PRACTICAL 1 - LINEAR (MIXED) MODELS

In this practical we are going to fit linear (mixed) models in `inlabru`. We are going to to:

- Fit a simple linear regression
- Fit a linear regression with discrete covariates and interactions
- Fit a linear mixed model.

In all cases we are first going to simulate our data. We then fit the data and explore the results.

Start by loading useful libraries:

```
library(dplyr)
library(INLA)
library(ggplot2)
library(patchwork)
library(inlabru)
# load some libraries to generate nice plots
library(scico)
```

1 Simple linear regression

2 Linear Mixed Model

In this practical we will:

- Understand the basic structure of a Linear Mixed Model (LLM)
- Simulate data from a LMM
- Learn how to fit a LMM with `inlabru` and predict from the model.

Consider the a simple linear regression model except with the addition that the data that comes in groups. Suppose that we want to include a random effect for each group j (equivalent to adding a group random intercept). The model is then:

$$y_{ij} = \beta_0 + \beta_1 x_i + u_j + \epsilon_{ij} \quad \text{for } i = 1, \dots, N \text{ and } j = 1, \dots, m.$$

Here the random group effect is given by the variable $u_j \sim \mathcal{N}(0, \tau_u^{-1})$ with $\tau_u = 1/\sigma_u^2$ describing the variability between groups (i.e., how much the group means differ from the overall mean). Then, $\epsilon_j \sim \mathcal{N}(0, \tau_\epsilon^{-1})$ denotes the residuals of the model and $\tau_\epsilon = 1/\sigma_\epsilon^2$ captures how much individual observations deviate from their group mean (i.e., variability within group).

The model design matrix for the random effect has one row for each observation (this is equivalent to a random intercept model). The row of the design matrix associated with the ij-th observation consists of zeros except for the element associated with u_j, which has a one.

$$\boldsymbol{\eta} = \boldsymbol{A}\boldsymbol{u} = \boldsymbol{A}_1\boldsymbol{u}_1 + \boldsymbol{A}_2\boldsymbol{u}_2 + \boldsymbol{A}_3\boldsymbol{u}_3$$

Supplementary material: LMM as a LGM

In matrix form, the linear mixed model for the *j*-th group can be written as:

$$\overset{N\times 1}{\hat{\mathbf{y}}_j} = \overset{N\times 2}{\widehat{X_j}} \underbrace{\beta}_{1\times 1} + \overset{n_j\times 1}{\widehat{Z_j}} \underbrace{u_j}_{1\times 1} + \overset{n_j\times 1}{\hat{\epsilon}_j} \ ,$$

In a latent Gaussian model (LGM) formulation the mixed model predictor for the *i*-th observation can be written as :

$$\eta_i = \beta_0 + \beta_1 x_i + \sum_k^K f_k(u_j)$$

where $f_k(u_j) = u_j$ since there's only one random effect per group (i.e., a random intercept for group j). The fixed effects (β_0, β_1) are assigned Gaussian priors (e.g., $\beta \sim \mathcal{N}(0, \tau_\beta^{-1})$). The random effects $\mathbf{u} = (u_1, \dots, u_m)^T$ follow a Gaussian density $\mathcal{N}(0, \mathbf{Q}_u^{-1})$ where $\mathbf{Q}_u = \tau_u \mathbf{I}_m$ is the precision matrix for the random intercepts. Then, the components for the LGM are the following:

- Latent field given by

$$\begin{bmatrix}\beta \\ \mathbf{u}\end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix}\tau_\beta^{-1}\mathbf{I}_2 & \mathbf{0} \\ \mathbf{0} & \tau_u^{-1}\mathbf{I}_m\end{bmatrix}\right)$$

- Likelihood:

$$y_i \sim \mathcal{N}(\eta_i, \tau_\epsilon^{-1})$$

- Hyperparameters:
  - $\tau_u \sim$ Gamma(a, b)
  - $\tau_\epsilon \sim$ Gamma(c, d)

2 Simulate example data

```
set.seed(12)
beta = c(1.5,1)
sd_error = 1
tau_group = 1

n = 100
n.groups = 5
x = rnorm(n)
v = rnorm(n.groups, sd = tau_group^{-1/2})
y = beta[1] + beta[2] * x + rnorm(n, sd = sd_error) +
  rep(v, each = 20)

df = data.frame(y = y, x = x, j = rep(1:5, each = 20))
```

Note that `inlabru` expects an integer indexing variable to label the groups.

```
ggplot(df) +
  geom_point(aes(x = x, colour = factor(j), y = y)) +
  theme_classic() +
  scale_colour_discrete("Group")
```

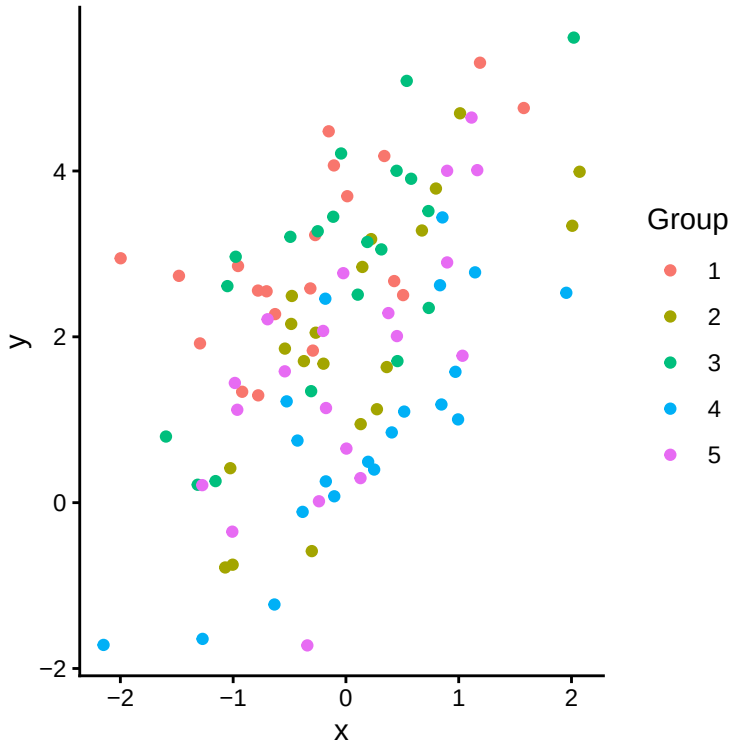


Figure 1: Data for the linear mixed model example with 5 groups

2 Fitting a LMM in `inlabru`

Defining model components and observational model

In order to specify this model we must use the `group` argument to tell `inlabru` which variable indexes the groups. The `model = "iid"` tells INLA that the groups are independent from one another.

```
# Define model components
cmp =  ~ -1 + beta_0(1) + beta_1(x, model = "linear") +
  u(j, model = "iid")
```

The group variable is indexed by column `j` in the dataset. We have chosen to name this component `v()` to connect with the mathematical notation that we used above.

```
# Construct likelihood
lik =  bru_obs(formula = y ~.,
               family = "gaussian",
               data = df)
```

Fitting the model

The model can be fitted exactly as in the previous examples by using the `bru` function with the components and likelihood objects.

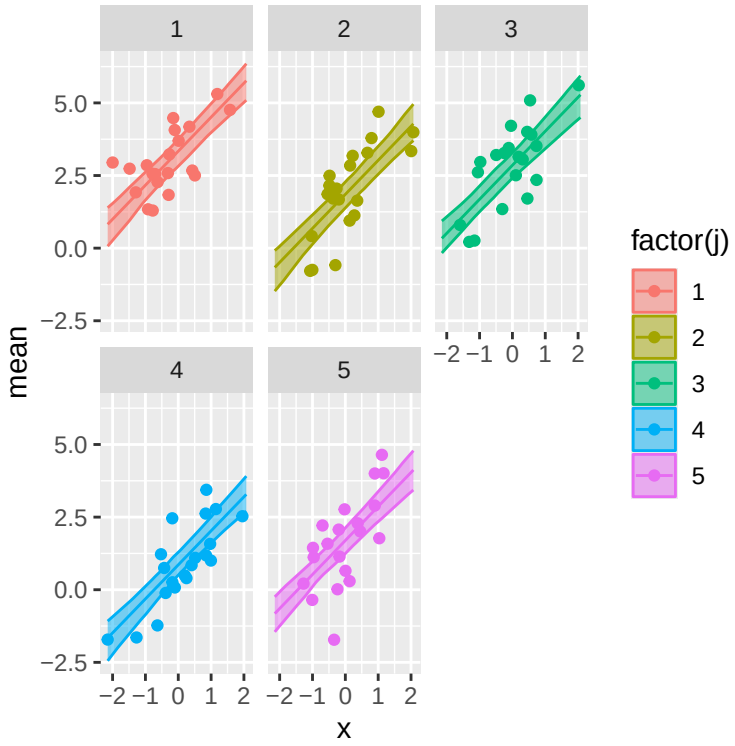
```
fit = bru(cmp, lik)
summary(fit)
## inlabru version: 2.13.0.9016
## INLA version: 25.11.22
## Components:
## Latent components:
## beta_0: main = linear(1)
## beta_1: main = linear(x)
## u: main = iid(j)
## Observation models:
##   Family: 'gaussian'
##     Tag: <No tag>
##     Data class: 'data.frame'
##     Response class: 'numeric'
##     Predictor: y ~ .
##     Additive/Linear: TRUE/TRUE
##     Used components: effects[beta_0, beta_1, u], latent[]
## Time used:
##     Pre = 1.11, Running = 0.22, Post = 0.0408, Total = 1.37
## Fixed effects:
##          mean    sd 0.025quant 0.5quant 0.975quant  mode kld
## beta_0 2.108 0.438      1.229    2.108      2.986 2.108   0
## beta_1 1.172 0.120      0.936    1.172      1.407 1.172   0
##
## Random effects:
##   Name     Model
##     u IID model
##
## Model hyperparameters:
##                                          mean    sd 0.025quant 0.5quant
## Precision for the Gaussian observations 0.995 0.144      0.738    0.986
## Precision for u                         1.613 1.060      0.369    1.356
##                                         0.975quant  mode
## Precision for the Gaussian observations       1.30 0.971
## Precision for u                               4.35 0.918
##
## Marginal log-Likelihood:  -179.93
##  is computed
## Posterior summaries for the linear predictor and the fitted values are computed
## (Posterior marginals needs also 'control.compute=list(return.marginals.predictor=TRUE)')
```

2 Model predictions

To compute model predictions we can create a `data.frame` containing a range of values of covariate where we want the response to be predicted for each group. Then we simply call the predict function while specifying the model components.

```
# New data
xpred = seq(range(x)[1], range(x)[2], length.out = 100)
j = 1:n.groups
pred_data = expand.grid(x = xpred, j = j)
pred = predict(fit, pred_data, formula = ~ beta_0 + beta_1 + u)


pred %>%
  ggplot(aes(x=x,y=mean,color=factor(j)))+
  geom_line()+
  geom_ribbon(aes(x,ymin = q0.025, ymax= q0.975,fill=factor(j)), alpha = 0.5) +
  geom_point(data=df,aes(x=x,y=y,colour=factor(j)))+
  facet_wrap(~j)
```



Question

Suppose that we are also interested in including random slopes into our model. Assuming intercept and slopes are independent, can your write down the linear predictor and the components of this model as a LGM?
Give me a hint
In general, the mixed model predictor can decomposed as:

$$\boldsymbol{\eta} = X\beta + Z\mathbf{u}$$

Where X is a $n \times p$ design matrix and β the corresponding p-dimensional vector of fixed effects. Then Z is a $n \times q_J$ design matrix for the q_J random effects and J groups; **v** is then a $q_J \times 1$ vector of q random effects for the J groups. In a latent Gaussian model (LGM) formulation this can be written as:

$$\eta_i = \beta_0 + \sum \beta_j x_{ij} + \sum_k f(k)(u_{ij})$$

See Solution

- The linear predictor is given by

$$\eta_i = \beta_0 + \beta_1 x_i + u_{0j} + u_{1j} x_i$$

- Latent field defined by:
  - $\beta \sim \mathcal{N}(0, \tau_\beta^{-1})$
  - $\mathbf{u}_j = \begin{bmatrix} u_{0j} \\ u_{1j} \end{bmatrix}, \mathbf{u}_j \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_u^{-1})$ where the precision matrix is a block-diagonal matrix with entries $\mathbf{Q}_u = \begin{bmatrix} \tau_{u_0} & 0 \\ 0 & \tau_{u_1} \end{bmatrix}$
- The hyperparameters are then:
  - τ_{u_0}, τ_{u_1} and τ_ϵ

To fit this model in `inlabru` we can simply modify the model components as follows:

```
cmp =  ~ -1 + beta_0(1) + beta_1(x, model = "linear") +
  u0(j, model = "iid") + u1(j,x, model = "iid")
```