**  
*Particle Smoothing for DZIM*

**Description**

Function to implement the particle smoothing method proposed by Gordsill et al. (2004).

**Usage**

```r
dzim.smooth(y, x, w, para, control)
```

**Arguments**

- `y`: response variable.
- `x`: design matrix.
- `w`: log(w) is used as an offset variable in the linear predictor.
- `para`: model parameters.
- `control`: control arguments.
References


See Also
dzim, dzim.fit, dzim.filter, dzim.control, dzim.sim, dzim.plot

---

Example: Injury Series from Occupational Health

**Description**

Monthly number of injuries in hospitals from July 1988 to October 1995.

**Source**

Numbers from Figure 1 of Yau et al. (2004).

**References**


**Examples**

data(injury)
plot(injury, type = "o", pch = 20, xaxt = "n", yaxt = "n", ylab = "Injury Count")
axis(side = 1, at = seq(1, 96, 8))
axis(side = 2, at = 0:9)
abline(v = 57, lty = 2)
mtext("Pre-intervention", line = 1, at = 25, cex = 1.5)
mtext("Post-intervention", line = 1, at = 80, cex = 1.5)

---

**pvalue**

*Function to Compute P-value.*

**Description**

Function to compute p-value based on a t-statistic.

**Usage**

```r
pvalue(t, df = Inf, alternative = c("two.sided", "less", "greater"))
```
Arguments

- \( t \): t-statistic.
- \( df \): degree of freedoms.
- alternative: type of alternatives.

Examples

\[
pvalue(1.96, \text{alternative = } "\text{greater}")
\]

---

**syph**  

*Example: Syphilis Series*

**Description**

Weekly number of syphilis cases in the United States from 2007 to 2010.

**Format**

A data frame with 209 observations on the following 69 variables:

| year | Week | a1 United States | a2 New England | a3 Connecticut | a4 Maine | a5 Massachusetts | a6 New Hampshire | a7 Rhode Island | a8 Vermont | a9 Mid. Atlantic | a10 New Jersey | a11 New York (Upstate) | a12 New York City | a13 Pennsylvania | a14 E.N. Central | a15 Illinois | a16 Indiana | a17 Michigan | a18 Ohio | a19 Wisconsin | a20 W.N. Central | a21 Iowa | a22 Kansas | a23 Minnesota | a24 Missouri |
Note

C.N.M.I.: Commonwealth of Northern Mariana Islands.
\textbf{Source}

CDC Morbidity and Mortality Weekly Report (\url{http://www.cdc.gov/MMWR}).

\textbf{Examples}

\begin{verbatim}
data(syph)
plot(ts(syph$a33), main = "Maryland")
\end{verbatim}

---

\textbf{Fitting Zero-Inflated Models}

\textbf{Description}

\texttt{zim} is used to fit zero-inflated models.

\textbf{Usage}

\texttt{zim(formula, data, subset, na.action, weights = 1, offset = 0, control = zim.control(...), ...)}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{formula} an objective of class "\texttt{formula}".
  \item \texttt{data} an optional dataframe, list or environment containing the variables in the model.
  \item \texttt{subset} an optional vector specifying a subset of observations to be used in the fitting process.
  \item \texttt{na.action} a function which indicates what should happen when the data contain NAs.
  \item \texttt{weights} an optional vector of 'prior weights' to be used in the fitting process.
  \item \texttt{offset} this can be used to specify a priori known component to be included in the linear predictor during fitting.
  \item \texttt{control} control arguments.
  \item \ldots\ additional arguments.
\end{itemize}

\textbf{Note}

\texttt{zim} is very similar to \texttt{zeroinfl} from the \texttt{pscl} package. Both functions can be used to fit observation-driven models for zero-inflated time series.

\textbf{See Also}

\texttt{zim.fit}, \texttt{zim.control}
**zim.control**  
*Auxiliary for Controlling ZIM Fitting*

**Description**

Auxiliary function for *zim* fitting. Typically only used internally by *zim.fit*, but may be used to construct a control argument for either function.

**Usage**

```r
zim.control(dist = c("zip", "znb"), method = c("EM-NR", "EM-FS"),
            type = c("solve", "ginv"), robust = FALSE, trace = FALSE,
            start = NULL, minit = 10, maxit = 10000, epsilon = 1e-08)
```

**Arguments**

- `dist` count model family.
- `method` algorithm for parameter estimation.
- `type` type of matrix inverse.
- `robust` logical; if TRUE, robust standard errors will be calculated.
- `trace` logical; if TRUE, display iteration history.
- `start` initial parameter values.
- `minit` minimum number of iterations.
- `maxit` maximum number of iterations.
- `epsilon` positive convergence tolerance.

**See Also**

*zim, zim.fit*

---

**zim.fit**  
*Fitter Function for Zero-Inflated Models*

**Description**

*zim.fit* is the basic computing engine called by *zim* used to fit zero-inflated models. This should usually *not* be used directly unless by experienced users.

**Usage**

```r
zim.fit(y, X, Z, weights = rep(1, nobs), offset = rep(0, nobs),
        control = zim.control(...), ...)
```
**Arguments**

- **y**
  - response variable.
- **X**
  - design matrix for log-linear part.
- **Z**
  - design matrix for logistic part.
- **weights**
  - an optional vector of 'prior weights' to be used in the fitting process.
- **offset**
  - offset variable
- **control**
  - control arguments from `zim.control`.
- ...
  - additional arguments.

**See Also**

`zim`, `zim.control`

---

**ZINB**

*The Zero-Inflated Negative Binomial Distribution*

**Description**

Density, distribution function, quantile function and random generation for the zero-inflated negative binomial (ZINB) distribution with parameters k, lambda, and omega.

**Usage**

```
dzinb(x, k, lambda, omega, log = FALSE)
pzinb(q, k, lambda, omega, lower.tail = TRUE, log.p = FALSE)
qzinb(p, k, lambda, omega, lower.tail = TRUE, log.p = FALSE)
rzinb(n, k, lambda, omega)
```

**Arguments**

- **x, q**
  - vector of quantiles.
- **p**
  - vector of probabilities.
- **n**
  - number of random values to return.
- **k**
  - dispersion parameter.
- **lambda**
  - vector of (non-negative) means.
- **omega**
  - zero-inflation parameter.
- **log, log.p**
  - logical; if TRUE, probabilities p are given as log(p).
- **lower.tail**
  - logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].

**Value**

dzinb gives the density, pzinb gives the distribution function, qzinb gives the quantile function, and rzinb generates random deviates.
ZIP

The Zero-Inflated Poisson Distribution

Description
Density, distribution function, quantile function and random generation for the zero-inflated Poisson (ZIP) distribution with parameters lambda and omega.

Usage

dzip(x, lambda, omega, log = FALSE)
pzip(q, lambda, omega, lower.tail = TRUE, log.p = FALSE)
qzip(p, lambda, omega, lower.tail = TRUE, log.p = FALSE)
rzip(n, lambda, omega)

Arguments

x, q
vector of quantiles.

p
vector of probabilities.

n
number of random values to return.

lambda
vector of (non-negative) means.

omega
zero-inflation parameter.

log, log.p
logical; if TRUE, probabilities p are given as log(p).

lower.tail
logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].

Value
dzip gives the density, pzip gives the distribution function, qzip gives the quantile function, and rzip generates random deviates.

See Also
dzinb, pzmb, qzinb, and rzinb for the zero-inflated negative binomial (ZINB) distribution.
Examples

dzip(x = 0:10, lambda = 1, omega = 0.5)
pzip(q = c(1, 5, 9), lambda = 1, omega = 0.5)
qzip(p = c(0.25, 0.50, 0.75), lambda = 1, omega = 0.5)
set.seed(123)
rzip(n = 100, lambda = 1, omega = 0.5)
Index

+ Topic datasets
  injury, 8
  syph, 9
+ Topic distribution
  ZINB, 13
  ZIP, 14
+ Topic misc
  bshift, 3
  pvalue, 8
+ Topic package
  ZIM-package, 2
+ Topic regression
  dzim, 4
  dzim.control, 4
  dzim.filter, 5
  dzim.fit, 6
  dzim.sim, 7
  dzim.smooth, 7
  zim, 11
  zim.control, 12
  zim.fit, 12
  bshift, 3
  dzim, 2, 4
  dzim.control, 4, 5
  dzim.filter, 5, 6
  dzim.fit, 6, 7
  dzim.sim, 6, 7
  dzim.smooth, 6, 7
  dzinb, 14
  dzinb (ZINB), 13
  dzip, 14
  dzip (ZIP), 14
  formula, 4, 11
  injury, 8
  lag, 3
  pvalue, 8
  pzinb, 14
  pzinb (ZINB), 13
  pzip, 14
  pzip (ZIP), 14
  qzinb, 14
  qzinb (ZINB), 13
  qzip, 14
  qzip (ZIP), 14
  rzinb, 14
  rzinb (ZINB), 13
  rzip, 14
  rzip (ZIP), 14
  syph, 9
  zeroinfl, 2, 11
  ZIM (ZIM-package), 2
  zim, 2, 11, 12, 13
  ZIM-package, 2
  zim.control, 11, 12, 13
  zim.fit, 11, 12, 13
  ZINB, 13
  ZIP, 14
  zlag, 3