

## qLogLogistic likelihood

### Parametrisation

The LogLogistic distribution has cumulative distribution function

$$F_0(y) = \frac{1}{1 + \lambda y^{-\alpha}}, \quad y > 0$$

if `variant=0`, or

$$F_1(y) = \frac{1}{1 + (\lambda y)^{-\alpha}}, \quad y > 0$$

if `variant=1`, where

$\alpha > 0$  is a shape parameter, and

$\lambda > 0$  is a scale parameter.

The  $\lambda$  is defined implicitly through the quantile, as

$$F_0(y_q) = q, \quad \text{or} \quad F_1(y_q) = q, \quad 0 < q < 1$$

and the linear predictor is defined on  $y_q$ .

### Link-functions

The parameter  $\lambda$  is linked to the linear predictor, implicitly through

$$y_q = \exp(\eta)$$

### Hyperparameters

The  $\alpha$  parameter is represented as

$$\theta = \log \alpha$$

and the prior is defined on  $\theta$ .

### Specification

- `family="qloglogistic"` (regression) or `family="qloglogistic.surv"` (survival)
- `variant=0` (default) or 1, choosing between parameterisation  $F_0$  or  $F_1$ .
- Required arguments:  $y$  (regression) or an `inla.surv`-object using `inla.surv()` (for survival data), and `quantile=q`.

### Hyperparameter specification and default values

#### Regression:

**doc** A quantile loglogistic likelihood

**hyper**

**theta**

**hyperid** 60011

**name** log alpha

```

    short.name alpha
    initial 1
    fixed FALSE
    prior loggamma
    param 25 25
    to.theta function(x) log(x)
    from.theta function(x) exp(x)

survival FALSE

discrete FALSE

link default log neglog

pdf qloglogistic

Survival:

doc A quantile loglogistic likelihood (survival)

hyper

```

```

    theta1
        hyperid 60021
        name log alpha
        short.name alpha
        initial 1
        fixed FALSE
        prior loggamma
        param 25 25
        to.theta function(x) log(x)
        from.theta function(x) exp(x)

```

```

    theta2
        hyperid 60022
        name beta1
        short.name beta1
        initial -5
        fixed FALSE
        prior normal
        param -4 100
        to.theta function(x) x
        from.theta function(x) x

```

```

    theta3
        hyperid 60023
        name beta2
        short.name beta2
        initial 0
        fixed FALSE
        prior normal

```

```

    param 0 100
    to.theta function(x) x
    from.theta function(x) x
theta4
    hyperid 60024
    name beta3
    short.name beta3
    initial 0
    fixed FALSE
    prior normal
    param 0 100
    to.theta function(x) x
    from.theta function(x) x
theta5
    hyperid 60025
    name beta4
    short.name beta4
    initial 0
    fixed FALSE
    prior normal
    param 0 100
    to.theta function(x) x
    from.theta function(x) x
theta6
    hyperid 60026
    name beta5
    short.name beta5
    initial 0
    fixed FALSE
    prior normal
    param 0 100
    to.theta function(x) x
    from.theta function(x) x
theta7
    hyperid 60027
    name beta6
    short.name beta6
    initial 0
    fixed FALSE
    prior normal
    param 0 100
    to.theta function(x) x
    from.theta function(x) x
theta8

```

```

hyperid 60028
name beta7
short.name beta7
initial 0
fixed FALSE
prior normal
param 0 100
to.theta function(x) x
from.theta function(x) x
theta9
hyperid 60029
name beta8
short.name beta8
initial 0
fixed FALSE
prior normal
param 0 100
to.theta function(x) x
from.theta function(x) x
theta10
hyperid 60030
name beta9
short.name beta9
initial 0
fixed FALSE
prior normal
param 0 100
to.theta function(x) x
from.theta function(x) x
theta11
hyperid 60031
name beta10
short.name beta10
initial 0
fixed FALSE
prior normal
param 0 100
to.theta function(x) x
from.theta function(x) x

```

**survival** TRUE

**discrete** FALSE

**link** default log neglog

**pdf** qloglogistic

## Example

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

```
lam_loglogistic = function(yq, alpha, q, variant = 0)
{
  if (variant == 0) {
    lambda = yq^alpha * (1/q-1)
  } else if (variant == 1) {
    lambda = 1/yq * (1/(1/q-1))^(1/alpha)
  } else
    stop("ERR")
  return (lambda)
}

rloglogistic = function(n, lambda, alpha, variant=0)
{
  u = runif(n)
  if (variant == 0) {
    y = (lambda/(1.0/u - 1.0))^(1.0/alpha)
  } else if (variant == 1) {
    y = (1.0/(1.0/u -1.0))^(1.0/alpha) / lambda
  } else {
    stop("ERROR")
  }
}

n = 500
alpha = 2.1
x = c(scale(runif(n)))
eta = 1.1+2.2*x
yq = exp(eta)

for(variant in 0:1) {
  for(q in c(0.2, 0.8)) {

    print(paste("variant=", variant, "quantile=", q))
    lambda = lam_loglogistic(yq, alpha, q, variant=variant)
    y = rloglogistic(n,
                     lambda = lambda,
                     alpha = alpha,
                     variant = variant)

    formula = y ~ 1 + x
    rr=inla(formula,
            family ="qloglogistic",
            data=data.frame(y, x),
            control.family = list(list(variant = variant, control.link = list(quantile = q))
    print("REGRESSION")
    print(summary(rr))
```

```

event = rep(1,n)
formula=inla.surv(y,event) ~ 1 + x
r=inla(formula,
      family ="qloglogisticsurv",
      data = list(y=y, event=event, x=x),
      control.family = list(list(variant = variant, control.link = list(quantile = q)
print("SURVIVAL")
print(summary(r))
}
}

```

## Notes

- Loglogisticsurv model can be used for right censored, left censored, interval censored data. If the observed times  $y$  are large/huge, then this can cause numerical overflow in the likelihood routine. If you encounter this problem, try to scale the observations, `time = time / max(time)` or similar.