

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 τ is represented as

$$\theta_1 = \log \tau$$

and the prior is defined on θ_1 .

The weight-parameter a is represented as

$$\theta_2 = \log a$$

and the prior is defined on θ_2 .

Specification

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

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

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.file` argument. The structure of this file is described below.

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

The option `adjust.for.con.comp` adjust the model if the graph has more than one connected component, and this adjustment can be disabled setting this option to `FALSE`.

Hyperparameter specification and default values

hyper

theta1

```
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
  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

```

Structure of the graph file

We describe the required format for the graph file using a small example. Let the file `gra.dat`, relative to a small graph of only 5 elements, be

```

5
1 1 2
2 2 1 3
3 3 2 4 5
4 1 3
5 1 3

```

Line 1 declares the total number of nodes in the graph (5), then, in lines 2-6 each node is described. For example, line 4 states that node 3 has 4 neighbours and these are nodes 2, 4 and 5.

The graph file can either have nodes indexed from 1 to n , or from 0 to $n - 1$. Note that in the latter case, node i seen from R corresponds to node $i - 1$ in the 0-indexed graph.

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.file=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.file=g) -1
r = inla(formula, family = "poisson", data = data.frame(E,y,i), E=E, verbose=TRUE)

```

Notes

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