Proportional odds model

Parametrisation

The proportional odds model, is for discrete observations (here described with the logit cdf)

$$y \in \{1, 2, \ldots, K\}, \qquad K > 1,$$

defined via the cummulative distribution function

$$F(k) = \mathrm{Prob}(y \leq k) = \frac{\exp(\gamma_k)}{1 + \exp(\gamma_k)}$$

where

$$\gamma_k = \alpha_k - \eta.$$

$\{\alpha_k\}$ is here increasing sequence of $K - 1$ cut-off points,

$$\alpha_0 = -\infty < \alpha_1 < \alpha_2 < \ldots < \alpha_{K-1} < \alpha_K = \infty,$$

and η is the linear predictor. The likelihood for an observation is then

$$\mathrm{Prob}(y = k) = F(k) - F(k-1).$$

Link-function

Not available.

Hyperparameters

The hyperparameters are $\theta_1, \ldots, \theta_{K-1}$, where

$$\alpha_1 = \theta_1,$$

and

$$\alpha_k = \alpha_{k-1} + \exp(\theta_k) = \theta_1 + \sum_{j=2}^{k} \exp(\theta_j)$$

for $k = 2, \ldots, K-1$. The posteriors for $\{\alpha_k\}$ must be found through simulations as shown in the example below.

Specification

There is also an option to use the probit cdf instead of the logit as described above

- family = `pom`
- Required arguments: y (observations)
- Within `control.family`, add `control.pom=list(cdf="logit")` or `control.pom=list(cdf="probit")` to control the cdf used. Default is the "logit".
- For the `cdf="probit"`, then `fast=TRUE` in `control.pom` will use a faster but approximate implementation of the probit cdf (default `FALSE`).

Number of classes, K is determined as the maximum of the observations. Empty classes are not allowed.

Example: logit

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

```
rpom = function(alpha, eta)
{
    ## alpha: the cutpoints. eta: the linear predictor
    F = function(x) 1.0/(1+exp(-x))

    ns = length(eta)
    y = numeric(ns)
    nc = length(alpha) + 1

    for(k in 1:ns) {
        p = diff(c(0.0, F(alpha - eta[k]), 1.0))
        y[k] = sample(1:nc, 1, prob = p)
    }
    return (y)
}

n = 3000
nsim = 1E5
x = rnorm(n, sd = 0.3)

eta = x
alpha = c(-1, 0, 0.5)
y = rpom(alpha, eta)
prior.alpha = 3 ## parameter in the Dirichlet prior
r = inla(y ~ -1 + x, data = data.frame(y, x, idx = 1:n), family = "pom",
         control.family = list(hyper = list(theta1 = list(param = prior.alpha))))
summary(r)

## compute the posterior for the cutpoints
theta = inla.hyperpar.sample(nsim, r, intern=TRUE)
nms = paste(paste0("theta", 1:length(alpha)), "for POM")
sim.alpha = matrix(NA, dim(theta)[1], length(alpha))
for(k in 1:length(alpha)) {
    if (k == 1) {
        sim.alpha[, k] = theta[, nms[1]]
    } else {
        sim.alpha[, k] = sim.alpha[, k-1] + exp(theta[, nms[k]])
    }
}
colnames(sim.alpha) = paste0("alpha", 1:length(alpha))
m1 = colMeans(sim.alpha)
m2 = colMeans(sim.alpha^2)
print(cbind(truth = alpha, estimate = m1, stdev = sqrt(m2 - m1^2)))

for(k in 1:length(alpha)) {
    d = density(sim.alpha[, k])
    if (k == 1) {
        plot(d, xlim = 1.2*range(c(sim.alpha)),
             ylim = c(0, 1.5 * max(d$y)), type="l", lty=k, lwd=2)
    } else {
        lines(d, xlim = range(c(sim.alpha)), lty = k, lwd=2)
    }
    abline(v = alpha[k], lty=k, lwd=2)
}
```

Example: probit

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

```
rpom = function(alpha, eta)
{
    ## alpha: the cutpoints. eta: the linear predictor
    F = function(a, eta) pnorm(a-eta)

    ns = length(eta)
    y = numeric(ns)
    nc = length(alpha) + 1

    for(k in 1:ns) {
        p = diff(c(0.0, F(alpha, eta[k]), 1.0))
        y[k] = sample(1:nc, 1, prob = p)
    }
    return (y)
}

library(INLA)
##inla.setOption(inla.mode = "experimental")

n = 3000
nsim = 1E5
x = rnorm(n, sd = 0.3)

eta = x + rnorm(n, sd = 0.2)
alpha = c(-1, 0, 0.5, 1.25)
y = rpom(alpha, eta)
xx = inla.group(x)

r = inla(y ~ -1 +
             f(xx, model="rw2", scale.model=TRUE,
               hyper = list(prec = list(prior = "pc.prec",
                                        param = c(0.5, 0.01)))) +
             f(idx, model="iid",
               hyper = list(prec = list(prior = "pc.prec",
                                        param = c(0.5, 0.01)))),
         data = data.frame(y, x, idx = 1:n, xx),
         family = "pom",
         control.family = list(control.pom = list(cdf = "probit")),
         ##control.inla = list(cmin=0),
         control.fixed = list(prec = 1, prec.intercept = 1))

theta = inla.hyperpar.sample(nsim, r, intern=TRUE)
nms = paste(paste0("theta", 1:length(alpha)), "for POM")
sim.alpha = matrix(NA, dim(theta)[1], length(alpha))
for(k in 1:length(alpha)) {
    if (k == 1) {
        sim.alpha[, k] = theta[, nms[1]]
    } else {
        sim.alpha[, k] = sim.alpha[, k-1] + exp(theta[, nms[k]])
    }
}
colnames(sim.alpha) = paste0("alpha", 1:length(alpha))
cbind(truth = alpha,
      estimated = colMeans(sim.alpha),
```

```
      "stdev(estimate)" = sqrt(colMeans(sim.alpha^2) - colMeans(sim.alpha)^2))

for(k in 1:length(alpha)) {
    d = density(sim.alpha[, k])
    if (k == 1) {
        plot(d, xlim = 1.2*range(c(sim.alpha)), ylim = c(0, 1.5 * max(d$y)), type="l", lty=k, lwd=2)
    } else {
        lines(d, xlim = range(c(sim.alpha)), lty = k, lwd=2)
    }
    abline(v = alpha[k], lty=k, lwd=2)
}
```

Notes

The prior for $\{\theta_k\}$ are fixed to the Dirichlet distribution for

$$\big(F^{-1}(\alpha_1), \quad F^{-1}(\alpha_2) - F^{-1}(\alpha_1), \quad F^{-1}(\alpha_3) - F^{-1}(\alpha_2), \quad \ldots, \quad 1 - F^{-1}(\alpha_{K-1})\big)$$

with a common scale parameter; see `inla.doc("dirichlet", sec="prior")`