Package ‘ZIM’

February 6, 2017

Type Package
Title Zero-Inflated Models for Count Time Series with Excess Zeros
Version 1.0.3
Date 2017-02-05
Author Ming Yang, Gideon K. D. Zamba, and Joseph E. Cavanaugh
Maintainer Ming Yang <mingyang@biostatstudio.com>
Description Fits observation-driven and parameter-driven models for zero-inflated time series.
License GPL-3
Depends MASS
NeedsCompilation no
Repository CRAN
Date/Publication 2017-02-06 10:25:36

R topics documented:

ZIM-package ................................................................. 2
bshift ................................................................. 3
dzim ................................................................. 3
dzim.control ........................................................... 4
dzim.filter ........................................................... 5
dzim.fit ............................................................... 5
dzim.plot ............................................................. 6
dzim.sim ............................................................... 6
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, 2014+). The main function for fitting observation-driven models is \texttt{zim}, and the main function for fitting parameter-driven models is \texttt{dzim}.

Note

The observation-driven models for zero-inflated count time series can also be fit using the function \texttt{zeroinfl} from the \texttt{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.

Author(s)

Ming Yang, Gideon K. D. Zamba, and Joseph E. Cavanaugh

Maintainer: Ming Yang <mingyang at hsp.harvard.edu>

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**

```
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 \texttt{dzim.control} 
... additional arguments

\textbf{See Also}

\texttt{dzim.fit, dzim.filter, dzim.smooth}\texttt{, dzim.control, dzim.sim, dzim.plot}

\begin{longtable}{ll}
\texttt{dzim.control} & \textit{Auxiliary for Controlling DZIM Fitting} \\
\end{longtable}

\textbf{Description}

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

\textbf{Usage}

\begin{verbatim}
dzm.control(dist = c("poisson", "nb", "zip", "zinb"), trace = FALSE, 
             start = NULL, order = 1, mu0 = rep(0, order), Sigma0 = diag(1, order), 
             N = 1000, R = 1000, niter = 500)
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{dist} count model family
  \item \texttt{trace} logical; if \texttt{TRUE}, display iteration history.
  \item \texttt{start} initial parameter values.
  \item \texttt{order} autoregressive order.
  \item \texttt{mu0} mean vector for initial state.
  \item \texttt{Sigma0} covariance matrix for initial state.
  \item \texttt{N} number of particles in particle filtering.
  \item \texttt{R} number of replications in particle smoothing.
  \item \texttt{niter} number of iterations.
\end{itemize}

\textbf{Note}

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

\textbf{See Also}

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


\textbf{dzim.filter} \hspace{1cm} \textit{Particle Filtering for DZIM}

\section*{Description}

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

\section*{Usage}

\begin{verbatim}
dzim.filter(y, X, w, para, control)
\end{verbatim}

\section*{Arguments}

- \texttt{y} \hspace{.5cm} response variable.
- \texttt{X} \hspace{.5cm} design matrix.
- \texttt{w} \hspace{.5cm} \texttt{log(w)} is used as an offset variable in the linear predictor.
- \texttt{para} \hspace{.5cm} model parameters.
- \texttt{control} \hspace{.5cm} control arguments.

\section*{References}


\section*{See Also}

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

\textbf{dzim.fit} \hspace{1cm} \textit{Fitter Function for Dynamic Zero-Inflated Models}

\section*{Description}

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

\section*{Usage}

\begin{verbatim}
dzim.fit(y, X, offset = rep(0, n), control = dzim.control(...), ...)
\end{verbatim}
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

`dzim.sim(X, w, omega, k, beta, phi, sigma, mu0, Sigma0)`
dzim.smooth

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

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)

Function to Compute P-value.

Description

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

Usage

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, alternative = "greater")
### 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.

<table>
<thead>
<tr>
<th>year</th>
<th>week</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1</td>
<td>United States</td>
<td></td>
</tr>
<tr>
<td>a2</td>
<td>New England</td>
<td></td>
</tr>
<tr>
<td>a3</td>
<td>Connecticut</td>
<td></td>
</tr>
<tr>
<td>a4</td>
<td>Maine</td>
<td></td>
</tr>
<tr>
<td>a5</td>
<td>Massachusetts</td>
<td></td>
</tr>
<tr>
<td>a6</td>
<td>New Hampshire</td>
<td></td>
</tr>
<tr>
<td>a7</td>
<td>Rhode Island</td>
<td></td>
</tr>
<tr>
<td>a8</td>
<td>Vermont</td>
<td></td>
</tr>
<tr>
<td>a9</td>
<td>Mid. Atlantic</td>
<td></td>
</tr>
<tr>
<td>a10</td>
<td>New Jersey</td>
<td></td>
</tr>
<tr>
<td>a11</td>
<td>New York (Upstate)</td>
<td></td>
</tr>
<tr>
<td>a12</td>
<td>New York City</td>
<td></td>
</tr>
<tr>
<td>a13</td>
<td>Pennsylvania</td>
<td></td>
</tr>
<tr>
<td>a14</td>
<td>E.N. Central</td>
<td></td>
</tr>
<tr>
<td>a15</td>
<td>Illinois</td>
<td></td>
</tr>
<tr>
<td>a16</td>
<td>Indiana</td>
<td></td>
</tr>
<tr>
<td>a17</td>
<td>Michigan</td>
<td></td>
</tr>
<tr>
<td>a18</td>
<td>Ohio</td>
<td></td>
</tr>
<tr>
<td>a19</td>
<td>Wisconsin</td>
<td></td>
</tr>
<tr>
<td>a20</td>
<td>W.N. Central</td>
<td></td>
</tr>
<tr>
<td>a21</td>
<td>Iowa</td>
<td></td>
</tr>
<tr>
<td>a22</td>
<td>Kansas</td>
<td></td>
</tr>
<tr>
<td>a23</td>
<td>Minnesota</td>
<td></td>
</tr>
<tr>
<td>a24</td>
<td>Missouri</td>
<td></td>
</tr>
<tr>
<td>a25</td>
<td>Nebraska</td>
<td></td>
</tr>
<tr>
<td>a26</td>
<td>North Dakota</td>
<td></td>
</tr>
<tr>
<td>a27</td>
<td>South Dakota</td>
<td></td>
</tr>
<tr>
<td>a28</td>
<td>S. Atlantic</td>
<td></td>
</tr>
<tr>
<td>a29</td>
<td>Delaware</td>
<td></td>
</tr>
<tr>
<td>a30</td>
<td>District of Columbia</td>
<td></td>
</tr>
<tr>
<td>a31</td>
<td>Florida</td>
<td></td>
</tr>
<tr>
<td>a32</td>
<td>Georgia</td>
<td></td>
</tr>
<tr>
<td>a33</td>
<td>Maryland</td>
<td></td>
</tr>
<tr>
<td>a34</td>
<td>North Carolina</td>
<td></td>
</tr>
</tbody>
</table>
a35  South Carolina
a36  Virginia
a37  West Virginia
a38  **E.S. Central**
a39  Alabama
a40  Kentucky
a41  Mississippi
a42  Tennessee
a43  **W.S. Central**
a44  Arkansas
a45  Louisiana
a46  Oklahoma
a47  Texas
a48  **Mountian**
a49  Arizona
a50  Colorado
a51  Idaho
a52  Montana
a53  Nevada
a54  New Mexico
a55  Utah
a56  Wyoming
a57  **Pacific**
a58  Alaska
a59  California
a60  Hawaii
a61  Oregon
a62  Washington
a63  American Samoa
a64  C.N.M.I.
a65  Guam
a66  Peurto Rico
a67  U.S. Virgin Islands

**Note**

C.N.M.I.: Commonwealth of Northern Mariana Islands.

**Source**


**Examples**

```r
data(syph)
plot(ts(syph$a33), main = "Maryland")
```
Description

zim is used to fit zero-inflated models.

Usage

\[
zim(formula, data, subset, na.action, weights = 1, offset = 0, \]
control = zim.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.
- **...**: additional arguments.

Note

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

See Also

zim.fit, 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

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

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`

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)`: gives the density.
- `pzinb(q, k, lambda, omega, lower.tail = TRUE, log.p = FALSE)`: gives the distribution function.
- `qzinb(p, k, lambda, omega, lower.tail = TRUE, log.p = FALSE)`: gives the quantile function.
- `rzinb(n, k, lambda, omega)`: generates random deviates.

Arguments

- `x`: vector of quantiles.
- `q`: vector of probabilities.
- `p`: vector of (non-negative) means.
- `k`: dispersion parameter.
- `lambd`: 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.
See Also
dzip, pzip, qzip, and rzip for the zero-inflated Poisson (ZIP) distribution.

Examples
dzinb(x = 0:10, k = 1, lambda = 1, omega = 0.5)
pzinb(q = c(1, 5, 9), k = 1, lambda = 1, omega = 0.5)
qzinb(p = c(0.25, 0.50, 0.75), k = 1, lambda = 1, omega = 0.5)
rzinb(n = 100, k = 1, lambda = 1, omega = 0.5)

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, pzinb, 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)
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, 3
  dzim.control, 4
  dzim.filter, 5
  dzim.fit, 5
  dzim.sim, 6
  dzim.smooth, 7
  zim, 11
  zim.control, 12
  zim.fit, 12
  bshift, 3
  dzim, 2, 3, 4–7
  dzim.control, 4, 4, 5–7
  dzim.filter, 4, 5, 6, 7
  dzim.fit, 4, 5, 5, 6, 7
  dzim.plot, 4–6, 6, 7
  dzim.sim, 4–6, 6, 7
  dzim.smooth, 4–7, 7
  dzinb, 14
  dzinb (ZINB), 13
  dzip, 14
  dzip (ZIP), 14
  formula, 3, 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, 11, 12, 13
  ZIM-package, 2
  zim.control, 11, 12, 13
  zim.fit, 11, 12, 12
  ZINB, 13
  ZIP, 14
  zlag, 3