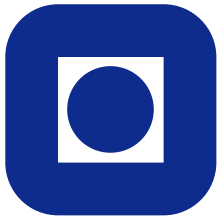


NTNU
Norwegian University of
Science and Technology

Introduction to the R-INLA R package

Elias T. Krainski

elias@r-inla.org.

Outline

Overview

Tokyo example

GAG urine data

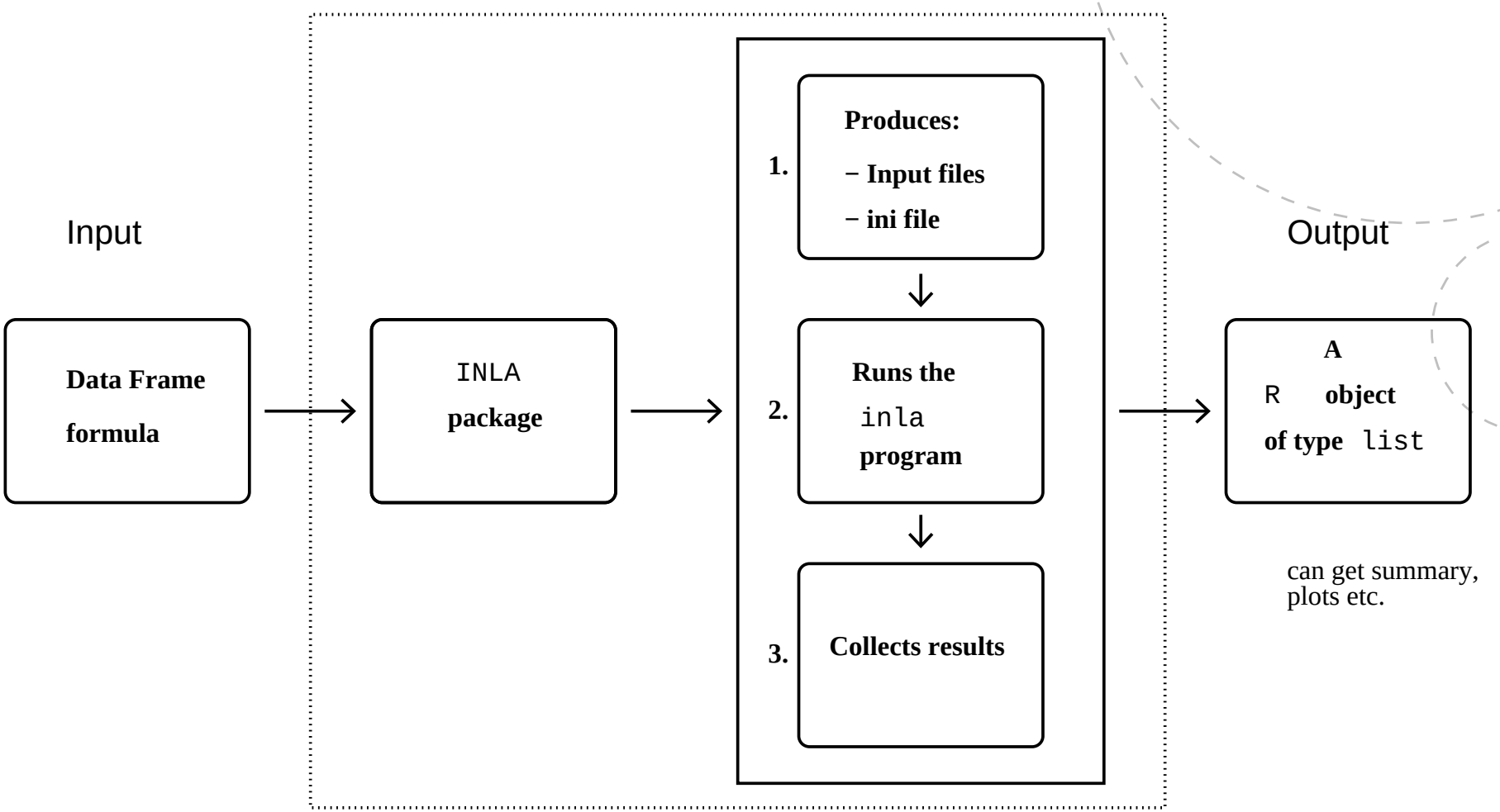
Close look at GAG urine example

R-INLA details

What is R-INLA?

- The algorithm name:
  Integrated Nested Laplace Aproximations - **INLA**
- R-INLA:
  - The **R**-package that implements **INLA**
  - R-INLA is a collection of R-code and C-code
    - C-code: fast computations for **GMRF**
    - R-code: to have it available for R-users
  - available at www.r-inla.org
- all the source code is public available at
  code.google.com/p/inla

R-INLA sheme


Input
Data Frame
formula
INLA
package
1.
Produces:
– Input files
– ini file
2.
Runs the
inla
program
3.
Collects results
Output
A
R object
of type list
can get summary,
plots etc.

How to install R-INLA?

new way (from 3rd January 2015)

```
> install.packages("INLA",
      repos="http://www.math.ntnu.no/inla/R/stable")
```

or

```
> install.packages("INLA",
      repos="http://www.math.ntnu.no/inla/R/testing")
```

R-INLA is in continuous development:

```
> inla.upgrade()
```

or

```
> inla.upgrade(testing=TRUE)
```

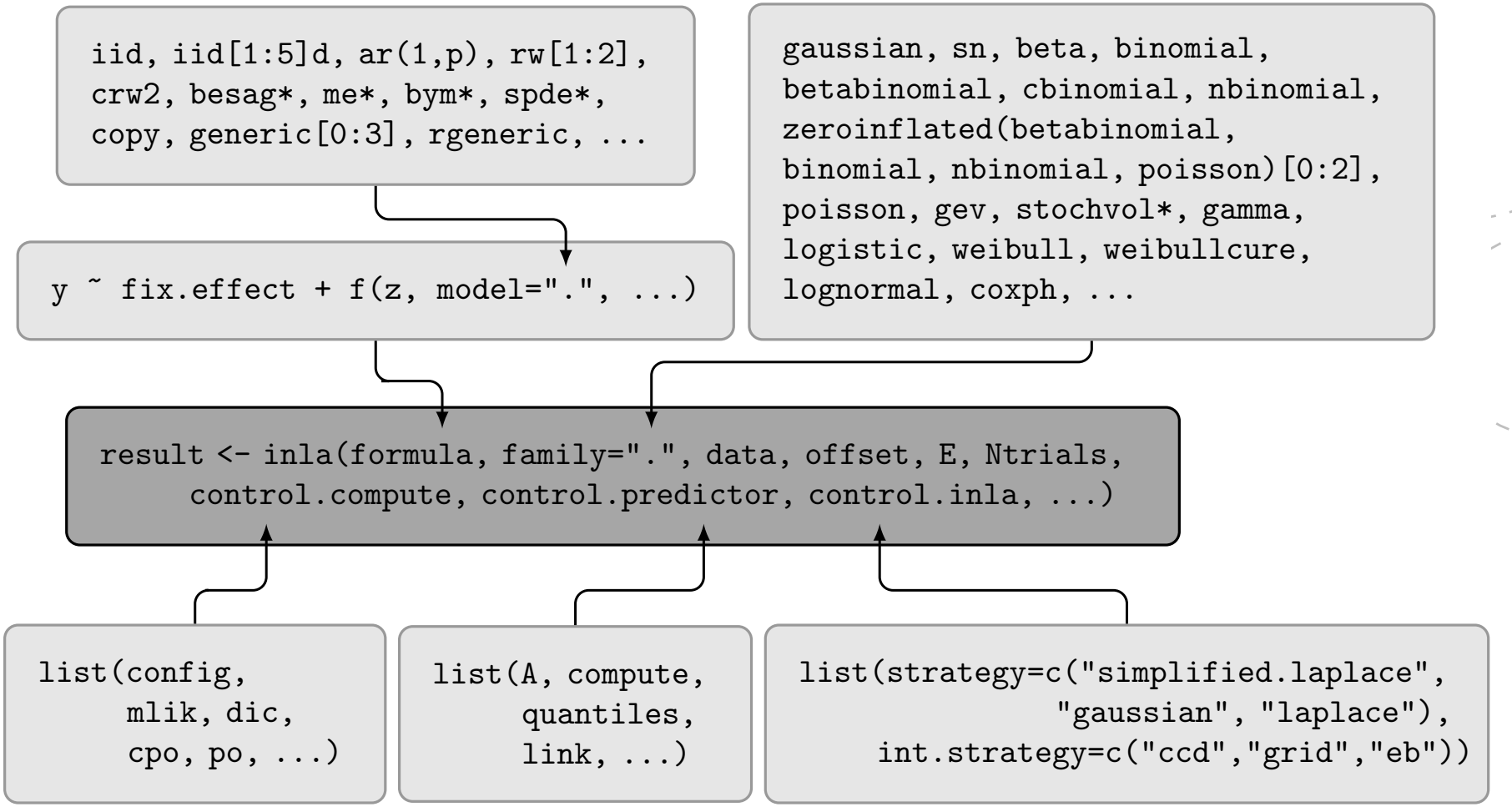
NTNU
Norwegian University of
Science and Technology

Data analisys steps using `R-INLA`

Five steps to fit an model

1. data organization
   response, covariates and index sets (for random effects)
2. Define the priors
3. define the `formula`
   - notation (similar to `lm` and `glm` functions
   - `f(.)` term to define random effects
4. call to the `inla`-program
5. extraction of posterior information

R-INLA **picture**



Outline

Overview

Tokyo example

GAG urine data

Close look at GAG urine example

R-INLA details

R-code for Tokyo example `rw1`

The dataset

```
data(Tokyo)
Tokyo[1:3,]
```

```
##    y n time
## 1 0 2    1
## 2 0 2    2
## 3 1 2    3
```

Define the formula (include hyperprior)

```
pcprec <- list(theta=list(
  prior='pc.prec', param=c(2, 0.05)))
formula1 <- y ~ 0 +
 f(time, model='rw1',
  cyclic=TRUE, constr=FALSE,
   scale.model=TRUE, hyper=pcprec)
```

```
tokyo1 <- inla(formula1, data=Tokyo, Ntrials=n,
 family='binomial', control.predictor=list(compute=TRUE),
  control.compute=list(dic=TRUE, cpo=TRUE))
```

R-code: Tokyo example `rw2`

Define the formula

```
formula2 = y ~ 0 +
 f(time, model='rw2', cyclic=TRUE, constr=FALSE,
   scale.model=TRUE, hyper=pcprec)
```

Fit the model

```
tokyo2 <- inla(formula2, data=Tokyo, Ntrials=n,
 family='binomial', control.predictor=list(compute=TRUE),
  control.compute=list(dic=TRUE, cpo=TRUE))
```

Tokyo `rw1` summary

```
summary(tokyo1)


##
## Call:
## c("inla(formula = formula1, family = \"binomial\", data = Tokyo, Ntrials = n, ",  "    control.compute
##
## Time used:
##  Pre-processing    Running inla Post-processing           Total
##          0.1061          0.2719          0.1436          0.5216
##
## The model has no fixed effects
##
## Random effects:
## Name   Model
##  time   RW1 model
##
## Model hyperparameters:
##                       mean     sd 0.025quant 0.5quant 0.975quant   mode
## Precision for time 1.045 0.7309      0.274   0.8489      2.971 0.5847
##
## Expected number of effective parameters(std dev): 22.51(5.173)
## Number of equivalent replicates : 16.26
##
## Deviance Information Criterion: 615.16
## Effective number of parameters: 22.73
##
## Marginal Likelihood:  -944.22
## CPO and PIT are computed
##
## Posterior marginals for linear predictor and fitted values computed
```

NTNU
Norwegian University of
Science and Technology

elias@r-inla.org

Some Tokyo results `rw1`

Random effects

```
round(tokyo1$summary.random$time[1:3,], 4)
```

```
##   ID    mean     sd 0.025quant 0.5quant 0.975quant    mode kld
## 1  1 -1.6343 0.4473    -2.5294  -1.6306    -0.7614 -1.6242   0
## 2  2 -1.6119 0.4468    -2.5039  -1.6090    -0.7380 -1.6040   0
## 3  3 -1.5734 0.4478    -2.4627  -1.5721    -0.6933 -1.5705   0
```

Fitted values

```
round(tokyo1$summary.fitted.values[1:3,], 4)
```

```
##                          mean     sd 0.025quant 0.5quant 0.975quant   mode
## fitted.predictor.001 0.1721 0.0628     0.0739   0.1637     0.3182 0.1476
## fitted.predictor.002 0.1752 0.0637     0.0757   0.1667     0.3234 0.1503
## fitted.predictor.003 0.1808 0.0655     0.0786   0.1719     0.3333 0.1549
```

Compare

Precision

```
round(rbind(rw1=tokyo1$summary.hy[, 1:6],
            rw2=tokyo2$summary.hy[, 1:6]), 4)

##        mean     sd 0.025quant 0.5quant 0.975quant   mode
## rw1 1.0448 0.7309     0.2740   0.8489     2.9713 0.5847
## rw2 1.7152 4.5392     0.1129   0.6668     9.7417 0.2374
```

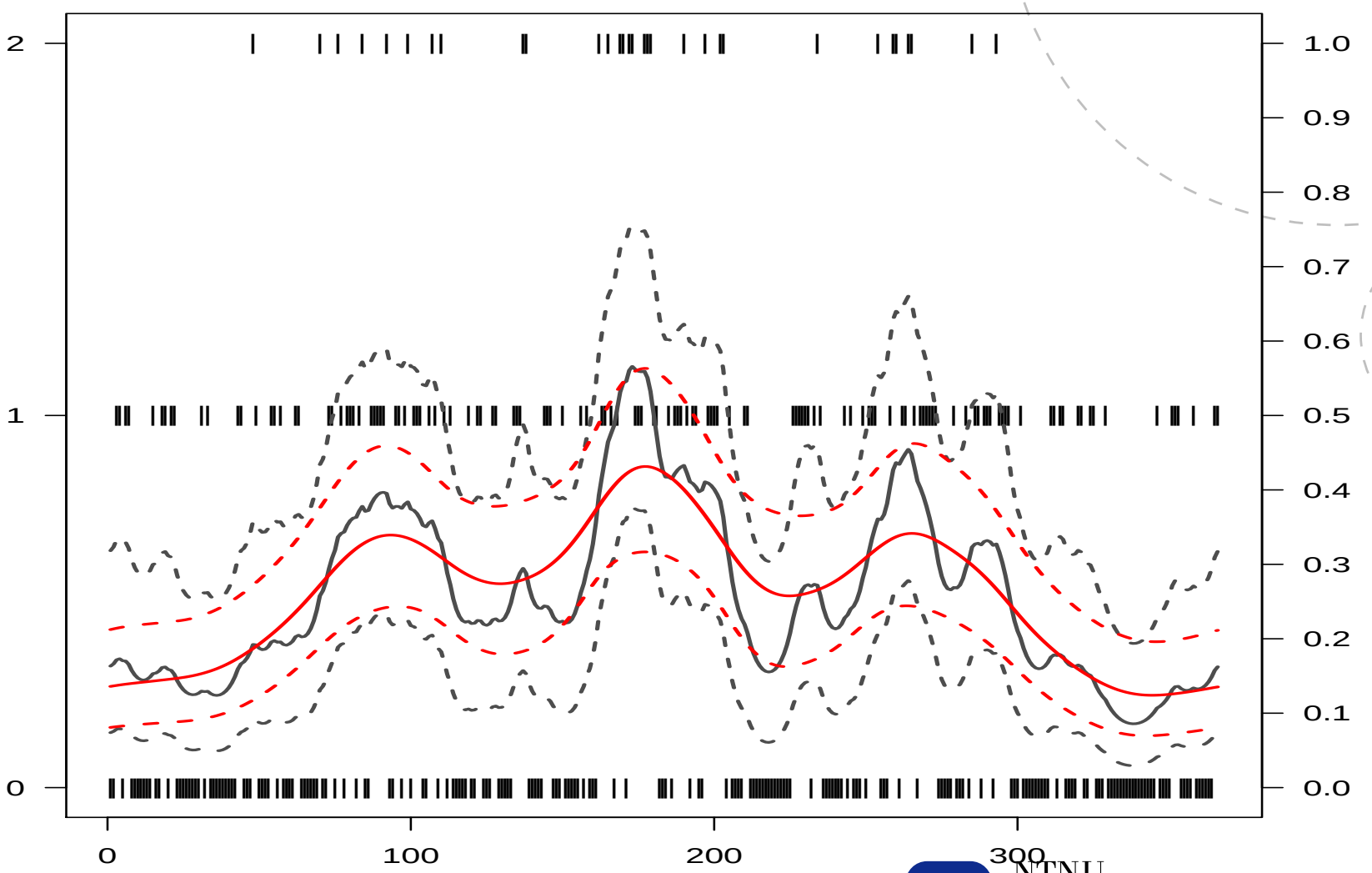
DIC, -LCPO

```
round(rbind(DIC=c(rw1=tokyo1$dic$dic, rw2=tokyo2$dic$dic),
 LCPO=c(sum(-log(tokyo1$cpo$cpo)), sum(-log(tokyo2$cpo$cpo))

##          rw1    rw2
## DIC  615.16 632.16
## LCPO 307.42 316.17
```

NTNU
Norwegian University of
Science and Technology

Tokyo results



Outline

Overview

Tokyo example

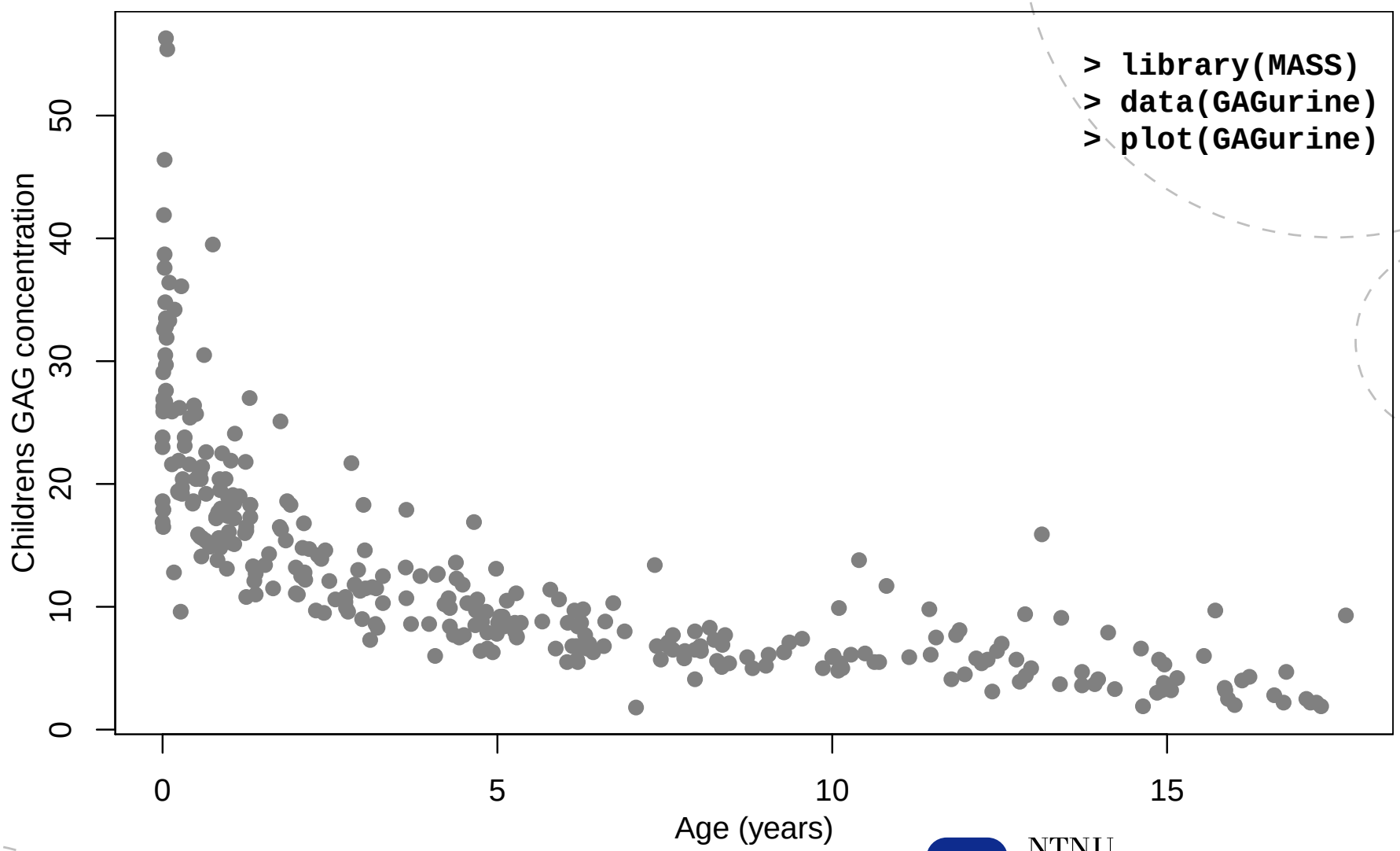
GAG urine data

Close look at GAG urine example

R-INLA details

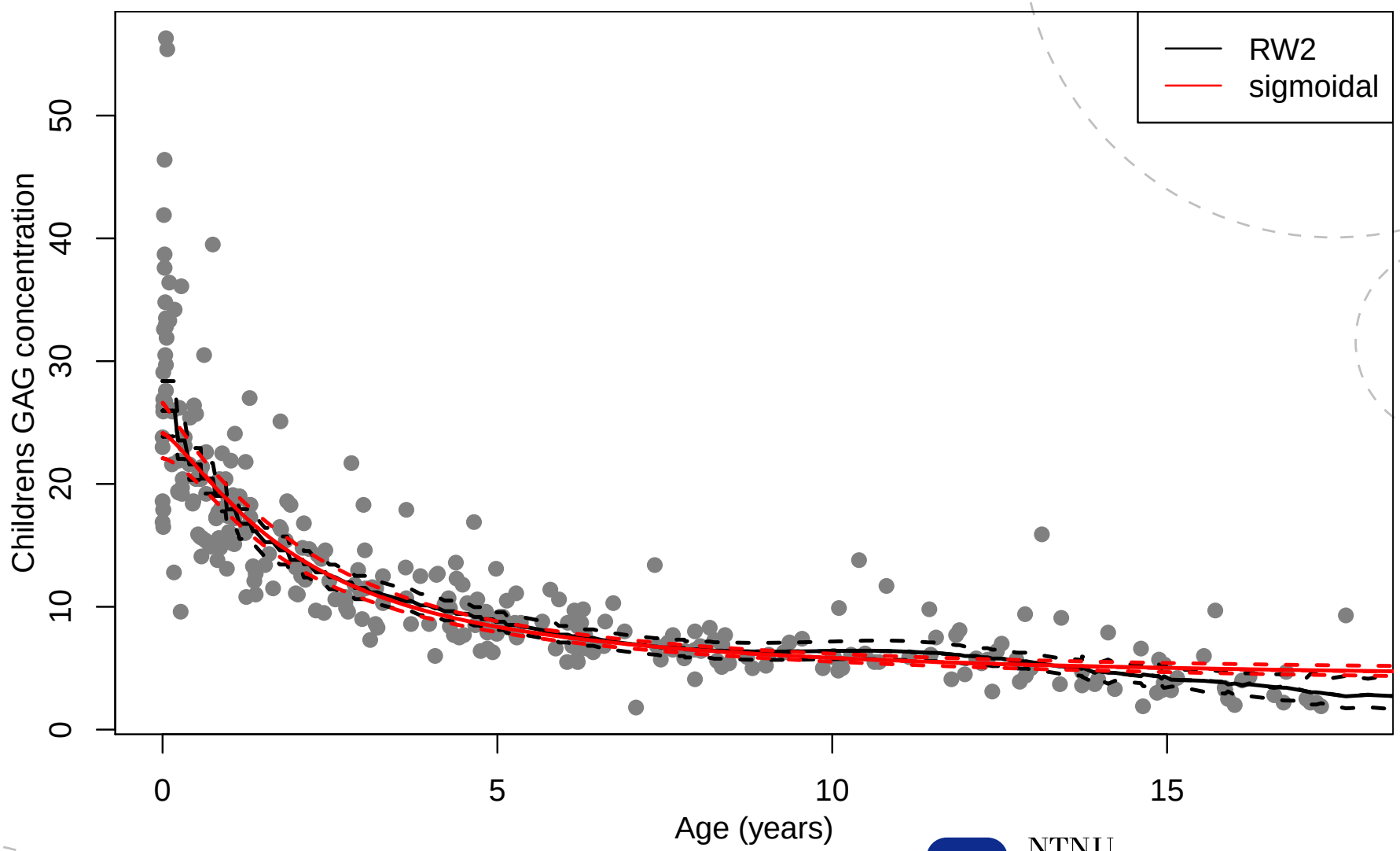
GAG urine

```
> library(MASS)
> data(GAGurine)
> plot(GAGurine)
```



NTNU
Norwegian University of
Science and Technology

GAG urine aim



GAG urine problem

— Problem: build a chart for GAG urine

$$E(GAG|age) = \mu(age)$$

— $\mu(age)$ must be positive
$\rightarrow$ Easy: model for the log of GAG

— How to relat *GAG* and age?
$\rightarrow$ People usually uses parametric non-linear models

— but we have INLA...
$\rightarrow$ Lets define a semi-parametric one

GAG urine model

— linear model for y = log (GAG)

$$y = \beta_0 + f(\text{age}) + \text{error}$$

error $\sim N(0, \sigma_e^2)$

— non linear effect from age

$$f(\text{age}) = x_{\text{age}}$$

where x for some age is similar to x for neighbour ages

— define some grid of ages: $a_1, a_2, ..., a_k$

— random walk order 1

$$x_{a_i} - x_{a_{i-1}} \sim N(0, \sigma^2)$$

— random walk order 2 (smoother)

$$x_{a_i} - (2x_{a_{i-1}} - x_{a_{i-2}}) \sim N(0, \sigma_x^2)$$

GAG urine analysis

GAG urine for 314 children aged from 0 to 17 years

```
require(MASS)
data(GAGurine)
GAGurine[c(1, 314), ]
```

```
##         Age  GAG
## 1      0.00 23.0
## 314 17.67  9.3
```

Prepare the data

```
GAGurine$y =
    log(GAGurine$GAG)
```

Define the formula

```
gag.form = y ~
 f(inla.group(Age), model='rw2',
   scale.model=TRUE, hyper=pcprec)
```

Fit the model

```
gag.rw2 <- inla(gag.form,
                data=GAGurine)
```

Outline

Overview

Tokyo example

GAG urine data

Close look at GAG urine example

- GAG reference limits

R-INLA details

GAG urine detailed

```
## prepare the data to predict new observations
gag.dat <- list(lgag=c(log(GAGurine$GAG), NA, NA),
                age=c(GAGurine$Age, 18, 19)) ## extrapolate
## two models for the latent effect
gag.rw2.form <- lgag ~ f(inla.group(age, 100), model='rw2',
  scale.model=TRUE, hyper=list(theta=list(prior='pc.prec', p
gag.sigm.form <- lgag ~ f(age, model='sigm')
## ask for more results
gag.rw2 <- inla(gag.rw2.form, data=gag.dat,
 control.predictor=list(compute=TRUE), ### to predict
 control.compute=list(config=TRUE, ## store configurations
    dic=TRUE, cpo=TRUE, po=TRUE), ## to compare
 control.inla=list(strategy='laplace')) ### best strategy
```

GAG model results

```
c(dic1=gag.rw2$dic$dic, lcpo1=-sum(log(gag.rw2$cpo$cpo), na.rm=TRUE),
  dic2=gag.sigm$dic$dic, lcpo2=-sum(log(gag.sigm$cpo$cpo), na.rm=TRUE))

##      dic1     lcpo1      dic2     lcpo2
## 147.17325  82.78154 175.27124  88.45040
```

```
round(gag.rw2$summary.hy[, 1:2], 4)

##                                            mean      sd
## Precision for the Gaussian observations 11.4996  0.9312
## Precision for inla.group(age, 100)      15.2599 12.4739
```

```
round(gag.sigm$summary.hy[, 1:2], 4)

##                                            mean     sd
## Precision for the Gaussian observations 10.0998 0.8267
## SIGM beta for age                       -1.8636 0.1119
## SIGM halflife for age                    4.0198 0.4519
## SIGM shape for age                       1.2921 0.1229
```

NTNU
Norwegian University of
Science and Technology

Prediction-extrapolation

```
cbind(true=c(GAGurine$GAG[314], NA, NA), # with 314
      rw2=exp(gag.rw2$summary.fitted.val[314:316, 1]),
      sigm=exp(gag.sigm$summary.fitted.val[314:316, 1]))

##      true      rw2     sigm
## [1,]  9.3 2.712722 4.804307
## [2,]   NA 2.808732 4.780401
## [3,]   NA 2.585115 4.715064
```

Error in the linear predictor scale

```
rbind(rw2=gag.rw2$summary.fitted.val[314:316, 2],
      sigm=gag.sigm$summary.fitted.val[314:316, 2])

##              [,1]       [,2]       [,3]
## rw2  0.24052898 0.20339756 0.31173712
## sigm 0.04285738 0.04343319 0.04510057
```

The variance

— compute the distribution for $\sigma|y$ (std.dev.)

```
tau.marginal <- gag.rw2$marginals.hyperpar[[1]]
s.marginal <- inla.tmarginal(function(x) sqrt(1/x), tau.marginal)
```

— summarize a marginal

```
inla.zmarginal(s.marginal)

## Mean            0.29561
## Stdev           0.0118601
## Quantile  0.025 0.273197
## Quantile  0.25  0.287356
## Quantile  0.5   0.295233
## Quantile  0.75  0.303454
## Quantile  0.975 0.319792
```

— other functions to work with marginals

```
apropos('marginal')

##  [1] "inla.dmarginal"   "inla.emarginal"   "inla.hpdmarginal"
##  [4] "inla.mmarginal"   "inla.pmarginal"   "inla.qmarginal"
##  [7] "inla.rmarginal"   "inla.smarginal"   "inla.tmarginal"
## [10] "inla.zmarginal"   "s.marginal"
```

GAG limits: The problem

- Define limits (2.5%, 95.5%) for GAG for each age
- The statistical model is for the mean: $E(y)$
- Additionally we have the *error*: GAG = E(GAG) + error

GAG limits: The problem

- Define limits (2.5%, 95.5%) for GAG for each age
- The statistical model is for the mean: $E(y)$
- Additionally we have the *error*: GAG = E(GAG) + error
- Easy way solution:
  - sample from $E(y)$ and the error
  - compute the sum
  - define the interval

GAG limits: code

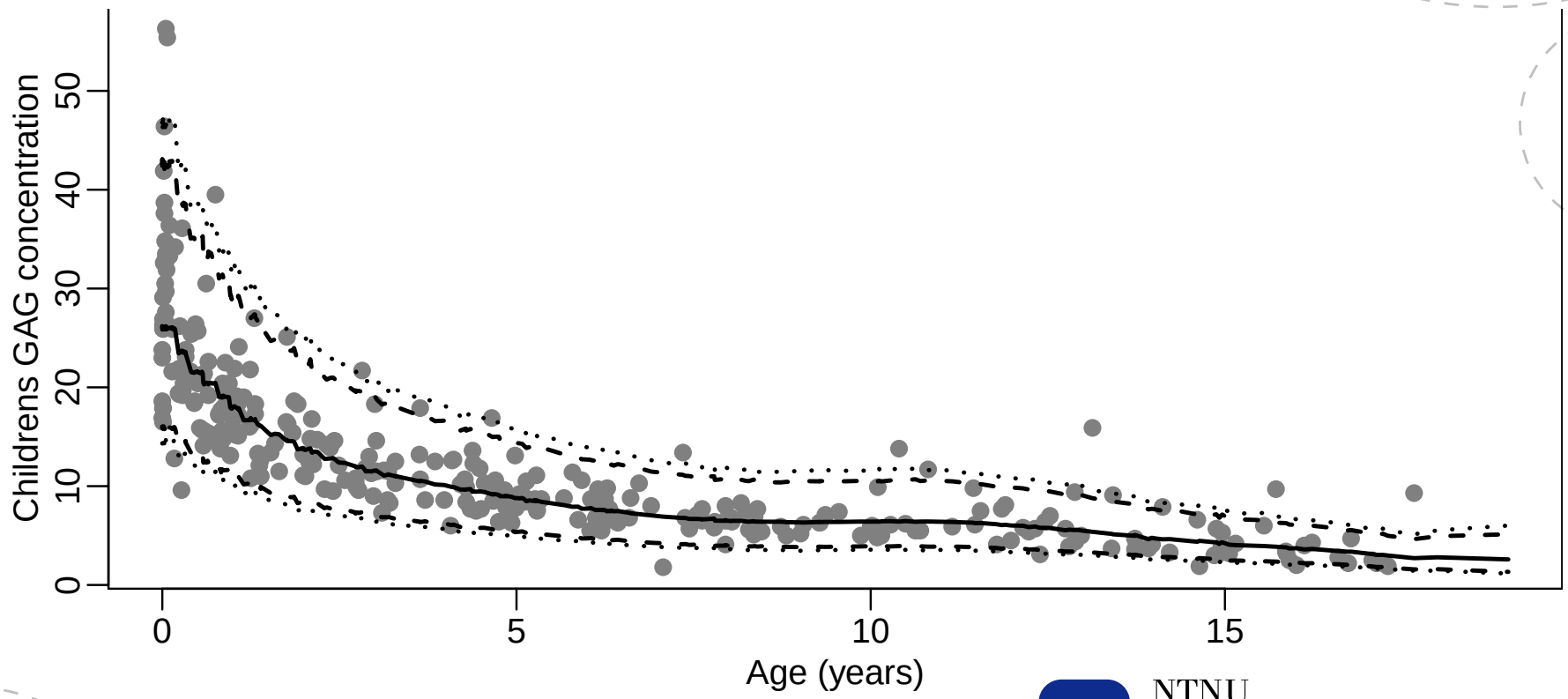
get samples from the posterior distribution

```
samples <- inla.posterior.sample(n=10000, res=gag.rw2)
```

for each sample, sample the error and compute the sum

```
psam <- sapply(samples, function(x)  {
    s.d <- sqrt(1/x$hyperpar[1])
    mu <- x$latent[1:316,1]
    err <- rnorm(316, 0, s.d)
    return(exp(mu+err)) ## in the GAG original scale
})
q.sam <- apply(psam, 1, quantile,
    c(.025, 0.05, 0.5, 0.95, .975))
```

```
par(mar=c(2.5, 2.5, 0, 0), mgp=c(1.5, 0.5, 0))
plot(gag.dat$age, c(GAGurine$GAG,NA,NA), pch=19,
     col=gray(.5), xlab='Age (years)',
     ylab='Childrens GAG concentration')
for (j in 1:5)
 lines(gag.dat$age, q.sam[j, ], lty=c(3,2,1,2,3)[j], lwd=2)
```



Visualize GAG limits

NTNU
Norwegian University of
Science and Technology

Outline

Overview

Tokyo example

GAG urine data

Close look at GAG urine example

R-INLA details

Results, post-processing and sampling

- Accessing results:
  - `summary(result)`
  - `plot(result)`
- post-processing
  - `result2 = inla.hyperpar(result)`
  - `result2 = inla.cpo(result)`
  - `result2 = inla.rerun(result)`
- sampling from posterior joint distribution
  - `samples = inla.hyperpar.sample(n=1000, result)`
  - `samples = inla.posterior.sample(n=1000, result)`

The interpretation of `NA`

`R-INLA` uses `NA` differently than other packages

- `NA` in the response means no likelihood contribution, i.e. response is unobserved
- `NA` in a fixed effect means no contribution to the linear predictor, i.e. the covariate is set equal to zero
- `NA` in a random effect `f(...)` means no contribution to the linear predictor

Changing the prior: Internal scale

— Hyperparameters are represented internally with more well-behaved transformations, e.g. correlation ρ and precision τ are internally represented as

$$\theta_1 = \log(\tau)$$

$$\theta_2 = \log\left(\frac{1+\rho}{1-\rho}\right)$$

— The prior must be set on the parameter in **internal scale**

— Initial values for the mode-search must be set in **internal scale**

— The functions `to.theta` and `from.theta` can be used to map back and forth.

Changing the prior: Code

```
## define the prior
hyper = list(prec = list(prior = "loggamma",
                         param = c(1, 0.1),
                         initial = 4,
                         fixed = FALSE))
## insert it into the formula
formula = y ~ f(idx, model = "iid", hyper = hyper) + ...
```

Changing the prior: Default options

```
## Default options can be seen with
inla.models()$latent$iid$hyper
```

```
theta
          name "log precision"
    short.name "prec"
         prior "loggamma"
         param c(1e+00, 5e-05)
       initial 4
         fixed FALSE
      to.theta function(x){log(x)}
    from.theta function(x){exp(x)}
```

Available priors

```
names(inla.models()$prior)

##  [1] "normal"                "gaussian"
##  [3] "wishart1d"             "wishart2d"
##  [5] "wishart3d"             "wishart4d"
##  [7] "wishart5d"             "loggamma"
##  [9] "minuslogsqrtruncnormal" "logtnormal"
## [11] "logtgaussian"          "flat"
## [13] "logflat"               "logiflat"
## [15] "mvnorm"                "pc.ar"
## [17] "none"                  "betacorrelation"
## [19] "logitbeta"             "pc.prec"
## [21] "pc.dof"                "pc.rho0"
## [23] "pc.rho1"               "pc.spde.GA"
## [25] "pc"                    "jeffreystdf"
## [27] "expression:"           "table:"
```

Available likelihoods

```
names(inla.models()$likelihood)
```

```
##  [1] "poisson"                    "gpoisson"
##  [3] "binomial"                   "testbinomial1"
##  [5] "gamma"                      "gammacount"
##  [7] "beta"                       "betabinomial"
##  [9] "cbinomial"                  "nbinomial"
## [11] "simplex"                    "gaussian"
## [13] "normal"                     "circularnormal"
## [15] "wrappedcauchy"              "iidgamma"
## [17] "iidlogitbeta"               "loggammafrailty"
## [19] "logistic"                   "skewnormal"
## [21] "sn"                         "gev"
## [23] "laplace"                    "lognormal"
## [25] "exponential"                "coxph"
## [27] "weibull"                    "loglogistic"
## [29] "weibullcure"                "stochvol"
## [31] "stochvolt"                  "stochvolnig"
## [33] "zeroinflatedpoisson0"       "zeroinflatedpoisson1"
## [35] "zeroinflatedpoisson2"       "zeroinflatedbetabinomial0"
## [37] "zeroinflatedbetabinomial1"  "zeroinflatedbetabinomial0"
## [39] "zeroinflatedbinomial1"      "zeroinflatedbinomial2"
## [41] "zeroinflatedbinomial2"      "zeroinflatedbetabinomial2"
```

Available latent models

```
names(inla.models()$latent)

##  [1] "linear"       "iid"          "mec"          "meb"
##  [5] "rgeneric"     "rw1"          "rw2"          "crw2"
##  [9] "seasonal"     "besag"        "besag2"       "bym"
## [13] "bym2"         "besagproper"  "besagproper2" "ar1"
## [17] "ar"           "ou"           "generic"      "generic0"
## [21] "generic1"     "generic2"     "generic3"     "spde"
## [25] "spde2"        "spde3"        "iid1d"        "iid2d"
## [29] "iid3d"        "iid4d"        "iid5d"        "2diid"
## [33] "z"            "rw2d"         "rw2diid"      "slm"
## [37] "matern2d"     "copy"         "clinear"      "sigm"
## [41] "revsigm"
```