# Sigmoidal effect of a covariate

## Parametrization

This model implements a non-linear effect of a positive covariate $x$ as a part of the linear predictor. It comes in two variants, *sigmoidal*

$$\beta \frac{x^k}{x^k + a^k} \qquad \text{or} \qquad \beta \frac{z}{z+1}$$

with $z = (x/a)^k$, $x \geq 0$, $k > 0$ and $a > 0$, and the *reverse-sigmoidal*

$$\beta \frac{a^k}{x^k + a^k} \qquad \text{or} \qquad \beta \frac{1}{z+1}.$$

Here, $a$ is the halflife parameter, $k$ the shape-parameter and $\beta$ the scaling.

## Hyperparameters

This model has three hyperparameters, the scaling $\beta$, halflife $a$ and shape $k$,

$$\theta_1 = \beta \qquad \theta_2 = \log(a) \qquad \theta_3 = \log(k)$$

and the priors are given for $\theta_1, \theta_2$ and $\theta_3$.

## Specification

```
f(x, model="sigm",    hyper = ..., precision = <precision>)
f(x, model="revsigm", hyper = ..., precision = <precision>)
```

where `precision` is the precision for the tiny noise used to implement this as a latent model.

### Hyperparameter specification and default values

**doc** Sigmoidal effect of a covariate

**hyper**

- **theta1**
  - **hyperid** 38001
  - **name** beta
  - **short.name** b
  - **initial** 1
  - **fixed** FALSE
  - **prior** normal
  - **param** 1 10
  - **to.theta** `function(x) x`
  - **from.theta** `function(x) x`
- **theta2**
  - **hyperid** 38002
  - **name** loghalflife
  - **short.name** halflife
  - **initial** 3

**fixed** FALSE

**prior** loggamma

**param** 3 1

**to.theta** `function(x) log(x)`

**from.theta** `function(x) exp(x)`

**theta3**

**hyperid** 38003

**name** logshape

**short.name** shape

**initial** 0

**fixed** FALSE

**prior** loggamma

**param** 10 10

**to.theta** `function(x) log(x)`

**from.theta** `function(x) exp(x)`

**constr** FALSE

**nrow.ncol** FALSE

**augmented** FALSE

**aug.factor** 1

**aug.constr**

**n.div.by**

**n.required** FALSE

**set.default.values** FALSE

**status** experimental

**pdf** sigm

## Example

```
sigm = function(x, halflife, shape = 1)
{
    xx = (x/halflife)^shape
    return (xx/(1.0+xx))
}
revsigm = function(x, halflife, shape = 1)
{
    xx = (x/halflife)^shape
    return (1.0/(1.0+xx))
}

n = 1000
lambda = 10
s=0.01
x = rpois(n, lambda = lambda)
```

```
halflife = lambda
shape = 2

y = sigm(x, halflife, shape) + rnorm(n, sd = s)
r = inla(y ~ -1 + f(x, model="sigm"),
        data = data.frame(y, x),
        family = "gaussian",
        control.family = list(
                hyper = list(
                        prec = list(
                                initial = log(1/s^2),
                                fixed = TRUE))))
summary(r)

y = revsigm(x, halflife, shape) + rnorm(n, sd = s)
r = inla(y ~ -1 + f(x, model="revsigm"),
        data = data.frame(y, x),
        family = "gaussian",
        control.family = list(
                hyper = list(
                        prec = list(
                                initial = log(1/s^2),
                                fixed = TRUE))))
summary(r)
```

## Notes

None