Package 'AZIAD'

August 14, 2022

Title Analyzing Zero-Inflated and Zero-Altered Data

Version 0.0.2

Maintainer Niloufar Dousti Mousavi <sdoust2@uic.edu>

Description Computes maximum likelihood estimate of general, zero inflated, and zero altered models for discrete and continuous distributions. Computes Kolmogorov-Smirnov (KS) test and likelihood ratio test for general, zero-inflated, and zero-altered data.
It obtains the inverse of fisher information matrix and confidence interval for the parameters of general, zero inflated, and zero altered models. It simulate random deviates from zero inflated or hurdle models to obtain maximum likelihood estimate.
It is based on the work of Aldirawi et al. (2022) <doi:10.1007/s42519-021-00230-y>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.1.9000

Imports extraDistr (>= 1.9.1), methods (>= 4.0.3), rootSolve (>= 1.8.2.1), foreach (>= 1.5.2), doParallel (>= 1.0.16), parallel (>= 4.1.2), QRM(>= 0.3-31), dplyr(>= 1.0.8), stats (>= 4.1.2), rmutil (>= 1.1.5), corpcor (>= 1.6.10), MixAll (>= 1.5.1), matrixcalc (>= 1.0-5), lqmm(>= 1.5.6), base(>= 4.1.2), EnvStats(>= 2.6.0)

Depends R (>= 4.1.0)

NeedsCompilation no

Author Niloufar Dousti Mousavi [aut, cre, cph], Hani Aldirawi [aut, cph], Jie Yang [aut, cph]

Repository CRAN

Date/Publication 2022-08-14 02:20:05 UTC

R topics documented:

`FI.ZI` *Inverse Fisher Information matrix and confidence intervals of the parameters for general, continuous, and discrete zero-inflated or hurdle distributions.*

Description

Computes the inverse of the fisher information matrix for Poisson, geometric, negative binomial, beta binomial, beta negative binomial, normal, lognormal, half normal, and exponential distributions and their zero-inflated and hurdle versions along with the confidence intervals of all parameters in the model.

Usage

```
FI.ZI(x,dist="poisson",r=NULL,p=NULL,alpha1=NULL,alpha2=NULL,
n=NULL,lambda=NULL,mean=NULL,sigma=NULL,lowerbound=0.01,upperbound=10000)
```

Arguments

`x`	A vector of count data. Should be non-negative integers for discrete cases. Random generation for continuous cases.
`dist`	The distribution used to calculate the inverse of fisher information and confidence interval. It can be one of 'poisson','geometric','nb','bb','bnb','normal','halfnormal','lognormal','ex 'zip','zigeom','zinb','zibb','zibnb', 'zinormal','zilognorm','zohalfnorm','ziexp', 'ph','geomh','nbh','bbh','bnbh' which corresponds to general Poisson, geometric, negative binomial, beta binomial, beta negative binomial, normal, log normal, half normal, exponential, Zero-Inflated Poisson, Zero-Inflated geometric, Zero-Inflated negative binomial, Zero-Inflated beta binomial, Zero-Inflated beta negative binomial, Zero-Inflated/hurdle normal, Zero-Inflated/hurdle log normal, Zero-Inflated/hurdle half normal, Zero-Inflated/hurdle exponential, Zero-Hurdle Poisson, Zero-Hurdle geometric, Zero-Hurdle negative binomial, Zero-Hurdle beta binomial, and Zero-Hurdle beta negative binomial distributions, respectively.
`r`	An initial value of the number of success before which m failures are observed, where m is the element of `x`. Must be a positive number, but not required to be an integer.
`p`	An initial value of the probability of success, should be a positive value within (0,1).

alpha1	An initial value for the first shape parameter of beta distribution. Should be a positive number.
alpha2	An initial value for the second shape parameter of beta distribution. Should be a positive number.
n	An initial value of the number of trials. Must be a positive number, but not required to be an integer.
lambda	An initial value of the rate. Must be a positive real number.
mean	An initial value of the mean or expectation.
sigma	An initial value of the standard deviation. Must be a positive real number.
lowerbound	A lower searching bound used in the optimization of likelihood function. Should be a small positive number. The default is 1e-2.
upperbound	An upper searching bound used in the optimization of likelihood function. Should be a large positive number. The default is 1e4.

Details

FI.ZI calculate the inverse of the fisher information matrix and the corresponding confidence interval of the parameter of general, Zero-Inflated, and Zero-Hurdle Poisson, geometric, negative binomial, beta binomial, beta negative binomial, normal, log normal, half normal, and exponential distributions. Note that zero-inflated and hurdle are the same in continuous distributions.

Value

A list containing the inverse of the fisher information matrix and the corresponding 95% confidence interval for all the parameters in the model.

References

- Aldirawi H, Yang J (2022). "Modeling Sparse Data Using MLE with Applications to Microbiome Data." Journal of Statistical Theory and Practice, 16(1), 1–16.

Examples

```
set.seed(111)
N=1000;lambda=10;
x<-stats::rpois(N,lambda=lambda)
FI.ZI(x,lambda=5,dist="poisson")
#$inversefisher
#     lambda
#[1,]  9.896

#$ConfidenceIntervals
#[1]  9.701025 10.090974
set.seed(111)
N=1000;lambda=10;phi=0.4;
x1<-sample.h1(N,lambda=lambda,phi=phi,dist="poisson")
FI.ZI(x1,lambda=4,dist="ph")
#$inversefisher
#       [,1]     [,2]
```

```
#[1,] 0.237679  0.00000
#[2,] 0.000000 16.12686

#$ConfidenceIntervals
#              [,1]       [,2]
#CI of Phi    0.3587835  0.4192165
#CI of lambda 9.6000082 10.0978060
set.seed(289)
N=2000;mean=10;sigma=2;phi=0.4;
x<-sample.zi1(N,phi=phi,mean=mean,sigma=sigma,dist="lognormal")
FI.ZI(x, mean=1,sigma=1, dist="zilognorm")
# $inversefisher
#        [,1]     [,2]     [,3]
#[1,] 0.6313214 0.000000 0.000000
#[2,] 0.0000000 6.698431 0.000000
#[3,] 0.0000000 0.000000 3.349215

#$ConfidenceIntervals
#              [,1]       [,2]
#CI of phi   0.3521776  0.4218224
#CI of mean  9.8860358 10.1128915
#CI of sigma 1.9461552  2.1065664
```

`kstest.A` *The Monte Carlo estimate for the p-value of a discrete KS Test based on zih.mle estimates.*

Description

Computes the Monte Carlo estimate for the p-value of a discrete one-sample Kolmogorov-Smirnov (KS) Test based on zih.mle function estimates for Poisson, geometric, negative binomial, beta binomial, beta negative binomial, normal, log normal, halfnormal, and exponential distributions and their zero-inflated as well as hurdle versions.

Usage

```
kstest.A(x,nsim=200,bootstrap=TRUE,dist='poisson',r=NULL,p=NULL,alpha1=NULL,
alpha2=NULL,n=NULL,lambda=NULL,mean=NULL,sigma=NULL,
lowerbound=1e-2,upperbound=1e4,parallel=FALSE)
```

Arguments

`x`	A vector of count data. Should be non-negative integers for discrete cases. Random generation for continuous cases.
`nsim`	The number of bootstrapped samples or simulated samples generated to compute p-value. If it is not an integer, nsim will be automatically rounded up to the smallest integer that is no less than nsim. Should be greater than 30. Default is 200.

bootstrap	Whether to generate bootstrapped samples or not. See Details. 'TRUE' or any numeric non-zero value indicates the generation of bootstrapped samples. The default is 'TRUE'.
dist	The distribution used as the null hypothesis. Can be one of 'poisson', 'geometric', 'nb', 'nb1', 'bb', 'bb1', 'bnb', 'bnb1', 'normal', 'lognormal', 'halfnormal', 'exponential', 'zip', 'zigeom', 'zinb', 'zibb', zibnb', 'zinormal', 'zilognorm', 'zihalfnorm', 'ziexp', 'ph', 'geomh','nbh','bbh','bnbh', 'normalh', 'lognormh', 'halfnormh', and 'exph' , which corresponds to Poisson, geometric, negative binomial, negative binomial1, beta binomial, beta binomial1, beta negative binomial, beta negative binomial1, normal, half normal, log normal, and exponential distributions and their zero-inflated as well as hurdle version, respectively. Defult is 'poisson'.
r	An initial value of the number of success before which m failures are observed, where m is the element of x. Must be a positive number, but not required to be an integer.
p	An initial value of the probability of success, should be a positive value within (0,1).
alpha1	An initial value for the first shape parameter of beta distribution. Should be a positive number.
alpha2	An initial value for the second shape parameter of beta distribution. Should be a positive number.
n	An initial value of the number of trials. Must be a positive number, but not required to be an integer.
lambda	An initial value of the rate. Must be a positive real number.
mean	An initial value of the mean or expectation.
sigma	An initial value of the standard deviation. Must be a positive real number.
lowerbound	A lower searching bound used in the optimization of likelihood function. Should be a small positive number. The default is 1e-2.
upperbound	An upper searching bound used in the optimization of likelihood function. Should be a large positive number. The default is 1e4.
parallel	whether to use multiple threads for paralleling computation. Default is FALSE. Please aware that it may take longer time to execute the program with parallel=FALSE.

Details

In arguments nsim, bootstrap, dist, if the length is larger than 1, only the first element will be used. For other arguments except for x, the first valid value will be used if the input is not NULL, otherwise some naive sample estimates will be fed into the algorithm. Note that only the initial values that is used in the null distribution dist are needed. For example, with dist=poisson, user should provide a value for lambda but not for other parameters. With an output p-value less than some user-specified significance level, x is very likely from a distribution other than the dist, given the current data. If p-values of more than one distributions are greater than the pre-specified significance level, user may consider a following likelihood ratio test to select a 'better' distribution. The methodology of computing Monte Carlo p-value is taken from Aldirawi et al. (2019) except

changing the zih.mle function and have accurate estimates and adding new discrete and continuous distributions. When bootstrap=TRUE, nsim bootstrapped samples will be generated by resampling x without replacement. Otherwise, nsim samples are simulated from the null distribution with the maximum likelihood estimate of original data x. Then compute the maximum likelihood estimates of nsim bootstrapped or simulated samples, based on which nsim new samples are generated under the null distribution. nsim KS statistics are calculated for the nsim new samples, then the Monte Carlo p-value is resulted from comparing the nsim KS statistics and the statistic of original data x. During the process of computing maximum likelihood estimates, the negative log likelihood function is minimized via basic R function optim with the searching interval decided by lowerbound and upperbound. For large sample sizes we may use kstest.A and for small sample sizes (less that 50 or 100), kstest.B is preferred.

Value

An object of class 'kstest.A' including the following elements:

- x: x used in computation.
- nsim: nsim used in computation.
- bootstrap: bootstrap used in computation.
- dist: dist used in computation.
- lowerbound: lowerbound used in computation.
- upperbound: upperboound used in computation.
- mle_new: A matrix of the maximum likelihood estimates of unknown parameters under the null distribution, using nsim bootstrapped or simulated samples.
- mle_ori: A row vector of the maximum likelihood estimates of unknown parameters under the null distribution, using the original data x.
- pvalue: Monte Carlo p-value of the one-sample KS test.
- N: length of x.
- r: initial value of r used in computation.
- p: initial value of p used in computation.
- alpha1: initial value of alpha1 used in computation.
- alpha2: initial value of alpha2 used in computation.
- lambda: initial value of lambda used in computation.
- n: initial value of n used in computation.
- mean: initial value of mean used in computation.
- sigma: initial value of sigma used in computation.

References

- H. Aldirawi, J. Yang, A. A. Metwally (2019). Identifying Appropriate Probabilistic Models for Sparse Discrete Omics Data, accepted for publication in 2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI).

See Also

lrt.A

Examples

```
set.seed(007)
x1=sample.zi1(2000,phi=0.3,dist='bnb',r=5,alpha=3,alpha2=3)
kstest.A(x1,nsim=200,bootstrap = TRUE,dist= 'zinb')$pvalue      #0
kstest.A(x1,nsim=200,bootstrap = TRUE,dist= 'zibnb')$pvalue     #1
kstest.A(x1,nsim=100,bootstrap = TRUE,dist= 'zibb')$pvalue      #0.03
x2=sample.h1(2000,phi=0.3,dist="normal",mean=10,sigma=2)
kstest.A(x2,nsim=100,bootstrap = TRUE,dist= 'normalh')$pvalue   #1
## Not run: kstest.A(x2,nsim=100,bootstrap = TRUE,dist= 'halfnormh')$pvalue #0.04
```

---

kstest.B *The Monte Carlo estimate for the p-value of a discrete KS test based on nested bootstrapped samples*

Description

Computes the Monte Carlo estimate for the p-value of a discrete one-sample Kolmogorov-Smirnov (KS) test based on nested bootstrapped samples for Poisson, geometric, negative binomial, beta binomial, beta negative binomial, normal, log normal, halfnormal, and exponential distributions and their zero-inflated as well as hurdle versions.

Usage

```
kstest.B(x,nsim=200,bootstrap=TRUE,dist="poisson",
r=NULL,p=NULL,alpha1=NULL,alpha2=NULL,n=NULL,lambda=NULL,mean=NULL,sigma=NULL,
lowerbound = 0.01, upperbound = 10000, parallel = FALSE)
```

Arguments

x	A vector of count data. Should be non-negative integers for discrete cases. Random generation for continuous cases.
nsim	The number of bootstrapped samples or simulated samples generated to compute p-value. If it is not an integer, nsim will be automatically rounded up to the smallest integer that is no less than nsim. Should be greater than 30. Default is 200.
bootstrap	Whether to generate bootstrapped samples or not. See Details. 'TRUE' or any numeric non-zero value indicates the generation of bootstrapped samples. The default is 'TRUE'.
dist	The distribution used as the null hypothesis. Can be one of 'poisson', 'geometric', 'nb', 'nb1', 'bb', 'bb1', 'bnb', 'bnb1', 'normal', 'lognormal', 'halfnormal', 'exponential', 'zip', 'zigeom', 'zinb', 'zibb', zibnb', 'zinormal', 'zilognorm', 'zihalfnorm', 'ziexp', 'ph', 'geomh','nbh','bbh','bnbh', 'normalh', 'lognormh', 'halfnormh', and 'exph' , which corresponds to Poisson, geometric, negative

	binomial, negative binomial1, beta binomial, beta binomial1, beta negative binomial, beta negative binomial1, normal, half normal, log normal, and exponential distributions and their zero-inflated as well as hurdle version, respectively. Defult is 'poisson'.
r	An initial value of the number of success before which m failures are observed, where m is the element of x. Must be a positive number, but not required to be an integer.
p	An initial value of the probability of success, should be a positive value within (0,1).
alpha1	An initial value for the first shape parameter of beta distribution. Should be a positive number.
alpha2	An initial value for the second shape parameter of beta distribution. Should be a positive number.
n	An initial value of the number of trials. Must be a positive number, but not required to be an integer.
lambda	An initial value of the rate. Must be a positive real number.
mean	An initial value of the mean or expectation.
sigma	An initial value of the standard deviation. Must be a positive real number.
lowerbound	A lower searching bound used in the optimization of likelihood function. Should be a small positive number. The default is 1e-2.
upperbound	An upper searching bound used in the optimization of likelihood function. Should be a large positive number. The default is 1e4.
parallel	whether to use multiple threads paralleling for computation. Default is FALSE. Please aware that it may take longer time to execute the program with parallel=FALSE.

Details

In arguments nsim, bootstrap, dist, if the length is larger than 1, the first element will be used. For other arguments except for x, the first valid value will be used if the input is not NULL, otherwise some naive sample estimates will be fed into the algorithm. Note that only the initial values that is used in the null distribution dist are needed. For example, with dist=poisson, user should provide a value for lambda and not the other parameters. With an output p-value less than some user-specified significance level, x is probably coming from a distribution other than the dist, given the current data. If p-values of more than one distributions are greater than the pre-specified significance level, user may consider a following likelihood ratio test to select a 'better' distribution. The methodology of computing Monte Carlo p-value is when bootstrap=TRUE, nsim bootstrapped samples will be generated by re-sampling x without replacement. Otherwise, nsim samples are simulated from the null distribution with the maximum likelihood estimate of original data x. Then compute the maximum likelihood estimates of nsim bootstrapped or simulated samples, based on which nsim new samples are generated under the null distribution. nsim KS statistics are calculated for the nsim new samples, then the Monte Carlo p-value is resulted from comparing the nsim KS statistics and the statistic of original data x. During the process of computing maximum likelihood estimates, the negative log likelihood function is minimized via basic R function optim with the searching interval decided by lowerbound and upperbound. Next simulate i.i.d. simulates from the estimated

parameters and calculate a new mle based on the bootstrapped samples. Then calculate the KS statistic and the p-value. For large sample sizes we may use kstest.A and for small sample sizes (less that 50 or 100), kstest.B is preferred.

Value

An object of class 'kstest.A' including the following elements:

- x: x used in computation.
- nsim: nsim used in computation.
- bootstrap: bootstrap used in computation.
- dist: dist used in computation.
- lowerbound: lowerbound used in computation.
- upperbound: upperboound used in computation.
- mle_new: A matrix of the maximum likelihood estimates of unknown parameters under the null distribution, using nsim bootstrapped or simulated samples.
- mle_ori: A row vector of the maximum likelihood estimates of unknown parameters under the null distribution, using the original data x.
- mle_c: A row vector of the maximum likelihood estimates of unknown parameters under the null distribution, using bootstrapped samples with parameters of mle_new.
- pvalue: Monte Carlo p-value of the one-sample KS test.
- N: length of x.
- r: initial value of r used in computation.
- p: initial value of p used in computation.
- alpha1: initial value of alpha1 used in computation.
- alpha2: initial value of alpha2 used in computation.
- lambda: initial value of lambda used in computation.
- n: initial value of n used in computation.
- mean: initial value of mean used in computation.
- sigma: initial value of sigma used in computation.

References

- H. Aldirawi, J. Yang, A. A. Metwally (2019). Identifying Appropriate Probabilistic Models for Sparse Discrete Omics Data, accepted for publication in 2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI).

See Also

kstest.A

Examples

```
set.seed(008)
x=sample.zi1(2000,phi=0.3,dist='bnb',r=5,alpha=3,alpha2=3)
kstest.B(x,nsim=100,bootstrap = TRUE,dist= 'zinb')$pvalue   #0.01
kstest.B(x,nsim=100,bootstrap = TRUE,dist= 'zibb')$pvalue   #0.02
kstest.B(x,nsim=100,bootstrap = TRUE,dist= 'zibnb')$pvalue  #0.67
x2=sample.h1(2000,phi=0.3,dist="halfnormal",sigma=4)
kstest.B(x2,nsim=100,bootstrap = TRUE,dist= 'halfnormh')$pvalue   #0.73
kstest.B(x2,nsim=100,bootstrap = TRUE,dist= 'lognormh')$pvalue    #0
```

---

lrt.A *likelihood ratio test for two models based on kstest.A*

Description

Conduct likelihood ratio test for comparing two different models.

Usage

```
lrt.A(d1, d2, parallel = FALSE)
```

Arguments

d1	An object of class 'kstest.A'.
d2	An object of class 'kstest.A'.
parallel	Whether to use multiple threads to paralleling computation. Default is FALSE. Please aware that it may take longer time to execute the program with parallel=FALSE.

Details

If the pvalue of d1 and d2 are greater than the user-specified significance level, which indicates that the original data x may come from the two distributions in d1 and d2, a likelihood ratio test is desired to choose a more 'possible' distribution based on the current data. NOTE that the x in d1 and d2 must be IDENTICAL! Besides, NOTE that the dist in d1 and d2 must be DIFFERENT! The dist inherited from d1 is the null distribution and that from d2 is used as the alternative distribution.

If the output p-value smaller than the user-specified significance level, the dist of d2 is more appropriate for modeling x. Otherwise, There is no significant difference between dist of d1 and dist of d2, given the current data.

Value

The p-value of the likelihood ratio test.

Examples

```
set.seed(1005)
x=sample.h1(2000,phi=0.3,dist='poisson',lambda=10)
d1=kstest.A(x,nsim=100,bootstrap = TRUE,dist= 'ph',lowerbound = 1e-10, upperbound = 100000)
d2=kstest.A(x,nsim=100,bootstrap = TRUE,dist= 'geomh',lowerbound = 1e-10, upperbound = 100000)
lrt.A(d1,d2, parallel = FALSE) #0.28
```

`new.mle` *Maximum likelihood estimate for general discrete and continuous distributions*

Description

calculate the Maximum likelihood estimate and the corresponding negative log likelihood value for general Poisson, geometric, negative binomial, negative binomial1, beta binomial, beta binomial1, beta negative binomial, beta negative binomial1, normal, half normal, log normal, and exponential distributions.

Usage

```
new.mle(x,r,p,alpha1,alpha2,n,lambda,mean,sigma,dist,lowerbound=0.01,upperbound=10000)
```

Arguments

`x`	A vector of count data which should non-negative integers for discrete cases. Real valued random generation for continuous cases.
`r`	An initial value for the number of success before which m failures are observed, where m is the element of x. Must be a positive number, but not required to be an integer.
`p`	An initial value for the probability of success, should be a positive value within (0,1).
`alpha1`	An initial value for the first shape parameter of beta distribution. Should be a positive number.
`alpha2`	An initial value for the second shape parameter of beta distribution. Should be a positive number.
`n`	An initial value for the number of trials. Must be a positive number, but not required to be an integer.
`lambda`	An initial value for the rate. Must be a positive real number.
`mean`	An initial value of the mean or expectation. A real number
`sigma`	An initial value of the standard deviation. Must be a positive real number.
`dist`	The distribution used to calculate the maximum likelihood estimate. Can be one of 'poisson', 'geometric', 'nb', 'nb1', 'bb', 'bb1', 'bnb', 'bnb1', 'normal', 'halfnormal', 'lognormal', 'exponential', which corresponds to Poisson, geometric, negative binomial, negative binomial1, beta binomial, beta binomial1, beta negative binomial, beta negative binomial1, normal, half normal, log normal, and exponential distributions.

lowerbound	A lower searching bound used in the optimization of likelihood function. Should be a small positive number. The default is 1e-2.
upperbound	An upper searching bound used in the optimization of likelihood function. Should be a large positive number. The default is 1e4.

Details

new.mle calculate Maximum likelihood estimate and corresponding negative log likelihood of general Poisson, geometric, negative binomial, negative binomial1, beta binomial, beta binomial1, beta negative binomial, beta negative binomial1, normal, half normal, log normal, and exponential distributions.

Value

A row vector containing the maximum likelihood estimate of the unknown parameters and the corresponding value of negative log likelihood.

If dist = poisson, the following values are returned:

- lambda: the maximum likelihood estimate of λ.
- loglik: the value of negative log likelihood with maximum likelihood estimate plugged-in.

If dist = geometric, the following values are returned:

- p: the maximum likelihood estimate of p.
- loglik: the value of negative log likelihood with maximum likelihood estimate plugged-in.

If dist = nb, the following values are returned:

- r: the maximum likelihood estimate of r.
- p: the maximum likelihood estimate of p.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = nb1, the following values are returned:

- r: the maximum likelihood estimate of rounded r (returns integer estimate).
- p: the maximum likelihood estimate of p.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = bb, the following values are returned:

- n: the maximum likelihood estimate of n.
- alpha1: the maximum likelihood estimate of α_1.
- alpha2: the maximum likelihood estimate of α_2.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = bb1, the following values are returned:

- n: the maximum likelihood estimate of rounded n (returns integer estimate).
- alpha1: the maximum likelihood estimate of α_1.

- alpha2: the maximum likelihood estimate of α_2.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = bnb, the following values are returned:

- r: the maximum likelihood estimate of r.
- a lpha1: the maximum likelihood estimate of α_1.
- alpha2: the maximum likelihood estimate of α_2.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = bnb1, the following values are returned:

- r: the maximum likelihood estimate of rounded r (returns integer estimate).
- alpha1: the maximum likelihood estimate of α_1.
- alpha2: the maximum likelihood estimate of α_2.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = normal, the following values are returned:

- mean: the maximum likelihood estimate of μ.
- sigma: the maximum likelihood estimate of σ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = lognormal, the following values are returned:

- mean: the maximum likelihood estimate of μ.
- sigma: the maximum likelihood estimate of σ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = halfnormal, the following values are returned:

- sigma: the maximum likelihood estimate of σ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = exponential, the following values are returned:

- lambda: the maximum likelihood estimate of λ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

References

- H. Aldirawi, J. Yang, A. A. Metwally (2019). Identifying Appropriate Probabilistic Models for Sparse Discrete Omics Data, accepted for publication in 2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI).

Examples

```
set.seed(001)
x1=stats::rpois(1000,lambda=10)
new.mle(x1,lambda=3,dist="poisson")                          #9.776 -2611.242
x2=stats::rgeom(1000,prob=0.2)
new.mle(x2,p=0.5,dist="geometric")                           #0.1963865 -2522.333
x3=stats::rnbinom(1000,size=5,prob=0.3)
new.mle(x3,r=2,p=0.6,dist="nb")                             #5.113298 0.3004412 -3186.163
new.mle(x3,r=2,p=0.6,dist="nb1")                             #5 0.299904 -3202.223
x4=extraDistr::rbbinom(1000,size=4,alpha=2,beta=3)
new.mle(x4,n=10,alpha1=3,alpha2=4,dist="bb")              #3.99 1.78774 2.680009 -1533.982
new.mle(x4,n=10,alpha1=3,alpha2=4,dist="bb1")               #4 1.800849 2.711264 -1534.314
x5=extraDistr::rbnbinom(1000, size=5, alpha=3,beta=3)
new.mle(x5,r=5,alpha1=3,alpha2=4,dist="bnb")           #5.472647 3.008349 2.692704 -3014.372
new.mle(x5,r=5,alpha1=3,alpha2=4,dist="bnb1")              #5 2.962727 2.884826 -3014.379
x6=stats::rnorm(1000,mean=10,sd=2)
new.mle(x6,mean=2,sigma=1,dist="normal")                     #9.976704 2.068796 -2145.906
x7=stats::rlnorm(1000, meanlog = 1, sdlog = 4)
new.mle(x7,mean=2,sigma=2,dist="lognormal")                  #0.9681913 3.299503 -3076.156
x8=extraDistr::rhnorm(1000, sigma = 3)
new.mle(x8,sigma=2,dist="halfnormal")                        #3.103392 -1858.287
x9=stats::rexp(1000,rate=1.5)
new.mle(x9,lambda=3,dist="exponential")                      #1.454471 -625.3576
```

`ofp` *Number of physician office visits.*

Description

A data set containing ofp (number of physician office visit) of 4406 individuals.

Usage

```
ofp
```

Format

The original data set is based on the work of Deb and Trivedi (1997) analyze data on 4406 individuals, aged 66 and over, who are covered by Medicare, a public insurance program. Originally obtained from the US National Medical Expenditure Survey (NMES) for 1987/88, the data are available from the data archive of the Journal of Applied Econometrics at http://www.econ.queensu.ca/jae/1997-v12.3/deb-trivedi/. In AZIAD package we work with the number of physicians office visits for the patients.Based on the analysis of kstest.A and kstest.B and lrt.A the data belongs to zero-inflated beta negative binomial or beta negative binomial hurdle model.

Source

- http://www.jstatsoft.org/v27/i08/paper

References

- Zeileis, A. and Kleiber, C. and Jackma, S. (2008). "Regression Models for Count Data in R". JSS 27, 8, 1–25.

Examples

```
ofp
set.seed(1008)
d1=kstest.A(ofp,nsim=200,bootstrap=TRUE,dist="geometric")
d2=kstest.A(ofp,nsim=200,bootstrap=TRUE,dist="zibnb")
lrt.A(d1,d2)       #0
```

`sample.h1` *Generate random deviates from hurdle models*

Description

Generate random deviates from hurdle Poisson, geometric, negative binomial, beta binomial, beta negative binomial, normal, log normal, half normal, and exponential models.

Usage

```
sample.h1(N,phi,dist="poisson",lambda=NA,r=NA,p=NA,
alpha1=NA,alpha2=NA,n=NA,mean=NA,sigma=NA)
```

Arguments

N	The sample size. Should be a positive number. If it is not an integer, N will be automatically rounded up to the smallest integer that no less than N.
phi	The structural parameter ϕ, should be a positive value within (0,1).
dist	The corresponding standard distribution. Can be one of 'poisson', 'geometric','nb','bb', 'bnb','normal', 'lognormal', 'halfnormal','exponential', which corresponds to Poisson, geometric, negative binomial, beta binomial, beta negative binomial, normal, log normal, hal fnormal, and exponential distributions respectively.
lambda	A value for the parameter of Poisson distribution. Should be a positive number.
r	the number of success before which m failures are observed, where m is a random variable from negative binomial or beta negative binomial distribution. Must be a positive number. If it is not an integer, r will be automatically rounded up to the smallest integer that no less than r.
p	The probability of success, should be a positive value within (0,1).
alpha1	The first shape parameter of beta distribution. Should be a positive number.
alpha2	The second shape parameter of beta distribution. Should be a positive number.
n	The number of trials. Must be a positive number. If it is not an integer, n will be automatically rounded up to the smallest integer that no less than n.
mean	A value for parameter of the mean or expectation.
sigma	A value of parameter for standard deviation. Must be a positive real number.

Details

- Setting dist=poisson and lambda, sample.h1 simulates N random deviates from hurdle Poisson distribution, respectively, and so on forth.
- Setting the dist=geometric and the argument p is for the use of hurdle geometric distributions.
- ASetting the dist=nb and the arguments r and p are for the use of and hurdle negative binomial distributions.
- Setting the dist=bb and the arguments n, alpha1, and alpha2 are for and hurdle beta binomial distributions.
- Setting the dist=bnb and the arguments r, alpha1, and alpha2 are used in hurdle beta negative binomial distributions.
- Setting the dist=normal and the arguments mean and sigma are used in and hurdle normal distributions.
- Setting the dist=lognormal and the arguments mean and sigma are used in and hurdle log normal distributions.
- Setting the dist=halfnormal and the argument sigma is used in and hurdle half normal distributions.
- Setting the dist=exponential and the argument lambda is used in and hurdle exponential distributions.

Random deviates from standard Poisson, geometric, negative binomial, normal, log normal, and exponential distributions can be generated by basic R function rpois, rgeom, rnbinom, rnorm, rlnorm, and rexp in R package stats.

Functions rbbinom and rbnbinom, and rhnorm are available for standard beta binomial, beta negative binomial, and half normal distributions in R package extraDistr.

Value

A vector of length N containing non-negative integers from the hurdle version of distribution determined by dist.

References

- H. Aldirawi, J. Yang, A. A. Metwally, Identifying Appropriate Probabilistic Models for Sparse Discrete Omics Data, accepted for publication in 2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI) (2019).
- T. Wolodzko, extraDistr: Additional Univariate and Multivariate Distributions, R package version 1.8.11 (2019), https://CRAN.R-project.org/package=extraDistr.

Examples

```
x6=sample.h1(2000,phi=0.3,dist='nb',r=10,p=0.3)              #hurdle negative binomial
x7=sample.h1(2000,phi=0.3,dist='bnb',r=5,alpha=3,alpha2=3)  #hurdle beta negative binomial
x8=sample.h1(2000,phi=0.3,dist="halfnormal",sigma=4)         #hurdle half normal
x9=sample.h1(2000,phi=0.3,dist="lognormal",mean=1,sigma=4)   #hurdle log normal
```

sample.zi1 *Generate random deviates from zero-inflated*

Description

Generate random deviates from zero-inflated Poisson, geometric, negative binomial, beta binomial, beta negative binomial, normal, log normal, half normal, and exponential models.

Usage

```
sample.zi1(N,phi,dist="poisson",lambda=NA,r=NA,p=NA,
alpha1=NA,alpha2=NA,n=NA,mean=NA,sigma=NA)
```

Arguments

N	The sample size. Should be a positive number. If it is not an integer, N will be automatically rounded up to the smallest integer that no less than N.
phi	The structural parameter ϕ, should be a positive value within (0,1).
dist	The corresponding standard distribution. Can be one of 'poisson', 'geometric','nb','bb', 'bnb','normal', 'lognormal', 'halfnormal','exponential', which corresponds to Poisson, geometric, negative binomial, beta binomial, beta negative binomial, normal, log normal, hal fnormal, and exponential distributions respectively.
lambda	A value for the parameter of Poisson distribution. Should be a positive number.
r	the number of success before which m failures are observed, where m is a random variable from negative binomial or beta negative binomial distribution. Must be a positive number. If it is not an integer, r will be automatically rounded up to the smallest integer that no less than r.
p	The probability of success, should be a positive value within (0,1).
alpha1	The first shape parameter of beta distribution. Should be a positive number.
alpha2	The second shape parameter of beta distribution. Should be a positive number.
n	The number of trials. Must be a positive number. If it is not an integer, n will be automatically rounded up to the smallest integer that no less than n.
mean	A value for parameter of the mean or expectation.
sigma	A value of parameter for standard deviation. Must be a positive real number.

Details

- Setting dist=poisson and lambda, sample.zi1 simulates N random deviates from zero-inflated Poisson distribution, respectively, and so on forth.
- Setting the dist=geometric and the argument p is for the use of zero-inflated geometric distributions.
- ASetting the dist=nb and the arguments r and p are for the use of zero-inflated negative binomial distributions.

- Setting the dist=bb and the arguments n, alpha1, and alpha2 are for zero-inflated beta binomial distributions.
- Setting the dist=bnb and the arguments r, alpha1, and alpha2 are used in zero-inflated beta negative binomial distributions.
- Setting the dist=normal and the arguments mean and sigma are used in zero-inflated normal distributions.
- Setting the dist=lognormal and the arguments mean and sigma are used in zero-inflated log normal distributions.
- Setting the dist=halfnormal and the argument sigma is used in zero-inflated half normal distributions.
- Setting the dist=exponential and the argument lambda is used in zero-inflated exponential distributions.

Random deviates from standard Poisson, geometric, negative binomial, normal, log normal, and exponential distributions can be generated by basic R function rpois, rgeom, rnbinom, rnorm, rlnorm, and rexp in R package stats.

Functions rbbinom and rbnbinom, and rhnorm are available for standard beta binomial, beta negative binomial, and half normal distributions in R package extraDistr.

Value

A vector of length N containing non-negative integers from the zero-inflated version of distribution determined by dist.

References

- H. Aldirawi, J. Yang, A. A. Metwally, Identifying Appropriate Probabilistic Models for Sparse Discrete Omics Data, accepted for publication in 2019 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI) (2019).
- T. Wolodzko, extraDistr: Additional Univariate and Multivariate Distributions, R package version 1.8.11 (2019), https://CRAN.R-project.org/package=extraDistr.

Examples

```
x1=sample.zi1(2000,phi=0.3,dist='poisson',lambda=10)         #zero-inflated Poisson
x2=sample.zi1(2000,phi=0.2,dist='geometric',p=0.2)           #zero-inflated geometric
x3=sample.zi1(2000,phi=0.3,dist='bb',n=10,alpha1=2,alpha2=4) #zero-inflated beta binomial
x4=sample.zi1(2000,phi=0.3,dist="normal",mean=10,sigma=2)    #zero-inflated normal
x5=sample.zi1(2000,phi=0.3,dist="exponential",lambda=20)     #zero-inflated exponential
```

zih.mle *Maximum likelihood estimate for Zero-Inflated or Zero-Altered discrete and continuous distributions.*

Description

Calculate the Maximum likelihood estimate and the corresponding negative log likelihood value for Zero-Inflated or Zero-Altered Poisson, geometric, negative binomial, negative binomial1, beta binomial, beta binomial1, beta negative binomial, beta negative binomial1, normal, half normal, log normal, and exponential distributions.

Usage

```
zih.mle(x,r,p,alpha1,alpha2,n,lambda,mean,sigma,
type=c("zi","h"),dist,lowerbound=0.01,upperbound = 10000 )
```

Arguments

x	A vector of count data which should non-negative integers for discrete cases. Real-valued random generation for continuous cases.
r	An initial value of the number of success before which m failures are observed, where m is the element of x. Must be a positive number, but not required to be an integer.
p	An initial value of the probability of success, should be a positive value within (0,1).
alpha1	An initial value for the first shape parameter of beta distribution. Should be a positive number.
alpha2	An initial value for the second shape parameter of beta distribution. Should be a positive number.
n	An initial value of the number of trials. Must be a positive number, but not required to be an integer.
lambda	An initial value of the rate. Must be a postive real number.
mean	An initial value of the mean or expectation.
sigma	An initial value of the standard deviation. Must be a positive real number.
type	the type of distribution used to calculate the sample estimate, where 'zi' stand for zero-inflated and 'h' stands for hurdle distributions.
dist	The distribution used to calculate the maximum likelihood estimate. Can be one of 'poisson.zihmle', 'geometric.zihmle', 'nb.zihmle', 'nb1.zihmle', 'bb.zihmle', 'bb1.zihmle', 'bnb.zihmle', 'bnb1.zihmle', 'normal.zihmle', 'halfnorm.zihmle', 'lognorm.zimle', 'exp.zihmle' which corresponds to Zero-Inflated or Zero-Hurdle Poisson, geometric, negative binomial, negative binomial1, beta binomial, beta binomial1, beta negative binomial, beta negative binomial1, normal, log normal, half normal, and exponential distributions.

lowerbound	A lower searching bound used in the optimization of likelihood function. Should be a small positive number. The default is 1e-2.
upperbound	An upper searching bound used in the optimization of likelihood function. Should be a large positive number. The default is 1e4.

Details

zih.mle calculate the Maximum likelihood estimate and the corresponding negative log likelihood of Zero-Inflated or Zero-Hurdle Poisson, geometric, negative binomial, negative binomial1, beta binomial, beta binomial1, beta negative binomial, beta negative binomial1, normal, log normal, half normal, and exponential distributions.

If dist = poisson.zihmle, the following values are returned:

- lambda: the maximum likelihood estimate of λ.
- phi: the maximum likelihood estimate of ϕ.
- loglik: the value of negative log likelihood with maximum likelihood estimate plugged-in.

If dist = geometric.zihmle, the following values are returned:

- p: the maximum likelihood estimate of p.
- phi: the maximum likelihood estimate of ϕ.
- loglik: the value of negative log likelihood with maximum likelihood estimate plugged-in.

If dist = nb.zihmle, the following values are returned:

- r: the maximum likelihood estimate of r.
- p: the maximum likelihood estimate of p.
- phi: the maximum likelihood estimate of ϕ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = nb1.zihmle, the following values are returned:

- r: the maximum likelihood estimate of rounded r (returns integer estimate).
- p: the maximum likelihood estimate of p.
- phi: the maximum likelihood estimate of ϕ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = bb.zihmle, the following values are returned:

- n: the maximum likelihood estimate of n.
- alpha1: the maximum likelihood estimate of α_1.
- alpha2: the maximum likelihood estimate of α_2.
- phi: the maximum likelihood estimate of ϕ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = bb1.zihmle, the following values are returned:

- n: the maximum likelihood estimate of rounded n (returns integer estimate).

- alpha1: the maximum likelihood estimate of α_1.
- alpha2: the maximum likelihood estimate of α_2.
- phi: the maximum likelihood estimate of ϕ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = bnb.zihmle, the following values are returned:

- r: the maximum likelihood estimate of r.
- alpha1: the maximum likelihood estimate of α_1.
- alpha2: the maximum likelihood estimate of α_2.
- phi: the maximum likelihood estimate of ϕ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = bnb1.zihmle, the following values are returned:

- r: the maximum likelihood estimate of rounded r (returns integer estimate).
- alpha1: the maximum likelihood estimate of α_1.
- alpha2: the maximum likelihood estimate of α_2.
- phi: the maximum likelihood estimate of ϕ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = normal.zihmle, the following values are returned:

- mean: the maximum likelihood estimate of μ.
- sigma: the maximum likelihood estimate of σ.
- phi: the maximum likelihood estimate of ϕ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = lognorm.zihmle, the following values are returned:

- mean: the maximum likelihood estimate of μ.
- sigma: the maximum likelihood estimate of σ.
- phi: the maximum likelihood estimate of ϕ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = halfnorm.zihmle, the following values are returned:

- sigma: the maximum likelihood estimate of σ.
- phi: the maximum likelihood estimate of ϕ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

If dist = exp.zihmle, the following values are returned:

- lambda: the maximum likelihood estimate of λ.
- phi: the maximum likelihood estimate of ϕ.
- loglik: the value of negative log likelihood with maximum likelihood estimates plugged-in.

Value

A row vector containing the maximum likelihood estimate of the unknown parameters and the corresponding value of negative log likelihood.

References

- H. Aldirawi, J. Yang (2019). Model Selection and Regression Analysis for Zero-altered or Zero-inflated Data, Statistical Laboratory Technical Report, no.2019-01, University of Illinois at Chicago.

Examples

```
set.seed(007)
x1=sample.zi1(2000,phi=0.3,dist='poisson',lambda=2)
zih.mle(x1,lambda=10,dist="poisson.zihmle",type="zi")
#2.00341 0.3099267 -3164.528
x2=sample.zi1(2000,phi=0.3,dist='geometric',p=0.2)
zih.mle(x2,p=0.3,dist="geometric.zihmle",type="zi")
#0.1976744 0.2795942 -4269.259
x3=sample.zi1(2000,phi=0.3,dist='nb',r=10,p=0.3)
zih.mle(x3,r=2,p=0.2,dist="nb.zihmle",type="zi")
#10.18374 0.3033975 0.2919962 -6243.002
zih.mle(x3,r=2,p=0.2,dist="nb1.zihmle",type="zi")
#10 0.2995633 0.2919959 -6243.059
x4=sample.zi1(2000,phi=0.3,dist='bb',n=10,alpha1=2,alpha2=4)
zih.mle(x4,n=10,alpha1=3,alpha2=4,dist="bb.zihmle",type="zi")
#9.99 1.862798 3.756632 0.2643813 -3982.646
zih.mle(x4,n=10,alpha1=3,alpha2=4,dist="bb1.zihmle",type="zi")
#10 1.866493 3.76888 0.2644992 -3982.682
x5=sample.zi1(2000,phi=0.3,dist='bnb',r=5,alpha=3,alpha2=3)
zih.mle(x5,r=10,alpha1=3,alpha2=4,dist="bnb.zihmle",type="zi")
#6.936502 3.346791 2.32905 0.285682 -5088.173
zih.mle(x5,r=10,alpha1=3,alpha2=4,dist="bnb1.zihmle",type="zi")
#7 3.353377 2.313633 0.2855203 -5088.173
x6=sample.zi1(2000,phi=0.3,dist="normal",mean=10,sigma=2)
zih.mle(x6,mean=2,sigma=2,dist="normal.zihmle",type="zi")
#9.988447 2.015987 0.28 -4242.18
x7=sample.zi1(2000,phi=0.3,dist="lognormal",mean=1,sigma=4)
zih.mle(x7,mean=4,sigma=2,dist="lognorm.zihmle",type="zi")
#1.003887 3.945388 0.2985 -6544.087
x8=sample.zi1(2000,phi=0.3,dist="halfnormal",sigma=4)
zih.mle(x8,sigma=1,dist="halfnorm.zihmle",type="zi")
#1.292081 0.294 -8573.562
x9=sample.zi1(2000,phi=0.3,dist="exponential",lambda=20)
zih.mle(x9,lambda=10,dist="exp.zihmle",type="zi")
#20.1165 0.294 1614.786

set.seed(008)
y1=sample.h1(2000,phi=0.3,dist='poisson',lambda=10)
zih.mle(y1,lambda=10,dist="poisson.zihmle",type="h")
#10.11842 0.3015 -4826.566
y2=sample.h1(2000,phi=0.3,dist='geometric',p=0.3)
```

```
zih.mle(y2,p=0.2,dist="geometric.zihmle",type="h")
#0.3050884 0.2925 -4061.65
y3=sample.h1(2000,phi=0.3,dist='nb',r=10,p=0.3)
zih.mle(y3,r=2,p=0.2,dist="nb.zihmle",type="h")
#9.50756 0.2862545 0.297 -6261.479
zih.mle(y3,r=2,p=0.2,dist="nb1.zihmle",type="h")
#10 0.2966819 0.297 -6261.932
y4=sample.h1(2000,phi=0.3,dist='bb',n=10,alpha1=2,alpha2=4)
zih.mle(y4,n=10,alpha1=3,alpha2=4,dist="bb.zihmle",type="h")
#9.99 1.894627 3.851142 0.293 -4092.983
zih.mle(y4,n=10,alpha1=3,alpha2=4,dist="bb1.zihmle",type="h")
#10 1.898415 3.863768 0.293 -4093.004
y5=sample.h1(2000,phi=0.3,dist='bnb',r=5,alpha=3,alpha2=3)
zih.mle(y5,r=10,alpha1=3,alpha2=4,dist="bnb.zihmle",type="h")
#3.875685 3.026982 3.874642 0.328 -5274.091
zih.mle(y5,r=10,alpha1=3,alpha2=4,dist="bnb1.zihmle",type="h")
#4 3.028185 3.756225 0.328 -5274.092
y6=sample.h1(2000,phi=0.3,dist="normal",mean=10,sigma=2)
zih.mle(y6,mean=2,sigma=2,dist="normal.zihmle",type="h")
#10.01252 1.996997 0.29 -4201.334
y7=sample.h1(2000,phi=0.3,dist="lognormal",mean=1,sigma=4)
zih.mle(y7,mean=4,sigma=2,dist="lognorm.zihmle",type="h")
#0.9305549 3.891624 0.287 -6486.92
y8=sample.h1(2000,phi=0.3,dist="halfnormal",sigma=4)
zih.mle(y8,sigma=1,dist="halfnorm.zihmle",type="h")
#1.26807 0.3 -8863.063
y9=sample.h1(2000,phi=0.3,dist="exponential",lambda=20)
zih.mle(y9,lambda=10,dist="exp.zihmle",type="h")
#20.26938 0.2905 1645.731
```

Index