

# NMixNB

## Parametrisation

The N-MixtureNB distribution is a negative Binomial mixture of the Binomials, as

$$\text{Prob}(y) = \sum_{n=y}^{\infty} \binom{n}{y} p^n (1-p)^{n-y} \times \frac{\Gamma(n+\delta)}{\Gamma(\delta)n!} q^\delta (1-q)^n$$

for responses  $y = 0, 1, 2, \dots, n$ , where  $n$  is Poisson number of trials, and  $p$  is probability of success. For  $\delta$  and  $q$ , see below. Replicated reponses  $y_1, y_2, \dots, y_r$ , are iid from the Binomial with the same  $p$ , conditioned on the same  $n$  from the negative Binomial,

$$\text{Prob}(y_1, \dots, y_r) = \sum_{n=\max\{y_1, \dots, y_r\}}^{\infty} \left\{ \prod_{i=1}^r \binom{n}{y_i} p^n (1-p)^{n-y_i} \right\} \times \frac{\Gamma(n+\delta)}{\Gamma(\delta)n!} q^\delta (1-q)^n$$

The negative binomial is parameterized in terms of the mean  $\lambda$  and overdispersion  $1/\delta$ , where  $q = \delta/(\delta + \mu)$ ; see the R documentation `?dnbinom` for this parameterisation (where  $\delta = \text{size}$ ).

## Link-function

The probability  $p$  is linked to the linear predictor by

$$p(\eta) = \frac{\exp(\eta)}{1 + \exp(\eta)}$$

for the default logit link, while  $\lambda$  depends on fixed covariates

$$\log(\lambda) = \sum_{j=1}^m \beta_j x_j$$

with one vector of covariates for each observation.  $m$  can be maximum 15 and minimum 1.

## Hyperparameters

The parameters  $\theta_1 = \beta_1, \theta_2 = \beta_2, \dots, \theta_m = \beta_m$ , and overdispersion  $\theta_{16} = \log(1/\delta)$ .

## Hyperparameter spesification and default values

**doc** NegBinomial-Poisson mixture

**hyper**

**theta1**

**hyperid** 101121

**name** beta1

**short.name** beta1

**initial** 2.30258509299405

**fixed** FALSE

**prior** normal

**param** 0 0.5

**to.theta** function(x) x

**from.theta** function(x) x

**theta2**

**hyperid** 101122  
**name** beta2  
**short.name** beta2  
**initial** 0  
**fixed** FALSE  
**prior** normal  
**param** 0 1  
**to.theta** function(x) x  
**from.theta** function(x) x

**theta3**

**hyperid** 101123  
**name** beta3  
**short.name** beta3  
**initial** 0  
**fixed** FALSE  
**prior** normal  
**param** 0 1  
**to.theta** function(x) x  
**from.theta** function(x) x

**theta4**

**hyperid** 101124  
**name** beta4  
**short.name** beta4  
**initial** 0  
**fixed** FALSE  
**prior** normal  
**param** 0 1  
**to.theta** function(x) x  
**from.theta** function(x) x

**theta5**

**hyperid** 101125  
**name** beta5  
**short.name** beta5  
**initial** 0  
**fixed** FALSE  
**prior** normal  
**param** 0 1  
**to.theta** function(x) x  
**from.theta** function(x) x

**theta6**

**hyperid** 101126  
**name** beta6  
**short.name** beta6

```

    initial 0
    fixed FALSE
    prior normal
    param 0 1
    to.theta function(x) x
    from.theta function(x) x
theta7
    hyperid 101127
    name beta7
    short.name beta7
    initial 0
    fixed FALSE
    prior normal
    param 0 1
    to.theta function(x) x
    from.theta function(x) x
theta8
    hyperid 101128
    name beta8
    short.name beta8
    initial 0
    fixed FALSE
    prior normal
    param 0 1
    to.theta function(x) x
    from.theta function(x) x
theta9
    hyperid 101129
    name beta9
    short.name beta9
    initial 0
    fixed FALSE
    prior normal
    param 0 1
    to.theta function(x) x
    from.theta function(x) x
theta10
    hyperid 101130
    name beta10
    short.name beta10
    initial 0
    fixed FALSE
    prior normal
    param 0 1

```

```

    to.theta function(x) x
    from.theta function(x) x
theta11
    hyperid 101131
    name beta11
    short.name beta11
    initial 0
    fixed FALSE
    prior normal
    param 0 1
    to.theta function(x) x
    from.theta function(x) x
theta12
    hyperid 101132
    name beta12
    short.name beta12
    initial 0
    fixed FALSE
    prior normal
    param 0 1
    to.theta function(x) x
    from.theta function(x) x
theta13
    hyperid 101133
    name beta13
    short.name beta13
    initial 0
    fixed FALSE
    prior normal
    param 0 1
    to.theta function(x) x
    from.theta function(x) x
theta14
    hyperid 101134
    name beta14
    short.name beta14
    initial 0
    fixed FALSE
    prior normal
    param 0 1
    to.theta function(x) x
    from.theta function(x) x
theta15
    hyperid 101135

```

```

    name beta15
    short.name beta15
    initial 0
    fixed FALSE
    prior normal
    param 0 1
    to.theta function(x) x
    from.theta function(x) x
theta16
    hyperid 101136
    name overdispersion
    short.name overdispersion
    initial 0
    fixed FALSE
    prior pc.gamma
    param 7
    to.theta function(x) log(x)
    from.theta function(x) exp(x)

```

**status** experimental

**survival** FALSE

**discrete** TRUE

**link** default logit loga probit

**pdf** nmixnb

## Specification

- `family="nmixnb"`
- Required arguments: the response  $Y$  and covariates  $X$  as `inla.mdata(Y, X [, additional.covariates])`

The response is a matrix where each row are replicates, where responses that are NA's are ignored. The covariates is one or many vectors, matrices or data.frames. Each row of the covariates  $(x_{i1}, x_{i2}, \dots, x_{im})$  defines the covariates used for the  $i$ 'th response(s) (the  $i$ 'th row of  $Y$ ). Note that  $\beta_{m+1}, \dots, \beta_{15}$  are fixed to zero.

## Example

In the following example we estimate the parameters in a simulated example with replications.

```

nrep.max = 5
n = 50
y = matrix(NA, n, nrep.max)
x = c()
xx = c()
size = 3

```

```

overdispersion = 1/size
intercept = 1

for(i in 1:n) {
  local.x = runif(1) - 0.5
  lambda = exp(2 + local.x)
  N = rnbino(1, mu=lambda, size=size)
  local.xx = runif(1) - 0.5
  eta = intercept + local.xx
  p = exp(eta)/(exp(eta) + 1)
  ## sample the number of replications
  nr = sample(1:nrep.max, 1)
  ## and sample these. note that 'y' is initialized with NA's,
  ## so if nr < nrep.max, then
  ## y[i,(nr+1):nrep.max] would be NA.
  y[i, 1:nr]= rbinom(nr, size = N, prob = p)
  x = c(x, local.x)
  xx = c(xx, local.xx)
}

Y = inla.mdata(y, 1, x)
r = inla(Y ~ 1 + xx,
  data = list(Y=Y, xx=xx),
  family = "nmixnb",
  control.fixed = list(prec.intercept=1, prec=1))

```

## Notes