The Classical Measurement Error (MEC) model

Parametrization

This is an implementation of the classical ME model for a fixed effect. It is best described by an example, let the model be

$$y = \beta x + \epsilon$$

where y is the response, β the effect of the true covariate x with zero mean Gaussian noise ϵ. The issue is that x is not observed directly, but only through x_{obs}, where

$$x_{\text{obs}} = x + \nu$$

where ν is zero mean Gaussian noise. Even though this setup is possible to implement using basic features ("copy" and multiple likelihoods), we provide the following model which replaces the above,

$$y = u + \epsilon$$

where u has the correct distribution depending on various parameters: β has prior $\pi(\beta)$, x is apriori $\mathcal{N}(\mu_x I, \tau_x I)$, and $s \times \tau_{\text{obs}}$ is the observation precision for x (ie $\text{Prec}(x_{\text{obs}}|x)$).[1] Here, s is a vector of fixed scalings.

Hyperparameters

This model has 4 hyperparameters, $\theta = (\theta_1, \theta_2, \theta_3, \theta_4)$ where θ_2, θ_3 and θ_4 are default set to be fixed (ie defined to be known). The values of θ_2, θ_3 and θ_4 are set to mimic a classical fixed effect, so they will always make sense. To achieve the ME model, please use the appropriate choices for (some of) these parameters!

The hyperparameter specification is as follows:

$$\theta_1 = \beta$$

and the prior is defined on θ_1,

$$\theta_2 = \log(\tau_{\text{obs}})p$$

and the prior is defined on θ_2,

$$\theta_3 = \mu_x$$

and the prior is defined on θ_3,

$$\theta_4 = \log(\tau_{\text{x}})$$

and the prior is defined on θ_4.

Specification

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

```
f(x.obs, [<weights>,] model="mec", hyper = <hyper>, scale = <s>)
```

The `x.obs` are the observed values of the unknown covariates x, with the *assumption*, that if two or more elements of `x.obs` are *identical*, then they refer to the *same* element in the true covariate x. The fixed scaling of the observational precision is given in argument `scale`. If the argument `scale` is not given, then s is set to 1.

[1] Note: The second argument in $\mathcal{N}(,)$ is the precision not the variance.

Hyperparameter specification and default values

hyper

- **theta1**
  - **name** beta
  - **short.name** b
  - **prior** gaussian
  - **param** 1 0.001
  - **initial** 1
  - **fixed** FALSE
  - **to.theta** `function(x) x`
  - **from.theta** `function(x) x`
- **theta2**
  - **name** prec.obs
  - **short.name** prec
  - **prior** loggamma
  - **param** 1 1e-04
  - **initial** 9.21034037197618
  - **fixed** TRUE
  - **to.theta** `function(x) log(x)`
  - **from.theta** `function(x) exp(x)`
- **theta3**
  - **name** mean.x
  - **short.name** mu.x
  - **prior** gaussian
  - **param** 0 1e-04
  - **initial** 0
  - **fixed** TRUE
  - **to.theta** `function(x) x`
  - **from.theta** `function(x) x`
- **theta4**
  - **name** prec.x
  - **short.name** prec.x
  - **prior** loggamma
  - **param** 1 10000
  - **initial** -9.21034037197618
  - **fixed** TRUE
  - **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 mec

Example

```
n = 100
prec.y = 100
prec.obs = 10
prec.x = 1
## true unobserved covariate
x = rnorm(n, sd = 1/sqrt(prec.x))
## the observed covariate
xobs = x + rnorm(n, sd = 1/sqrt(prec.obs))
## regression model using the unobserved 'x'
y = 1 + 4*x + rnorm(n, sd = 1/sqrt(prec.y))

## prior parameters
prior.prec = c(1, 0.01)
prior.beta = c(0, 0.1)

formula = y ~ 1 +
    f(xobs, model="me",
      hyper = list(
              beta = list(
                      param = prior.beta,
                      fixed = FALSE
                      ),
              prec.obs = list(
                      param = prior.prec,
                      initial = log(prec.obs),
                      fixed = TRUE
                      ),
              prec.x = list(
                      param = prior.prec,
                      initial = log(prec.x),
                      fixed = FALSE
                      ),
              mean.x = list(
                      initial = 0,
                      fixed=TRUE
                      )
              )
      )
```

```
r = inla(formula,
        data = data.frame(y, xobs),
        family = "gaussian",
        control.family = list(
                hyper = list(
                        prec = list(param = prior.prec,
                                initial = log(prec.y),
                                fixed=FALSE
                                )
                        )
                )
        )

summary(r)
```

Notes

- `INLA` provide the posterior of u and NOT x.
- The posterior of u comes in the order given by the sorted (from low to high) values of `x.obs`. The entry `$ID` gives the mapping.