Package ‘hurdlr’

July 2, 2017

Version 0.1

Title Zero-Inflated and Hurdle Modelling Using Bayesian Inference

Description When considering count data, it is often the case that many more zero counts than would be expected of some given distribution are observed. It is well established that data such as this can be reliably modelled using zero-inflated or hurdle distributions, both of which may be applied using the functions in this package. Bayesian analysis methods are used to best model problematic count data that cannot be fit to any typical distribution. The package functions are flexible and versatile, and can be applied to varying count distributions, parameter estimation with or without explanatory variable information, and are able to allow for multiple hurdles as it is also not uncommon that count data have an abundance of large-number observations which would be considered outliers of the typical distribution. In lieu of throwing out data or misspecifying the typical distribution, these extreme observations can be applied to a second, extreme distribution. With the given functions of this package, such a two-hurdle model may be easily specified in order to best manage data that is both zero-inflated and over-dispersed.

Date 2017-07-01

Depends R (>= 3.3.0)

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

NeedsCompilation no

Author Earvin Balderama [aut, cre],
Taylor Trippe [aut]

Maintainer Earvin Balderama <ebalderama@luc.edu>

Repository CRAN

Date/Publication 2017-07-02 00:04:08 UTC

R topics documented:

  dist_ll .......................... 2
dist_ll

Distributional Likelihood for Hurdle Model Count Data Regression

Description

dist_ll is the data likelihood function for hurdle model regression using hurdle.

Usage

dist_ll(y, hurd = Inf, lam = NULL, size = 1, mu = NULL, xi = NULL,
        sigma = NULL, dist = c("poisson", "nb", "lognormal", "gpd"), g.x = F,
        log = T)

Arguments

  y numeric response vector.
  hurd numeric threshold for 'extreme' observations of two-hurdle models. Inf for one-hurdle models.
  lam current value for the poisson likelihood lambda parameter.
  size size parameter for negative binomial likelihood distributions.
  mu current value for the negative binomial or log normal likelihood mu parameter.
  xi current value for the generalized pareto likelihood xi parameter.
  sigma current value for the generalized pareto likelihood sigma parameter.
  dist character specification of response distribution.
  g.x logical operator. TRUE if operating within the third component of the likelihood function (the likelihood of 'extreme' observations).
  log logical operator. if TRUE, probabilities p are given as log(p).
Details
Currently, Poisson, Negative Binomial, log-Normal, and Generalized Pareto distributions are available.

Value
The log-likelihood of the zero-inflated Poisson fit for the current iteration of the MCMC algorithm.

Author(s)
Taylor Trippe <<ttrippe@luc.edu>>
Earvin Balderama <<ebalderama@luc.edu>>

See Also
hurdle

---

GenPareto  The Generalized Pareto Distribution

Description
Density, distribution function, quantile function and random generation for the Generalized Pareto distribution with parameters \( \mu \), \( \sigma \), and \( \xi \).

Usage
\[
\text{dgpd}(x, \mu = 0, \sigma = 1, \xi = 1, \text{log} = F) \\
\text{mgpd}(x, \mu = 0, \sigma = 1, \xi = 1, \text{log} = F) \\
\text{pgpd}(q, \mu = 0, \sigma = 1, \xi = 1, \text{lower.tail} = T) \\
\text{qgpd}(p, \mu = 0, \sigma = 1, \xi = 1, \text{lower.tail} = T) \\
\text{rgpd}(n, \mu = 0, \sigma = 1, \xi = 1)
\]

Arguments
- \( x, q \)  vector of quantiles.
- \( \mu \)  location parameter.
- \( \sigma \)  (non-negative) scale parameter.
- \( \xi \)  shape parameter.
- \( \text{log} \)  logical; if \( \text{TRUE} \), probabilities \( p \) are given as \( \log(p) \).
- \( \text{lower.tail} \)  logical; if \( \text{TRUE} \), probabilities are \( P[X \leq x] \), otherwise, \( P[X > x] \).
- \( p \)  numeric predictor matrix.
- \( n \)  number of random values to return.
Details

The generalized pareto distribution has density

\[ f(x) = \frac{\sigma^\xi}{(\sigma + \xi(x - \mu))^{\xi+1}} \]

Value

dgpd gives the continuous density, pgpd gives the distribution function, qgpd gives the quantile function, and rgpd generates random deviates.
mgpd gives a probability mass function for a discretized version of GPD.

Author(s)

Taylor Trippe <<ttrippe@luc.edu>>
Earvin Balderama <<ebalderama@luc.edu>>

Examples

dexp(1,rate=.5) #Exp(rate) equivalent to gpd with mu=0 AND xi=0, and sigma=1/rate.
dgpd(1, mu=0, sigma=2, xi=0) #cannot take xi=0.
dgpd(1, mu=0, sigma=2, xi=0.000001) #but can get close.

##"mass" function of GPD
mgpd(8) == pgpd(8.5) - pgpd(7.5)

hurdle is used to fit single or double-hurdle regression models to count data via Bayesian inference.

Usage

hurdle(y, x = NULL, hurd = Inf, dist = c("poisson", "nb", "lognormal"),
       dist.2 = c("gpd", "poisson", "lognormal", "nb"),
       control = hurdle_control(), iters = 1000, burn = 500, nthin = 1,
       plots = FALSE, progress.bar = TRUE)

Arguments

<table>
<thead>
<tr>
<th>y</th>
<th>numeric response vector.</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>numeric predictor matrix.</td>
</tr>
<tr>
<td>hurd</td>
<td>numeric threshold for 'extreme' observations of two-hurdle models. Inf for one-hurdle models.</td>
</tr>
</tbody>
</table>
hurdle

- `dist` character specification of response distribution.
- `dist.2` character specification of response distribution for 'extreme' observations of two-hurdle models.
- `control` list of parameters for controlling the fitting process, specified by `hurdle_control`.
- `iters` number of iterations for the Markov chain to run.
- `burn` numeric burn-in length.
- `nthin` numeric thinning rate.
- `plots` logical operator. TRUE to output plots.
- `progress.bar` logical operator. TRUE to print progress bar.

**Details**

Setting `dist` and `dist.2` to be the same distribution creates a single `dist`-hurdle model, not a double-hurdle model. However, this is being considered in future package updates.

**Value**

`hurdle` returns a list which includes the items

- `pD` measure of model dimensionality $p_D$ where $p_D = \bar{D} - D(\bar{\theta})$ is the "meanposteriordeviance−devianceofposteriormeans".
- `DIC` Deviance Information Criterion where $DIC = \bar{D} - p_D$
- `PPO` Posterior Predictive Ordinate (PPO) measure of fit
- `CPO` Conditional Predictive Ordinate (CPO) measure of fit
- `pars.means` posterior mean(s) of third-component parameter(s) if hurdle != Inf
- `ll.means` posterior means of the log-likelihood distributions of all model components
- `beta.means` posterior means regression coefficients
- `dev` posterior deviation where $D = -2LogL$
- `beta` posterior distributions of regression coefficients
- `pars` posterior distribution(s) of third-component parameter(s) if hurdle != Inf

**Author(s)**

Taylor Trippe <<ttrippe@luc.edu>>  
Earvin Balderama <<ebalderama@luc.edu>>

**Examples**

```
#Generate some data:
p=0.5; q=0.25; lam=3;  
mu=10; sigma=7; xi=0.75;  
n=200

set.seed(2016)
y <- rbinom(n,1,p)
```
nz <- sum(1-y)
extremes <- rbinom(sum(y),1,q)
ne <- sum(extremes)
nt <- n-nz-ne
yt <- sample(mu-1.nt,replace=TRUE,prob=dpois((mu-1),(3))/((ppois(mu-1,lambda)-ppois(0,lambda)))
yz <- round(rgpd(nz,mu,sigma,xi))
y[y==1] <- c(yt,yz)
g <- hurdle(y)

hurdle_control
Control Parameters for Hurdle Model Count Data Regression

Description

Various parameters for fitting control of hurdle model regression using hurdle.

Usage

hurdle_control(a = 1, b = 1, size = 1, beta_prior_mean = 0,
beta_prior_sd = 1000, beta_tune = 1, pars_tune = 0.2, lam-start = 1,
mu.start = 1, sigma.start = 1, xi.start = 1)

Arguments

a shape parameter for gamma prior distributions.
b rate parameter for gamma prior distributions.
size size parameter for negative binomial likelihood distributions.
beta_prior_mean mu parameter for normal prior distributions.
beta_prior_sd standard deviation for normal prior distributions.
beta_tune Markov-chain tuning for regression coefficient estimation.
pars_tune Markov chain tuning for parameter estimation of 'extreme' observations distribution.
lam_start initial value for the poisson likelihood lambda parameter.
**loglik_zinb**

- **mu.start**: initial value for the negative binomial or log normal likelihood mu parameter.
- **sigma.start**: initial value for the generalized pareto likelihood sigma parameter.
- **xi.start**: initial value for the generalized pareto likelihood xi parameter.

**Value**

A list of all input values.

**Author(s)**

Taylor Trippe <<ttrippe@luc.edu>>
Earvin Balderama <<ebalderama@luc.edu>>

**See Also**

hurdle

---

**loglik_zinb**    
Zero-inflated Negative Binomial Data Likelihood

**Description**

Data likelihood function for zero-inflated negative binomial model regression using zero_nb.

**Usage**

loglik_zinb(y, z, mu, size, p)

**Arguments**

- **y**: numeric response vector.
- **z**: vector of binary operators. z == 0 for observations considered belonging to the negative binomial distribution, z == 1 for observations considered to be ’extra’ zeros.
- **mu**: current value for the negative binomial likelihood mu parameter.
- **size**: size parameter for negative binomial distribution.
- **p**: vector of ’extra’ zero-count probabilities.

**Value**

The log-likelihood of the zero-inflated negative binomial fit for the current iteration of the MCMC algorithm.

**Author(s)**

Taylor Trippe <<ttrippe@luc.edu>>
Earvin Balderama <<ebalderama@luc.edu>>
See Also

zero_nb

---

loglik_zip

Zero-inflated Poisson Data Likelihood

Description

Data likelihood function for zero-inflated Poisson model regression using `zero_poisson`.

Usage

`loglik_zip(y, z, lam, p)`

Arguments

- `y`: numeric response vector.
- `z`: vector of binary operators. `z == 0` for observations considered belonging to the negative binomial distribution, `z == 1` for observations considered to be 'extra' zeros.
- `lam`: current value for the Poisson likelihood lambda parameter.
- `p`: vector of 'extra' zero-count probabilities.

Value

The log-likelihood of the zero-inflated Poisson fit for the current iteration of the MCMC algorithm.

Author(s)

Taylor Trippe <<ttrippe@luc.edu>>
Earvin Balderama <<ebalderama@luc.edu>>

See Also

`zero_poisson`
**mlnorm**  
*Density Function for Discrete Log Normal Distribution*

**Description**
Density function of the discrete log normal distribution whose logarithm has mean equal to meanlog and standard deviation equal to sdlog.

**Usage**
mlnorm(x, meanlog = 0, sdlog = 1, log = T)

**Arguments**
- x: vector of quantiles.
- meanlog: mean of the distribution on the log scale.
- sdlog: standard deviation of the distribution on the log scale.
- log: logical; if TRUE, probabilities p are given as log(p).

**Value**
Discrete log-normal distributional density.

**Author(s)**
Taylor Trippe <<ttrippe@luc.edu>>
Earvin Balderama <<ebalderama@luc.edu>>

---

**PE**  
*Extreme Count Probability Likelihood*

**Description**
PE is used to calculate the likelihood of a user-defined 'extreme' value count observation in a double-hurdle regression model.

**Usage**
PE(p, q, log = T)

**Arguments**
- p: vector of zero-count probabilities.
- q: vector of 'extreme' count probabilities.
- log: logical operator. If TRUE, probabilities p and q are given as log(p), log(q).
**Value**

A vector of probabilities.

**Author(s)**

Taylor Trippe <<ttrippe@luc.edu>>
Earvin Balderama <<ebalderama@luc.edu>>

**See Also**

hurdle

---

**PT**

*Typical Count Probability Likelihood*

**Description**

PT is used to calculate the likelihood of a user-defined 'typical' value count observation in a double-hurdle regression model.

**Usage**

\[ \text{PT}(p, q, \log = T) \]

**Arguments**

- \( p \) vector of zero-count probabilities.
- \( q \) vector of 'typical' count probabilities.
- \( \log \) logical operator. If TRUE, probabilities \( p \) and \( q \) are given as \( \log(p) \), \( \log(q) \).

**Value**

A vector of probabilities.

**Author(s)**

Taylor Trippe <<ttrippe@luc.edu>>
Earvin Balderama <<ebalderama@luc.edu>>

**See Also**

hurdle
PZ

Zero Count Probability Likelihood

Description

PZ is used to calculate the likelihood of a zero-value count observation in a single or double-hurdle regression model.

Usage

PZ(p, log = T)

Arguments

p vector of zero-count probabilities.
log logical operator. If TRUE, probabilities p and q are given as log(p), log(q).

Value

A vector of probabilities.

Author(s)

Taylor Trippe <ttrippe@luc.edu>
Earvin Balderama <ebalderama@luc.edu>

See Also

hurdle

update_beta

MCMC Second-Component Parameter Update Function for Hurdle Model Count Data Regression

Description

MCMC algorithm for updating the second-component likelihood parameters in hurdle model regression using hurdle.

Usage

update_beta(y, x, hurd, dist, like.part, beta.prior.mean, beta.prior.sd, beta, XB, beta.acc, beta.tune, g.x = F)
**Arguments**

- $y$: numeric response vector of observations within the bounds of the second component of the likelihood function, $y[0 < y \& y < \text{hurd}]$.
- $x$: optional numeric predictor matrix for response observations within the bounds of the second component of the likelihood function, $y[0 < y \& y < \text{hurd}]$.
- $\text{hurd}$: numeric threshold for ‘extreme’ observations of two-hurdle models.
- $\text{dist}$: character specification of response distribution for the third component of the likelihood function.
- $\text{like.part}$: numeric vector of the current third-component likelihood values.
- $\text{beta.prior.mean}$: mu parameter for normal prior distributions.
- $\text{beta.prior.sd}$: standard deviation for normal prior distributions.
- $\text{beta}$: numeric matrix of current regression coefficient parameter values.
- $\text{XB}$: $x \ast \text{beta}[1]$ product matrix for response observations within the bounds of the second component of the likelihood function, $y[0 < y \& y < \text{hurd}]$.
- $\text{beta.acc}$: numeric matrix of current MCMC acceptance rates for regression coefficient parameters.
- $\text{beta.tune}$: numeric matrix of current MCMC tuning values for regression coefficient estimation.
- $g.x$: logical operator. TRUE if operating within the third component of the likelihood function (the likelihood of ‘extreme’ observations).

**Value**

A list of MCMC-updated regression coefficients for the estimation of the second-component likelihood parameter as well as each coefficient’s MCMC acceptance ratio.

**Author(s)**

Taylor Trippe <<ttrippe@luc.edu>>
Earvin Balderama <<ebalderama@luc.edu>>

**See Also**

- hurdle
- dist_ll
update_pars

MCMC Third-Component Parameter Update Function for Hurdle Model Count Data Regression

Description

MCMC algorithm for updating the third-component likelihood parameters in hurdle model regression using hurdle.

Usage

update_pars(y, hurd, dist, like.part, a, b, size, lam, mu, xi, sigma, lam.acc, mu.acc, xi.acc, sigma.acc, lam.tune, mu.tune, xi.tune, sigma.tune, g.x = F)

Arguments

- **y**: numeric response vector of observations within the bounds of the third component of the likelihood function, \( y[y \geq \text{hurd}] \).
- **hurd**: numeric threshold for ‘extreme’ observations of two-hurdle models.
- **dist**: character specification of response distribution for the third component of the likelihood function.
- **like.part**: numeric vector of the current third-component likelihood values.
- **a**: shape parameter for gamma prior distributions.
- **b**: rate parameter for gamma prior distributions.
- **size**: size parameter for negative binomial likelihood distributions.
- **lam**: current value for the poisson likelihood lambda parameter.
- **mu**: current value for the negative binomial or log normal likelihood mu parameter.
- **xi**: current value for the generalized pareto likelihood xi parameter.
- **sigma**: current value for the generalized pareto likelihood sigma parameter.
- **lam.acc, mu.acc, xi.acc, sigma.acc**: current MCMC values for third-component parameter acceptance rates.
- **lam.tune, mu.tune, xi.tune, sigma.tune**: current MCMC tuning values for each third-component parameter.
- **g.x**: logical operator. TRUE if operating within the third component of the likelihood function (the likelihood of ‘extreme’ observations).

Value

A list of MCMC-updated likelihood estimator(s) for the third-component parameter(s) and each parameter’s MCMC acceptance ratio.

Author(s)

Taylor Trippe <<ttrippe@luc.edu>>
Earvin Balderama <<ebalderama@luc.edu>>
update_probs

MCMC Probability Update Function for Hurdle Model Count Data Regression

Description

MCMC algorithm for updating the likelihood probabilities in hurdle model regression using hurdle.

Usage

update_probs(y, x, hurd, p, q, beta.prior.mean, beta.prior.sd, pZ, pT, pE, beta, XB2, XB3, beta.acc, beta.tune)

Arguments

y           numeric response vector.
x           optional numeric predictor matrix.
hurd        numeric threshold for 'extreme' observations of two-hurdle models.
p           numeric vector of current 'p' probability parameter values for zero-value observations.
q           numeric vector of current 'q' probability parameter values for 'extreme' observations.
beta.prior.mean
            mu parameter for normal prior distributions.
beta.prior.sd
            standard deviation for normal prior distributions.
pZ          numeric vector of current 'zero probability' likelihood values.
pT          numeric vector of current 'typical probability' likelihood values.
pE          numeric vector of current 'extreme probability' likelihood values.
beta        numeric matrix of current regression coefficient parameter values.
XB2         $x \times beta[, 2]$ product matrix.
XB3         $x \times beta[, 3]$ product matrix.
beta.acc    numeric matrix of current MCMC acceptance rates for regression coefficient parameters.
beta.tune   numeric matrix of current MCMC tuning values for regression coefficient estimation.
Value

A list of MCMC-updated regression coefficients for the estimation of the parameters 'p' (the probability of a zero-value observation) and 'q' (the probability of an 'extreme' observation) as well as each coefficient’s MCMC acceptance ratio.

Author(s)

Taylor Trippe <<ttrippe@luc.edu>>
Earvin Balderama <<ebalderama@luc.edu>>

See Also

hurdle
dist_ll

---

zero_nb	Zero-Inflated Negative Binomial Regression Model

Description

zero_nb is used to fit zero-inflated negative binomial regression models to count data via Bayesian inference.

Usage

```r
zero_nb(y, x, size, a = 1, b = 1, mu.start = 1, beta.prior.mean = 0,
    beta.prior.sd = 1, iters = 1000, burn = 500, nthin = 1, plots = T,
    progress.bar = T)
```

Arguments

- `y`: numeric response vector.
- `x`: numeric predictor matrix.
- `size`: size parameter for negative binomial likelihood distributions.
- `a`: shape parameter for gamma prior distributions.
- `b`: rate parameter for gamma prior distributions.
- `mu.start`: initial value for mu parameter.
- `beta.prior.mean`: mu parameter for normal prior distributions.
- `beta.prior.sd`: standard deviation for normal prior distributions.
- `iters`: number of iterations for the Markov chain to run.
- `burn`: numeric burn-in length.
- `nthin`: numeric thinning rate.
- `plots`: logical operator. TRUE to output plots.
- `progress.bar`: logical operator. TRUE to print progress bar.
Details

Fits a zero-inflated negative binomial (ZINB) model.

Value

zero_nb returns a list which includes the items

- **mu**: numeric vector; posterior distribution of mu parameter
- **beta**: numeric matrix; posterior distributions of regression coefficients
- **p**: numeric vector; posterior distribution of parameter 'p', the probability of a given zero observation belonging to the model’s zero component
- **ll**: numeric vector; posterior log-likelihood

Author(s)

Taylor Trippe <<ttrippe@luc.edu>>
Earvin Balderama <<ebalderama@luc.edu>>

---

**zero_poison**  
**Zero-Inflated Poisson Regression Model**

Description

`zero_poison` is used to fit zero-inflated poisson regression models to count data via Bayesian inference.

Usage

```r
zero_poison(y, x, a = 1, b = 1, lam.start = 1, beta.prior.mean = 0,
beta.prior.sd = 1, iters = 1000, burn = 500, nthin = 1, plots = T,
progress.bar = T)
```

Arguments

- **y**: numeric response vector.
- **x**: numeric predictor matrix.
- **a**: shape parameter for gamma prior distributions.
- **b**: rate parameter for gamma prior distributions.
- **lam.start**: initial value for lambda parameter.
- **beta.prior.mean**: mu parameter for normal prior distributions.
- **beta.prior.sd**: standard deviation for normal prior distributions.
- **iters**: number of iterations for the Markov chain to run.
- **burn**: numeric burn-in length.
- **nthin**: numeric thinning rate.
- **plots**: logical operator. TRUE to output plots.
- **progress.bar**: logical operator. TRUE to print progress bar.
Details

Fits a zero-inflated Poisson (ZIP) model.

Value

zero_poisson returns a list which includes the items

- **lam** numeric vector; posterior distribution of lambda parameter
- **beta** numeric matrix; posterior distributions of regression coefficients
- **p** numeric vector; posterior distribution of parameter 'p', the probability of a given zero observation belonging to the model's zero component
- **ll** numeric vector; posterior log-likelihood

Author(s)

Taylor Trippe <<ttrippe@luc.edu>>
Earvin Balderama <<ebalderama@luc.edu>>
Index

dgpd (GenPareto), 3
dist_ll, 2, 12, 14, 15

GenPareto, 3

hurdle, 2, 3, 4, 6, 7, 10–15
hurdle_control, 5, 6

loglik_zinb, 7
loglik_zip, 8

mgpd (GenPareto), 3
mlnorm, 9

PE, 9
pgpd (GenPareto), 3
PT, 10
PZ, 11

qgpd (GenPareto), 3

rgpd (GenPareto), 3

update_beta, 11
update_pars, 13
update_probs, 14

zero_nb, 7, 8, 15
zero_poisson, 8, 16