

# Package ‘ZIGP’

February 11, 2010

**Type** Package

**Title** Zero Inflated Generalized Poisson (ZIGP) regression models

**Version** 3.8

**Date** 2010-02-11

**Author** Vinzenz Erhardt

**Maintainer** Vinzenz Erhardt <erhardt@ma.tum.de>

**Description** Fit, analyze and compare GP, ZIP and ZIGP regression models

**License** GPL (>= 3)

**Repository** CRAN

**Date/Publication** 2010-02-11 18:33:18

## R topics documented:

ZIGP-package . . . . .	2
clarke . . . . .	3
dzipg . . . . .	4
eda.od . . . . .	5
eda.zi . . . . .	6
est.zigp . . . . .	7
fit.zigp . . . . .	9
fit.zigpl . . . . .	10
FM . . . . .	11
gradient . . . . .	12
gradientl . . . . .	13
loglikelihood.zigp . . . . .	13
loglikelihood.zigp.full . . . . .	14
loglikelihood.zigp.vuong . . . . .	14
mle.zigp . . . . .	14
mle.zigp.full.like . . . . .	16

optimized.run . . . . .	16
pzip . . . . .	16
qzip . . . . .	17
response.zip . . . . .	17
rzip . . . . .	18
summaryzip1 . . . . .	19
vuong . . . . .	19

<b>Index</b>	<b>21</b>
--------------	-----------

---

ZIGP-package	<i>Zero-Inflated Generalized Poisson (ZIGP) Models</i>
--------------	--

---

## Description

Fits GP, ZIP and ZIGP models by Maximum Likelihood estimation. Regression is allowed not only on the mean but also on overdispersion and zero-inflation level.

## Details

Package: ZIGP  
 Type: Package  
 Version: 3.8  
 Date: 2010-02-11  
 License: GPL (>= 3)

Distribution functions in R notation are 'dzip', 'pzip', 'qzip', 'rzip'.

The main function is 'est.zip'. This function can be fed with design formulas for the mean, overdispersion and zero-inflation level. It returns a summary-like overview of the covariates together with significance statistics referring to the Wald test. It can be used for sequential elimination of non-significant effects. Function 'mle.zip' returns estimates of regression coefficients, AIC etc. for further use.

Other useful functions are 'loglikelihood.zip', which evaluates the loglikelihood function on the given parameter value. Scores can be calculated using 'gradient', the Fisher Information matrix using 'FM'.

Tools for an exploratory data analysis for the overdispersion level and zero-inflation level are given in 'eda.od' and 'eda.zi', respectively.

Nonnested model comparison (also for the Negative Binomial distribution) can be facilitated using a test proposed by Vuong which is implemented in function 'vuong' or by a test proposed by Clarke using function 'clarke'.

## Author(s)

Vinzenz Erhardt

Maintainer: Vinzenz Erhardt <erhardt@ma.tum.de>

## References

Czado, C., Erhardt, V., Min, A., Wagner, S. (2007) Zero-inflated generalized Poisson models with regression effects on the mean, dispersion and zero-inflation level applied to patent outsourcing rates. *Statistical Modelling* 7 (2), 125-153.

Masterthesis in German: Erhardt, Vinzenz. Verallgemeinerte Poisson und Nullenueberschuss- Regressionsmodelle mit regressiertem Erwartungswert, Dispersions- und Nullenüberschuß-Parameter und eine Anwendung zur Patentmodellierung. ("<http://www-m4.ma.tum.de/Diplarb/>"), 2006.

Vuong, Q.H. (1989). Likelihood Ratio tests for model selection and nonnested hypotheses. *Econometrica* 57(2), 307-333.

Clarke, Kevin A. (2007). A Simple Distribution-Free Test for Nonnested Model Selection. *Political Analysis* 2007 15(3), 347-363.

Schwarz, G. (1978). Estimating the Dimension of a Model. *Annals of Statistics* 6, 461-464.

---

 clarke

---

*Clarke's test for non-nested model comparison*


---

## Description

'clarke' suggests the better of two (not necessarily nested) models.

## Usage

```
clarke(modell, model2, alpha=0.05, correction=T)
```

## Arguments

modell, model2	the output of two model fits obtained by using 'mle.zigp'.
alpha	significance level, defaults to 0.05.
correction	boolean, if TRUE (default), the Schwarz correction will be used on the differences of log-likelihoods.

## References

Clarke, Kevin A. (2007). A Simple Distribution-Free Test for Nonnested Model Selection. *Political Analysis* 2007 15(3), 347-363.

Schwarz, G. (1978). Estimating the Dimension of a Model. *Annals of Statistics* 6, 461-464.

**Examples**

```

data(Seatbelts)
DriversKilled <- as.vector(Seatbelts[,1]) # will be response
kms <- as.vector(Seatbelts[,5]/mean(Seatbelts[,5])) # will be exposure
PetrolPrice <- as.vector(Seatbelts[,6]) # will be covariate 1
law <- as.vector(Seatbelts[,8]) # will be covariate 2

fm.X.poi <- ~ PetrolPrice + law

fm.X.gp <- ~ PetrolPrice + law
fm.W.gp <- ~ 1

fm.X.zigp <- ~ PetrolPrice + law
fm.W.zigp <- ~ 1
fm.Z.zigp <- ~ 1

poi <- mle.zigp(Yin=DriversKilled, fm.X=fm.X.poi, fm.W=NULL,
               fm.Z=NULL, Offset = kms, init = FALSE)
gp <- mle.zigp(Yin=DriversKilled, fm.X=fm.X.gp, fm.W=fm.W.gp,
               fm.Z=NULL, Offset = kms, init = FALSE)
zigp <- mle.zigp(Yin=DriversKilled, fm.X=fm.X.zigp, fm.W=fm.W.zigp,
                 fm.Z=fm.Z.zigp, Offset = kms, init = FALSE)
# it is possible to compare to a Negative Binomial fit:
library(MASS)
nb <- glm.nb(DriversKilled ~ offset(log(kms)) + PetrolPrice + law)

clarke(poi,gp)
clarke(gp,zigp)
clarke(poi,zigp)
clarke(gp,nb)

```

---

dzigp

*PMF of ZIGP distribution*


---

**Description**

'dzigp' calculates the probability mass function of the ZIGP distribution.

**Usage**

```
dzigp(x, mu, phi, omega)
```

**Arguments**

x	vector of discrete points
mu	mean
phi	dispersion parameter
omega	zero inflation parameter

**Value**

Calculates a vector of the same length as of x of pmf-values.

**Examples**

```
x <- 0:10
dzigp(x, 2, 1.5, 0.2)
```

---

eda.od

*Exploratory data analysis tool for overdispersion level*

---

**Description**

'eda.od' performs an impact study on the influence of a covariate on the overdispersion design (where the shifted log-link is assumed). Thereby, a discretation using scoring classes will be applied and the overdispersion function be calculated for each scoring class (see Czado et. al (2007)).

**Usage**

```
eda.od(x, y, Offset=rep(1,length(y)), numberclasses=5)
```

**Arguments**

x	Covariate considered
y	Response considered
Offset	Exposure for individual observation lengths. Defaults to a vector of 1. The offset MUST NOT be in 'log' scale.
numberclasses	Number of classes for discretization. Defaults to 5.

**Details**

As covariate x, discrete or continuous variables may be considered. Categorical covariates only make sense if they have only two levels.

**References**

Czado, C., Erhardt, V., Min, A., Wagner, S. (2007) Zero-inflated generalized Poisson models with regression effects on the mean, dispersion and zero-inflation level applied to patent outsourcing rates. *Statistical Modelling* 7 (2), 125-153.

## Examples

```

data(Seatbelts)
DriversKilled <- as.vector(Seatbelts[,1])           # will be response
kms <- as.vector(Seatbelts[,5]/mean(Seatbelts[,5])) # will be exposure
PetrolPrice <- as.vector(Seatbelts[,6])           # will be covariate 1
law <- as.vector(Seatbelts[,8])                   # will be covariate 2

eda.od(x=PetrolPrice, y=DriversKilled, Offset=kms)
eda.od(x=PetrolPrice, y=DriversKilled, Offset=kms, numberclasses=20)
eda.od(x=law, y=DriversKilled, Offset=kms)

```

---

eda.zi

*Exploratory data analysis tool for zero-inflation level*

---

## Description

'eda.zi' performs an exploratory data analysis on the influence of a covariate on the zero-inflation design (where the logit-link is assumed). Thereby, a discretization using scoring classes will be applied and empirical logits be calculated for each scoring class (see Czado et. al (2007)). Here, a shift of 1/2 is used to obtain well defined empirical logits even for 0. The dashed line is the empirical logit of 1/(number of scoring classes). Empirical logits further away from this line indicate high influence on zero-inflation.

## Usage

```
eda.zi(x, y, numberclasses=5)
```

## Arguments

x	Covariate considered
y	Response considered
numberclasses	Number of classes for discretization. Defaults to 5.

## Details

As covariate x, discrete or continuous variables may be considered. Categorical covariates with two levels are allowed as well.

Notwithstanding the description in Czado et. al (2007), the empirical logits are now adjusted for individual class sizes.

## References

Czado, C., Erhardt, V., Min, A., Wagner, S. (2007) Zero-inflated generalized Poisson models with regression effects on the mean, dispersion and zero-inflation level applied to patent outsourcing rates. *Statistical Modelling* 7 (2), 125-153.

**Examples**

```

data(Seatbelts)
DriversKilled <- as.vector(Seatbelts[,1])           # will be response
kms <- as.vector(Seatbelts[,5]/mean(Seatbelts[,5])) # will be exposure
PetrolPrice <- as.vector(Seatbelts[,6])           # will be covariate 1
law <- as.vector(Seatbelts[,8])                   # will be covariate 2

# artificially create some zeros
DriversKilled[PetrolPrice<0.09] <- 0

eda.zi(x=PetrolPrice, y=DriversKilled)
eda.zi(x=PetrolPrice, y=DriversKilled, numberclasses=200)
eda.zi(x=law, y=DriversKilled)

```

---

est.zigp

*Fitting ZIGP( $\mu(i)$ ,  $\phi(i)$ ,  $\omega(i)$ ) - Regression Models*


---

**Description**

'est.zigp' is used to fit ZIGP( $\mu(i)$ ,  $\phi(i)$ ,  $\omega(i)$ ) - Regression Models (GP and ZIP as well).

**Usage**

```

est.zigp(Yin, fm.X, fm.W=NULL, fm.Z=NULL,
         Offset = rep(1, length(Yin)), init = T, tex=F, reltol = sqrt(.Machine$double.eps))

```

**Arguments**

Yin	Response vector of length n.
fm.X	Formula for mean design.
fm.W	Formula for overdispersion design (optional).
fm.Z	Formula for zero inflation design (optional).
Offset	Exposure for individual observation lengths. Defaults to a vector of 1. The offset MUST NOT be in 'log' scale.
init	A logical value indicating whether initial optimization values for dispersion are set to -2.5 and values for zero inflation regression parameters are set to -1 (init = F) or are estimated by a ZIGP( $\mu(i)$ , $\phi$ , $\omega$ )-model (init = T). Defaults to 'T'.
tex	Should the output be a TeX table? Defaults to regular output.
reltol	Relative tolerance for 'optim' routine.

## Details

Constant overdispersion and/or zero-inflation can be modelled using an Intercept design on the corresponding level. Setting fm.W to NULL corresponds to modelling a ZIP model. Setting fm.Z to NULL corresponds to modelling a GP model. Setting fm.W and fm.Z to NULL corresponds to modelling a Poisson GLM.

For numerical stability it may be very useful to center and standardize all non-categorical covariates, i.e. use `'x <- (x-mean(x))/sd(x)'`.

## References

Czado, C., Erhardt, V., Min, A., Wagner, S. (2007) Zero-inflated generalized Poisson models with regression effects on the mean, dispersion and zero-inflation level applied to patent outsourcing rates. *Statistical Modelling* 7 (2), 125-153.

## See Also

Estimated regression coefficients etc. can be obtained using `'mle.zigp'`.

Exploratory data analysis tools for overdispersion and zero-inflation designs are `'eda.od'` and `'eda.zi'`.

## Examples

```
# Number of damages in car insurance.
# (not a good fit, just to illustrate how the software is used)

damage <- c(0,1,0,0,0,4,2,0,1,0,1,1,0,2,0,0,1,0,0,1,0,0,0)
insurance.year <- c(1,1.2,0.8,1,2,1,1.1,1,1,1.1,1.2,1.3,0.9,1.4,1,1,1,
1.2,1,1,1,1,1)
drivers.age <- c(25,19,30,48,30,18,19,29,24,54,56,20,38,18,23,58,
47,36,25,28,38,39,42)
# for overdispersion: car brand dummy in {1,2,3}, brand = 1 is reference
brand <- c(1,2,1,3,3,2,2,1,1,3,2,2,1,3,1,3,2,2,1,1,3,3,2)
# abroad: driver has been abroad for longer time (=1)
abroad <- c(0,0,0,1,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1,0,1,1,1)
Y <- damage
fm.X <- ~ drivers.age
fm.W <- ~ 0 + factor(brand)
fm.Z <- ~ abroad

est.zigp(Yin=Y, fm.X=fm.X, fm.W=fm.W, fm.Z=fm.Z, Offset = insurance.year,
init = FALSE)

# approximate equivalence of Poisson-glm and ZIGP-package results
# glm uses IWLS, ZIGP uses numerical maximization of the log-likelihood
# (time series character of the data is neglected)

data(Seatbelts)
DriversKilled <- as.vector(Seatbelts[,1]) # will be response
kms <- as.vector(Seatbelts[,5]/mean(Seatbelts[,5])) # will be exposure
```

```

PetrolPrice <- as.vector(Seatbelts[,6])           # will be covariate 1
law <- as.vector(Seatbelts[,8])                 # will be covariate 2

fm.X <- DriversKilled ~ PetrolPrice + law
out.glm <- glm(fm.X, family=poisson, offset=log(kms))
summary(out.glm)

fm.X <- ~ PetrolPrice + law
est.zigp(DriversKilled, fm.X = fm.X, NULL, NULL, Offset = kms)

# GP with constant overdispersion
fm.X <- ~ PetrolPrice + law
fm.W <- ~ 1
est.zigp(DriversKilled, fm.X, fm.W, NULL, Offset = kms)

# ZIP with constant zero-inflation
fm.X <- ~ PetrolPrice + law
fm.Z <- ~ 1
est.zigp(DriversKilled, fm.X, NULL, fm.Z, Offset = kms)

# ZIGP with constant overdispersion and constant zero-inflation
fm.X <- ~ PetrolPrice + law
fm.W <- ~ 1
fm.Z <- ~ 1
est.zigp(DriversKilled, fm.X, fm.W, fm.Z, Offset = kms)
# no significant zero-inflation according to the Wald test
# (not surprising since not a single zero outcome in data)

# generate TeX output
est.zigp(DriversKilled, fm.X, fm.W, fm.Z, Offset = kms, tex=TRUE)

```

---

fit.zigp

*Fitted Values*


---

### Description

'fit.zigp' computes fitted values.

### Usage

```
fit.zigp(delta)
```

### Arguments

delta a parameter vector of length  $(p+r+q)$  with  $\dim(X) = (n \times p)$ ,  $\dim(W) = (n \times r)$ ,  $\dim(Z) = (n \times q)$ . Create delta by pasting '`delta <- c(beta, alpha, gamma)`'. beta is the vector of regression parameters for the mean modelling. alpha is the vector of regression parameters for overdispersion modelling. gamma is the vector of regression parameters for the ZI modelling.

**Details**

The design matrices have to be defined as Xsave (for mean), Wsave (for overdispersion) and Zsave (for ZI).

n has to be defined as the number of observations.

k.beta has to be defined as the length of beta.

k.alpha has to be defined as the length of alpha.

k.gamma has to be defined as the length of gamma.

t.i has to be defined as the exposure.

**Examples**

```
Xsave <- matrix(c(1:3,4,3,5),3,2)
Wsave <- c(3,-4,-1)
Zsave <- rep(1,3)
n <- dim(Xsave)[1]
beta <- c(5,-2)
alpha <- 3.4
gamma <- -10
k.beta <- length(beta)
k.alpha <- 1
k.gamma <- length(gamma)
t.i <- rep(1,n)
delta <- c(beta,alpha,gamma)
fit.zigp(delta)
#[1] $fit
#[2] [1,]
#[3] [1,] 0.04978481
#[4] [2,] 54.59567139
#[5] [3,] 148.40642146
#[6] $mu
#[7] [1,]
#[8] [1,] 0.04978707
#[9] [2,] 54.59815003
#[10] [3,] 148.41315910
#[11] $phi
#[12] [1] 26904.186074 1.000001 1.033373
#[13] $omega
#[14] [1] 4.539787e-05 4.539787e-05 4.539787e-05
```

---

fit.zigpl

*Non-executable auxiliary function*


---

**Description**

auxiliary function

FM

*Fisher Information***Description**

'FM' calculates the (Expected) Fisher Information matrix.

**Usage**

```
FM(beta, alpha, gamma, X, W, Z, Offset = NULL)
```

**Arguments**

beta	regression parameters for mean of length p
alpha	regression parameters for overdispersion of length r
gamma	regression parameters for zero inflation of length q
X	design matrix of dim (n x p) for mean modelling.
W	design matrix of dim (n x r) for overdispersion modelling.
Z	design matrix of dim (n x q) for zero inflation modelling.
Offset	exposure for individual observation lengths. Defaults to a vector of 1. The offset <b>MUST NOT</b> be in 'log' scale.

**Details**

n has to be defined as the number of observations.

**Examples**

```
X <- matrix(c(1:3, 4, 3, 5), 3, 2)
W <- c(3, -4, -1)
Z <- rep(1, 3)
n <- 3
beta <- c(5, -2)
alpha <- 3.4
gamma <- -10
FM(beta, alpha, gamma, X, W, Z)
#           [,1]      [,2]      [,3]      [,4]
#[1,]  1.469180e+03  2.412237e+03  6.249334e-03 -8.400641e-11
#[2,]  2.412237e+03  3.965799e+03  1.040260e-02 -3.360257e-10
#[3,]  6.249334e-03  1.040260e-02  2.101615e-03  2.520099e-10
#[4,] -8.400641e-11 -3.360257e-10  2.520099e-10  9.079162e-05
```

---

 gradient

*Gradient of log likelihood*


---

### Description

'gradient' calculates the gradient of the log likelihood function.

### Usage

```
gradient(delta)
```

### Arguments

delta a parameter vector of length  $(p+r+q)$  with  $\dim(X) = (n \times p)$ ,  $\dim(W) = (n \times r)$ ,  $\dim(Z) = (n \times q)$ . Create delta by pasting 'delta <- c(beta, alpha, gamma)'. beta is the vector of regression parameters for the mean modelling. alpha is the vector of regression parameters for overdispersion modelling. gamma is the vector of regression parameters for the ZI modelling.

### Details

This function used for the 'optim' method. Some 'optim' methods need a gradient and have to approximate it in every iteration step. This slows down the process a lot. Therefore, an explicit gradient function increases convergence speed and ensures correct convergence even if the log likelihood function is not continuous due to the indicator function of the ZI parameter.

The response has to be defined as Ysave. The design matrices have to be defined as Xsave (for mean), Wsave (for overdispersion) and Zsave (for ZI).

n has to be defined as the number of observations.

k.beta has to be defined as the length of beta.

k.alpha has to be defined as the length of alpha.

k.gamma has to be defined as the length of gamma.

t.i has to be defined as the exposure.

### Examples

```
Ysave <- c(3,0,2)
Xsave <- matrix(c(1:3,4,3,5),3,2)
Wsave <- c(3,-4,-1)
Zsave <- rep(1,3)
n <- dim(Xsave)[1]
beta <- c(5,-2)
alpha <- 3.4
gamma <- -10
k.beta <- length(beta)
k.alpha <- 1
k.gamma <- length(gamma)
```

```
t.i <- rep(1,n)
delta <- c(beta,alpha,gamma)
gradient(delta)
#[1] 423.8616107 704.1026859 7.5116293 -0.9998638
```

---

gradient1

*Non-executable auxiliary function*


---

### Description

auxiliary function

---

loglikelihood.zigp *Log Likelihood of ZIGP distribution*


---

### Description

'loglikelihood.zigp' calculates the log likelihood function of the ZIGP distribution.

### Usage

```
loglikelihood.zigp(delta)
```

### Arguments

delta a parameter vector of length  $(p+r+q)$  with  $\dim(X) = (n \times p)$ ,  $\dim(W) = (n \times r)$ ,  $\dim(Z) = (n \times q)$ . Create delta by pasting 'delta <- c(beta, alpha, gamma)'. beta is the vector of regression parameters for the mean modelling. alpha is the vector of regression parameters for overdispersion modelling. gamma is the vector of regression parameters for the ZI modelling.

### Details

The response has to be defined as Ysave. The design matrices have to be defined as Xsave (for mean), Wsave (for overdispersion) and Zsave (for ZI).

n has to be defined as the number of observations.

k.beta has to be defined as the length of beta.

k.alpha has to be defined as the length of alpha.

k.gamma has to be defined as the length of gamma.

Y has to be defined as the response vector.

t.i has to be defined as the exposure.

**Examples**

```

Ysave <- c(3,0,2)
Xsave <- matrix(c(1:3,4,3,5),3,2)
Wsave <- c(3,-4,-1)
Zsave <- rep(1,3)
n <- dim(Xsave)[1]
t.i <- rep(1,n)
beta <- c(5,-2)
alpha <- 3.4
gamma <- -10
k.beta <- length(beta)
k.alpha <- 1
k.gamma <- length(gamma)
delta <- c(beta,alpha,gamma)
loglikelihood.zigp(delta)
#[1] 160.2377

```

---

```
loglikelihood.zigp.full
```

*Non-executable auxiliary function*

---

**Description**

auxiliary function

---

```
loglikelihood.zigp.vuong
```

*Non-executable auxiliary function*

---

**Description**

auxiliary function

---

```
mle.zigp
```

*Maximum Likelihood Estimates*

---

**Description**

'mle.zigp' is used to calculate the MLEs of the regression parameters for mean, overdispersion and zero-inflation.

**Usage**

```

mle.zigp(Yin, fm.X, fm.W=NULL, fm.Z=NULL, Offset = rep(1, length(Yin)),
         init = TRUE, reltol = sqrt(.Machine$double.eps))

```

**Arguments**

Yin	Response vector of length n.
fm.X	Formula for mean design.
fm.W	Formula for overdispersion design (optional).
fm.Z	Formula for zero inflation design (optional).
Offset	Exposure for individual observation lengths. Defaults to a vector of 1. The offset <b>MUST NOT</b> be in 'log' scale.
init	A logical value indicating whether initial optimization values for dispersion are set to -2.5 and values for zero inflation regression parameters are set to -1 (init = F) or are estimated by a ZIGP(mu(i), phi, omega)-model (init = T). Defaults to 'T'.
reltol	Relative tolerance for 'optim' routine.

**Details**

Constant overdispersion and/or zero-inflation can be modelled using an Intercept design on the corresponding level. Setting fm.W to NULL corresponds to modelling a ZIP model. Setting fm.Z to NULL corresponds to modelling a GP model. Setting fm.W and fm.Z to NULL corresponds to modelling a Poisson GLM.

For numerical stability it may be very useful to center and standardize all non-categorical covariates, i.e. use `'x <- (x-mean(x))/sd(x)'`.

**References**

Czado, C., Erhardt, V., Min, A., Wagner, S. (2007) Zero-inflated generalized Poisson models with regression effects on the mean, dispersion and zero-inflation level applied to patent outsourcing rates. *Statistical Modelling* 7 (2), 125-153.

**Examples**

```
# Number of damages in car insurance.
# (not a good fit, just to illustrate how the software is used)

damage <- c(0,1,0,0,0,4,2,0,1,0,1,1,0,2,0,0,1,0,0,1,0,0,0)
insurance.year <- c(1,1.2,0.8,1,2,1,1.1,1,1,1.1,1.2,1.3,0.9,1.4,1,1,1,
1.2,1,1,1,1,1)
drivers.age <- c(25,19,30,48,30,18,19,29,24,54,56,20,38,18,23,58,
47,36,25,28,38,39,42)
# for overdispersion: car brand dummy in {1,2,3}, brand = 1 is reference
brand <- c(1,2,1,3,3,2,2,1,1,3,2,2,1,3,1,3,2,2,1,1,3,3,2)
# abroad: driver has been abroad for longer time (=1)
abroad <- c(0,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,1,0,1,0,1,1,1)
Y <- damage
fm.X <- ~ drivers.age
fm.W <- ~ 0 + factor(brand)
fm.Z <- ~ abroad

mle.zigp(Yin=Y, fm.X=fm.X, fm.W=fm.W, fm.Z=fm.Z, Offset = insurance.year,
```

```
init = FALSE)
```

---

```
mle.zigp.full.like Non-executable auxiliary function
```

---

### Description

auxiliary function

---

```
optimized.run Non-executable auxiliary function
```

---

### Description

auxiliary function

---

```
pzigp Distribution function of ZIGP distribution
```

---

### Description

'pzigp' calculates the distribution function of the ZIGP distribution.

### Usage

```
pzigp(x, mu, phi, omega)
```

### Arguments

x	vector of discrete points
mu	mean
phi	dispersion parameter
omega	zero inflation parameter

### Value

Calculates a vector of the same length as of x evaluating the ZIGP distribution function at x.

### Examples

```
x <- 0:10
pzigp(x, 2, 1.5, 0.2)
```

---

qzigp *Quantile function of ZIGP distribution*

---

**Description**

'qzigp' calculates the quantiles of the ZIGP distribution.

**Usage**

```
qzigp(p, mu, phi, omega)
```

**Arguments**

p	vector of probabilities
mu	mean
phi	dispersion parameter
omega	zero inflation parameter

**Value**

Calculates a vector of the same length as of p containing quantiles of the ZIGP distribution.

**Examples**

```
p <- seq(0, 1, 0.1)
qzigp(p, 2, 1.5, 0.2)
#[1] 0 0 0 0 0 1 1 2 3 4 75
```

---

response.zigp *ZIGP response generator*

---

**Description**

'response.zigp' generates a ZIGP response vector for a given setting of linear predictors.

**Usage**

```
response.zigp(X, W, Z, beta, alpha, gamma)
```

**Arguments**

X	design matrix of dim (n x p) for modelling of mean
W	design matrix of dim (n x r) for modelling of overdispersion
Z	design matrix of dim (n x q) for modelling of zero inflation
beta	regression parameters for mean of length p
alpha	regression parameters for overdispersion of length r
gamma	regression parameters for zero inflation of length q

**Value**

Generates a ZIGP response vector of length n.

**Examples**

```
X <- matrix(c(1:3, 4, 3, 5), 3, 2)
W <- c(3, -4, -1)
Z <- rep(1, 3)
beta <- c(5, -2)
alpha <- 3.4
gamma <- -10
# set seed for random variable generator
set.seed(1)
response.zigp(X, W, Z, beta, alpha, gamma)
#[1] 0 55 132
```

---

rzigp

*ZIGP random variable generator*


---

**Description**

'rzigp' generates ZIGP random variables using the inversion method.

**Usage**

```
rzigp(n, mu, phi, omega)
```

**Arguments**

n	length of output random vector.
mu	mean
phi	dispersion parameter
omega	zero inflation parameter

**Value**

Generates a ZIGP random vector of length n.

**Examples**

```
# set seed for random variable generator
set.seed(2)
rzigp(3, 2, 1.5, 0.2)
#[1] 3 1 0
```

---

summaryzignl	<i>Non-executable auxiliary function</i>
--------------	--

---

**Description**

auxiliary function

---

vuong	<i>Vuong's test for non-nested model comparison</i>
-------	---

---

**Description**

'vuong' suggests the better of two (not necessarily nested) models according to Vuong's statistic.

**Usage**

```
vuong(model1, model2, alpha=0.05, correction=T)
```

**Arguments**

model1, model2	the output of two model fits obtained by using 'mle.zignl'.
alpha	significance level, defaults to 0.05.
correction	boolean, if TRUE (default), the Schwarz correction will be used on the differences of log-likelihoods.

**References**

Vuong, Q.H. (1989). Likelihood Ratio tests for model selection and nonnested hypotheses. *Econometrica* 57(2), 307-333.

Schwarz, G. (1978). Estimating the Dimension of a Model. *Annals of Statistics* 6, 461-464.

**Examples**

```
data(Seatbelts)
DriversKilled <- as.vector(Seatbelts[,1])           # will be response
kms <- as.vector(Seatbelts[,5]/mean(Seatbelts[,5])) # will be exposure
PetrolPrice <- as.vector(Seatbelts[,6])           # will be covariate 1
law <- as.vector(Seatbelts[,8])                   # will be covariate 2

fm.X.poi <- ~ PetrolPrice + law

fm.X.gp <- ~ PetrolPrice + law
fm.W.gp <- ~ 1

fm.X.zignl <- ~ PetrolPrice + law
```

```

fm.W.zigp <- ~ 1
fm.Z.zigp <- ~ 1

poi <- mle.zigp(Yin=DriversKilled, fm.X=fm.X.poi, fm.W=NULL,
               fm.Z=NULL, Offset = kms, init = FALSE)
gp <- mle.zigp(Yin=DriversKilled, fm.X=fm.X.gp, fm.W=fm.W.gp,
               fm.Z=NULL, Offset = kms, init = FALSE)
zigp <- mle.zigp(Yin=DriversKilled, fm.X=fm.X.zigp, fm.W=fm.W.zigp,
                 fm.Z=fm.Z.zigp, Offset = kms, init = FALSE)
# it is possible to compare to a Negative Binomial fit:
library(MASS)
nb <- glm.nb(DriversKilled ~ offset(log(kms)) + PetrolPrice + law)

vuong(poi, gp)
vuong(gp, zigp)
vuong(poi, zigp)
vuong(gp, nb)

# compare: since gp and zigp are almost identical for this data (zero-
# inflation is estimated to be zero), the Schwarz correction for the
# (unnecessary) additional parameter has a great impact:

# GP with constant overdispersion
fm.X <- ~ PetrolPrice + law
fm.W <- ~ 1
est.zigp(DriversKilled, fm.X, fm.W, NULL, Offset = kms)

# ZIGP with constant overdispersion and constant zero-inflation
fm.X <- ~ PetrolPrice + law
fm.W <- ~ 1
fm.Z <- ~ 1
est.zigp(DriversKilled, fm.X, fm.W, fm.Z, Offset = kms)

vuong(gp, zigp, correction=FALSE)
vuong(gp, zigp)

```

# Index

## \*Topic **datagen**

fit.zigp, 8  
fit.zigp1, 10  
response.zigp, 17  
rzigp, 18

## \*Topic **distribution**

dzigp, 4  
FM, 10  
gradient, 11  
gradient1, 12  
loglikelihood.zigp, 12  
loglikelihood.zigp.full, 13  
loglikelihood.zigp.vuong, 13  
pzigp, 16  
qzigp, 16

## \*Topic **list**

summaryzigp1, 18

## \*Topic **models**

clarke, 2  
eda.od, 4  
eda.zi, 5  
est.zigp, 6  
mle.zigp, 14  
vuong, 19  
ZIGP-package, 1

## \*Topic **regression**

ZIGP-package, 1

## \*Topic **robust**

mle.zigp.full.like, 15  
optimized.run, 15

clarke, 2

dzigp, 4

eda.od, 4  
eda.zi, 5  
est.zigp, 6

fit.zigp, 8

fit.zigp1, 10

FM, 10

gradient, 11  
gradient1, 12

loglikelihood.zigp, 12  
loglikelihood.zigp.full, 13  
loglikelihood.zigp.vuong, 13

mle.zigp, 14  
mle.zigp.full.like, 15

optimized.run, 15

pzigp, 16

qzigp, 16

response.zigp, 17  
rzigp, 18

summaryzigp1, 18

vuong, 19

ZIGP (*ZIGP-package*), 1  
ZIGP-package, 1