

# Discrete generalized Pareto distribution

## Parametrisation

The discrete generalized Pareto (GP) distribution with positive shape parameter has cumulative distribution function

$$F(y; \sigma, \xi) = 1 - \left(1 + \xi \frac{y+1}{\sigma}\right)^{-1/\xi}, \quad y = 0, 1, 2, \dots$$

for a discrete response  $y$  where

$\xi$ : is the tail parameter,  $\xi > 0$

$\sigma$ : is the scale parameter,  $\sigma > 0$

## Link function

The linear predictor  $\eta$  controls the  $\alpha$  quantile of the corresponding continuous GP

$$P(y \leq q_\alpha) = \alpha$$

and  $q_\alpha = \exp(\eta)$ . The scaling  $\sigma$ , is then a function of  $(q_\alpha, \xi)$ , as

$$\sigma = \frac{\xi \exp(\eta)}{(1 - \alpha)^{-\xi} - 1}$$

## Hyperparameters

The GP model has one hyperparameter. The tail  $\xi > 0$  is represented as

$$\xi = \xi_{\text{low}} + (\xi_{\text{high}} - \xi_{\text{low}}) \frac{\exp(\theta)}{1 + \exp(\theta)}$$

and the prior is defined on  $\theta$ , with constant low and high values. The prior is FIXED to `pc.gevtail`, see `inla.doc("pc.gevtail")` for more info.

## Specification

- `family=dgp`
- Required arguments:  $y$  and the quantile  $\alpha$ .

The quantile is given as `control.family=list(control.link=list(quantile= $\alpha$ ))`.

## Hyperparameter specification and default values

**doc** Discrete generalized Pareto likelihood

**hyper**

**theta**

**hyperid** 101201

**name** tail

**short.name** xi

**initial** 2

**fixed** FALSE

**prior** pc.gevtail

**param** 7 0 0.5

**to.theta** function(x, interval = c(REPLACE.ME.low, REPLACE.ME.high)) log(-(interval

**from.theta** function(x, interval = c(REPLACE.ME.low, REPLACE.ME.high)) interval[1]

**status** experimental

**survival** FALSE

**discrete** TRUE

**link** default quantile

**pdf** dgp

## Example

```
F = function(y, sigma, xi) 1.0 - (1.0 + xi * (y+1)/sigma)^(-1/xi)
```

```
f = function(y, sigma, xi) F(y, sigma, xi) - F(y-1, sigma, xi)
```

```
rdgp = function(n, sigma, eta, alpha, xi = 0.001)
{
  if (missing(sigma)) {
    stopifnot(!missing(eta) && !missing(alpha))
    stopifnot(length(eta) == 1)
    sigma = exp(eta) * xi / ((1.0 - alpha)^(-xi) - 1.0)
  }
  stopifnot(length(sigma) == 1)
  eps = 1e-7
  y.max = ceiling((eps^(-xi) - 1) * sigma/xi)
  return (sample(0:y.max, prob = f(0:y.max, sigma, xi),
                size=n, replace=TRUE))
}
```

```

n = 300
x = runif(n)-0.5
eta = 5+x
alpha = 0.95
xi = 0.3
y = numeric(n)
for(i in 1:n) {
  y[i] = rdgp(1, eta = eta[i], alpha = alpha, xi=xi)
}

r = inla(y ~ 1+x,
        data = data.frame(y, x),
        family = "dgp",
        control.family = list(
          control.link = list(quantile = alpha),
          hyper = list(tail = list(
            prior = "pc.gevtail",
            param = c(7, 0.0, 0.5))))),
        control.predictor = list(compute=TRUE),
        verbose=TRUE)

summary(r)
plot(r, plot.prior=TRUE)

dev.new()
plot(cbind(r$summary.fitted.values$mean, exp(eta)))
abline(a=0, b=1)

```

## Notes