NMix

Parametrisation

The N-Mixture distribution is a Poisson mixture of the Binomials, as

$$\text{Prob}(y) = \sum_{n=y}^{\infty} \binom{n}{y} p^n (1-p)^{n-y} \times \frac{\lambda^n}{n!} \exp(-\lambda)$$

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

$$\text{Prob}(y_1, \ldots, y_r) = \sum_{n=\max\{y_1,\ldots,y_r\}}^{\infty} \left\{ \prod_{i=1}^{r} \binom{n}{y_i} p^n (1-p)^{n-y_i} \right\} \times \frac{\lambda^n}{n!} \exp(-\lambda)$$

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 λ 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 10 and minimum 1.

Hyperparameters

The parameters $\theta_1 = \beta_1, \theta_2 = \beta_2, \ldots, \theta_m = \beta_m$.

Hyperparameter spesification and default values

doc Binomial-Poisson mixture

hyper

> **theta1**
>> **hyperid** 101101
>> **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** 101102

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 101103

    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 101104

    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 101105

    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 101106

    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 101107

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 101108

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 101109

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 101110

name beta10

short.name beta10

initial 0

fixed FALSE

prior normal

param 0 1

to.theta function(x) x

from.theta function(x) x

status experimental

survival FALSE

discrete TRUE

link default logit probit

pdf nmix

Specification

- family = `nmix`

- 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 $(x_{i1}, x_{i2}, \ldots, x_{im})$ defines the covariates used for the i'th response(s) (the i'th row of `Y`). Note that $\beta_{m+1}, \ldots, \beta_{10}$ 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()
intercept = 1

for(i in 1:n) {
    local.x = runif(1) - 0.5
    lambda = exp(2 + local.x)
    N = rpois(1, lambda)
    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 = "nmix",
```

```
control.fixed = list(prec.intercept=1,  prec=1))
```

Notes