Generic 2 model

Parametrization

The generic2 model implements the following precision matrix

$$\mathbf{Q} = \begin{bmatrix} \tau_u \boldsymbol{I} & -\tau_u \boldsymbol{I} \\ -\tau_u \boldsymbol{I} & \tau_u \boldsymbol{I} + \tau_v \boldsymbol{C} \end{bmatrix} \tag{1}$$

where $\mathbf{C}$ is (a given) symmetric matrix. This model arrives from the hierarchical model,

$$\boldsymbol{v} \sim \mathcal{N}(\mathbf{0}, \tau_v \boldsymbol{C})$$

and

$$\boldsymbol{u} \mid \boldsymbol{v} \sim \mathcal{N}(\boldsymbol{v}, \tau_u \boldsymbol{I})$$

and the precision matrix in Eq. (1) implements the joint precision matrix of

$$\begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \end{bmatrix}$$

using the following hyperparameters

$$\tau_v \qquad \text{and} \qquad h^2 = \frac{1/\tau_v}{1/\tau_v + 1/\tau_u}.$$

Hyperparameters

The two parameters in the `generic2` model are represented as

$$\begin{aligned} \theta_1 &= \log(\tau_v) \\ \theta_2 &= \log(\tau_u) \end{aligned}$$

and priors are assigned to (θ_1, θ_2).

YES THIS IS CORRECT! This is because the prior spesification would then be similar to another spesification, but allows one to compute the posterior marginal for h^2 more easily.

Specification

The generic2model is specified inside the `f()` function as

```
f(<whatever>,model="generic2",Cmatrix = <Cmat>,
              prior=c(<prior.model.theta1>,<prior.model.theta2>),
              param=c(<param.prior.theta1>,<param.prior.theta1>,
                      <param.prior.theta2>,<param.prior.theta2>))
```

where `<Cmat>` can be given in two different ways:

- a list of type `Cmatrix = list(i = c(), j = c(), values = c())`, where `i`, `j` and `values` are vectors of the non-zero elements of $\mathbf{C}$. Note that `i` and `j` start from 1, and only the upper or lower part of $\mathbf{C}$ need to be given.
- the name of a file giving the structure matrix. The file should have the following format

  $$i \quad j \quad \mathbf{C}_{ij}$$

  where i and j are the row and column index and $\mathbf{C}_{ij}$ is the corresponding element of the precision matrix. Only the non-zero elements of the precision matrix need to be stored in the file. **NOTE:** the indexes for i and j start from 0, as this matrix is passed directly into the `inla`-program.

Example

```
require(mvtnorm)

n = 200
Cm = matrix(runif(n^2,min=-1,max=1),n,n)
Cm = Cm %*% t(Cm)
Sigma = solve(Cm)

sd = 0.001
z = rnorm(n,sd=sd)
eta = rmvnorm(n=1,sigma = Sigma)
s = 0.1
y = c(eta) + rnorm(n,sd=s) + z

file = "Cmatrix.dat"
cat("",file=file, append = FALSE)
for(i in 1:n)
{
    j = i
    cat(i,j,Cm[i,j], "\n", sep = " ", file=file, append=TRUE)
    if (i < n)
        for(j in (i+1):n)
            cat(i, j, Cm[i,j], "\n", sep = " ", file=file, append=TRUE)
}


idx = 1:n
formula = y ~ f(idx, model = "generic2", Cmatrix = file,
                initial=c(0,0), fixed=c(F,F))

result = inla(formula, data=data.frame(y,idx),
              control.data = list(initial = log(1/sd^2), fixed=TRUE),
              verbose = TRUE)


## tau.u should be about 1/s^2 = 100. increase 'n' above to get
## it...
tau.u = result$summary.hyperpar["Precision-cmatrix for idx", "mean"]
h2 = result$summary.hyperpar["h2 for idx", "mean"]
tau.v = h2/(1-h2)*tau.u
print(paste("tau.v", tau.v, "should be (for large n)", 1/s^2))
```

Notes

The option `constr=TRUE` will impose a sum-to-zero constraint on $\boldsymbol{v}$ only.