Zero-inflated models: Beta-Binomial

Parameterisation

There is support for a further zero-inflated model of type 2 (see zero-inflated.pdf), the zero-inflated beta-binomial. It is only defined for type 2.

Type 2

The likelihood is defined as

$$\text{Prob}(y \mid \ldots) = p \times 1_{[y=0]} + (1-p) \times \text{Beta-binomial}(y)$$

where:

$$p = 1 - \left(\frac{\exp(x)}{1+\exp(x)}\right)^{\alpha}$$

Link-function

As for the Binomial (see Zero-inflated.pdf).

Hyperparameters

The Beta-binomial distribution has two arguments (β_1 & β_2) which we assume are a (specific) function of an underlying hyperparameter (δ) & x. There is a further hyperparameter, α, governing zero-inflation where:

The parameter controlling the degree of overdispersion, δ, is represented as

$$\theta_1 = \log(\delta)$$

and the prior is defined on θ_1.

The zero-inflation parameter α, is represented as

$$\theta_2 = \log(\alpha)$$

and the prior and initial value is is given for θ_2.

Specification

- family = `zeroinflatedbetabinomial2`
- Required arguments: As for the zero-inflated-nbinomial2 likelihood.

Example

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

```
Example-zero-inflated-beta-binomial2.R

nx = 1000                 # number of x's to consider
n.trial = 20              # size of each binomial trial
x = rnorm(nx)             # generating x
```

```
delta = 10                   #hyperparameter 1
p = exp(1+x)/(1+exp(1+x))    #hyperparameter 2
alpha = 2                      #ZI parameter
q = p^alpha                    #prob presence

beta_1=delta*p                    #beta-bin parameter 1
beta_2=delta*(1-p)                #beta-bin parameter 2
rb = rbeta(nx, beta_1, beta_2, ncp = 0)


y = rep(0,nx)                       #generating data
abs.pres = rbinom(nx,1,q)
y[abs.pres==1] = rbinom( sum(abs.pres>0), n.trial, rb[abs.pres==1])

formula = y ~ x +1
r = inla(formula, data = data.frame(x,y), family = "zeroinflatedbetabinomial2",
        control.data = list(prior = c("flat", "flat"),
                fixed = c(F,F)),
        Ntrials = rep(n.trial, nx),
        verbose=TRUE)
```