# Linkmodel: powerlogit

### **Parametrization**

This is the link that map  $p \in (0,1)$  into  $x \in \Re$ , where

$$F^{\beta}(x) = p, \qquad \beta > 0$$

and F(x) is the cumulative distribution function for the logit,

$$F(x) = \frac{1}{1 + \exp(-x)}.$$

This link is renormalized so its corresponding density have zero mean and unit variance for every value of  $\beta$ .

## Hyperparameters

The parameter  $\beta$  represent the power

$$\beta = \exp(\theta_1)$$

and the prior is defined on  $\theta_1$ .

The intercept is represented by a quantile level  $\alpha$ , where

$$\alpha = \frac{\exp(\theta_2)}{1 + \exp(\theta_2)}$$

and the prior is defined on  $\theta_2$ .

### **Specification**

Use model="powerlogit" within control.link.

# Hyperparameter spesification and default values

doc Power logit link

hyper

### theta1

**hyperid** 49131

name power

short.name power

initial 0.00123456789

fixed FALSE

**prior** normal

**param** 0 10

to.theta function(x) log(x)

from.theta function(x) exp(x)

### theta2

hyperid 49132

name intercept

short.name intercept

```
initial 0
          fixed FALSE
          prior logitbeta
          param 11
          to.theta function(x) log(x / (1 - x))
          from.theta function(x) exp(x) / (1 + exp(x))
pdf powerlogit
Example
map <- function(x) log(x/(1-x))
imap \leftarrow function(x) 1/(1+exp(-x))
power.link <- function(power) {</pre>
    qfunc <- function(q) -log(q^(-1/power) -1)
    dfunc <- function(x) power * exp(-x) / (1+exp(-x))^{(power + 1)}
    pfunc <- function(x) 1/(1+exp(-x))^power</pre>
    dq < -0.005
    qq \leftarrow unique(sort(c(0+exp(seq(-10, 0, by = dq)),
                           1-\exp(\text{seq}(-10, 0, \text{by = dq}))))
    qq \leftarrow qq[(qq < 1) & (qq > 0)]
    xx <- qfunc(qq)
    ## total probability, need to account for this in cdf/icdf so it will extrapolate no need
    ## do this in the m \& s calcuation
    p <- qq[length(qq)]</pre>
    dens <- dfunc(xx)</pre>
    dx \leftarrow (c(0, diff(xx)) + c(diff(xx), 0))/2
    dens \leftarrow (dens * dx) / sum(dens * dx)
    m = sum(xx*dens)
    s = sqrt(sum(xx^2*dens) - m^2)
    xx <- (xx-m)/s
    csum <- p * cumsum(dens)</pre>
    cdf <- splinefun(xx, map(csum), method = "monoH.FC")</pre>
    icdf <- splinefun(map(csum), xx, method = "monoH.FC")</pre>
    return(list(cdf = cdf, icdf = icdf, mean = m, sd = s))
}
n <- 500
size <- 25
power <- 0.25
link <- power.link(power)</pre>
x \leftarrow rnorm(n, sd = 0.5)
x \leftarrow x-mean(x)
intercept <- 1
prob <- imap(link$cdf(intercept + x))</pre>
y <- rbinom(n, size = size, prob = prob)
r \leftarrow inla(y ~1 + x,
```

```
data = data.frame(y, x),
          family = "binomial",
          Ntrials = size,
          inla.mode = "experimental",
          verbose = TRUE,
          control.inla = list(int.strategy = "eb"),## cmin = 0),
          control.predictor = list(compute = TRUE, link = 1),
          control.fixed = list(prec = 1,
                               remove.names = "(Intercept)"),
          control.family = list(
              ## start with the true values to demonstrate how its done
              control.link = list(
                  model = "powerlogit",
                  hyper = list(
                      intercept = list(initial = link$cdf(intercept),
                                       prior = "normal",
                                       param = c(0,1)),
                      power = list(initial = log(power),
                                   param = c(0, 1),
                                   fixed = FALSE)))),
          num.threads = "4:1")
summary(r)
## this should be, about, the values of the intercept
print(round(dig = 4, c(intercept = intercept,
                       approximate.intercept.est =
                           as.numeric(link$icdf(r$mode$theta[2])))))
```

### Notes

- This link is EXPERIMENTAL, and new default priors will be added in the near future
- Setting the initial value for the hyperparameter "intercept" to infinity, will remove the intercept from the link-model.