

# Besag2 model for weighted spatial effects

## Parametrization

The besag2 model is an extension to the besag model. Let the random vector  $\mathbf{z} = (x_1, \dots, x_n)$  be the besag model, then the besag2 is the following extensions

$$\mathbf{x} = (a\mathbf{z}, \mathbf{z}/a)$$

where  $a > 0$  is an additional hyperparameter and  $\dim(\mathbf{x}) = 2n$ , and  $\mathbf{z}$  is the *same* (up to tiny additive noise) random vector.

## Hyperparameters

This model has two hyperparameters  $\theta = (\theta_1, \theta_2)$ .

The precision parameter  $\tau$  is represented as

$$\theta_1 = \log \tau$$

and the prior is defined on  $\theta_1$ .

The weight-parameter  $a$  is represented as

$$\theta_2 = \log a$$

and the prior is defined on  $\theta_2$ .

## Specification

The besag2 model is specified inside the `f()` function as

```
f(<whatever>, model="besag2", graph=<graph>
    precision=<precision>, hyper = <hyper>,
    adjust.for.con.comp = TRUE,
    scale.model = FALSE)
```

The precision is the precision defining how equal the two copies of  $\mathbf{z}$  is. The neighbourhood structure of  $\mathbf{x}$  is passed to the program through the `graph` argument.

Note that the besag2 model has dimension  $2n$ , where  $n$  is the size of the graph.

If the option `adjust.for.con.comp=TRUE` then the model is adjusted if the graph has more than one connected component. This adjustment can be disabled setting this option to `FALSE`. If `adjust.for.con.comp=TRUE` then `constr=TRUE` is interpreted as a sum-to-zero constraint on *each* connected component in the graph and the `rankdef` parameter is set depending on the number of connected components.

The logical option `scale.model` determine if the model  $z$  should be scaled to have an average variance (the diagonal of the generalized inverse) equal to 1. This makes prior specification much easier. Default is `FALSE` so that the model is not scaled.

## Hyperparameter specification and default values

**doc** The shared Besag model

**hyper**

**theta1**

**hyperid** 9001

```

    name log precision
    short.name prec
    prior loggamma
    param 1 5e-05
    initial 4
    fixed FALSE
    to.theta function(x) log(x)
    from.theta function(x) exp(x)
  theta2
    hyperid 9002
    name scaling parameter
    short.name a
    prior loggamma
    param 10 10
    initial 0
    fixed FALSE
    to.theta function(x) log(x)
    from.theta function(x) exp(x)

  constr FALSE

  nrow.ncol FALSE

  augmented FALSE

  aug.factor 1

  aug.constr 1 2

  n.div.by 2

  n.required TRUE

  set.default.values TRUE

  pdf besag2

```

## Example

This is a simulated example.

```

data(Oral)
g = system.file("demodata/germany.graph", package="INLA")

## use data Oral to estimate a spatial field in order to simulate a
## 'realistic' dataset.
formula = Y ~ f(region, model="bym", graph=g)
result = inla(formula, data = Oral, family = "poisson", E = E)

x = result$summary.random$region$mean
n = length(x)/2

```

```

## simulate two new datasets. 'a' is the weighting between the
## log.rel.risk:
a = 2
xx = x[1:n]+1
x = c(a*xx, xx/a)
E = c(Oral$E, Oral$E)
N = 2*n
y = rpois(N, lambda = E*exp(x))

## model='besag2' defines a model with length N = 2*graph->n, the
## first half is weighted with 'a' the other half is weighted with
## 1/a. here there is no unstructured terms.
i = 1:N
formula = y ~ f(i, model="besag2", graph=g) -1
r = inla(formula, family = "poisson", data = data.frame(E,y,i), E=E, verbose=TRUE)

```

## Details on the implementation

This gives some details of the implementation, which depends on the following variables

**nc1** Number of connected components in the graph with size 1. These nodes, *singletons*, have no neighbours.

**nc2** Number of connected components in the graph with size  $\geq 2$ .

**scale.model** The value of the logical flag, if the model should be scaled or not. (Default FALSE)

**adjust.for.con.comp** The value of the logical flag if the `constr=TRUE` option should be reinterpreted.

**The case** (`scale.model==FALSE && adjust.for.con.comp == FALSE`)

The option `constr=TRUE` is interpreted as a sum-to-zero constraint over the whole graph. Singletons are given a uniform distribution on  $(-\infty, \infty)$  before the constraint.

**The case** (`scale.model==TRUE && adjust.for.con.comp == FALSE`)

The option `constr=TRUE` is interpreted as a sum-to-zero constraint over the whole graph. Let  $Q = \tau R$  be the standard precision matrix from the **besag**-model with precision parameter  $\tau$ . Then  $R$ , except the singletons, are scaled so that the geometric mean of the marginal variances is 1, and  $R$  is modified so that singletons have a standard Normal distribution.

**The case** (`scale.model==FALSE && adjust.for.con.comp == TRUE`)

The option `constr=TRUE` is interpreted as one sum-to-zero constraint over each of the **nc2** connected components of size  $\geq 2$ . Singletons are given a uniform distribution on  $(-\infty, \infty)$ .

**The case** (`scale.model==TRUE && adjust.for.con.comp == TRUE`)

The option `constr=TRUE` is interpreted as **nc2** sum-to-zero constraints for each of the connected components of size  $\geq 2$ . Let  $Q = \tau R$  be the standard precision matrix from the **besag**-model with precision parameter  $\tau$ . Then  $R$ , are scaled so that the geometric mean of the marginal variances in each connected component of size  $\geq 2$  is 1, and modified so that singletons have a standard Normal distribution.

## Notes

The `besag2` model has default `constr=FALSE`, and `constr=TRUE` does not make sense.