Session 1: Introduction

Geir-Arne Fuglstad <fuglstad@math.ntnu.no>
Department of Mathematical Sciences, NTNU
April 14, 2016
Instructors

— Geir-Arne Fuglstad
  ● Postdoc with Håvard Rue
  ● Working with sensitivity and robustifying results from INLA
  ● PhD on non-stationary spatial modelling with SPDE-models

— Jingyi Guo
  ● Ph.D. student with Andrea Riebler and Håvard Rue
  ● Master in Statistics from Lund University
Practical information

Slides and code for the sessions available at http://www.math.ntnu.no/~fuglstad/Lund2016
Practical information

Information about the software, examples, papers and help can be found at http://www.r-inla.org
What is INLA?

We separate between three different parts:
1. The INLA method
2. The SPDE models
3. The **INLA** R-package
The INLA method

An approach for fast Bayesian inference with “latent Gaussian models”

Read paper:
The SPDE models

A novel way to get around the computational inefficiencies of continuously indexed spatial fields (GRFs)

Read paper:
Journal of the royal statistical society: Series B. 73, 319–392
The **INLA package**

The R-package is an implementation of the INLA method and the SPDE models with a flexible and simple interface

Download with:
```
source("http://www.math.ntnu.no/inla/givemeINLA-testing.R")
```
The history of **INLA**

— The development has been driven by Håvard Rue and is the result of many years of hard work

— Around 2002–2004 he and Leonard Held started to realize the importance of the class of models that INLA handles

— In 2005 Håvard Rue and Leonard Held wrote the book “Gaussian Markov Random Fields: Theory and Applications”
The history of INLA

— The first implementation in C was finished in 2007, but required hand-crafted input-files
— Arnoldo Frigessi (Oslo) suggested that an R-interface was necessary to reach a broad audience
— Sara Martino wrote the first prototype of the R-interface in January/February 2008
— The source code now consists of many, many, many lines...
— The source code is available at https://bitbucket.org/hrue/r-inla/
Who develops INLA?

Håvard Rue, Finn Lindgren, Daniel Simpson, Andrea Riebler, (Sara Martino, Thiago Guerrera Martins, Rupali Akerkar) and others (photo 2011)
Aims of the course

— Get an overview of latent Gaussian models
— Get an overview of the INLA method
— Learn how to use INLA for (generalized) linear models, and more
— Learn how to do spatial modelling with INLA
Structure of the course

Day 1:
10:30–11:45 Session 1: Introduction
13:15–15:00 Session 2: R-INLA
15:30–17:00 Session 3: Practical session with R-INLA

Day 2:
09:15–10:00 Session 4: Advanced Example
10:30–11:45 Session 5: Spatial modelling with INLA
13:15–15:00 Session 6: Practical session with spatial modelling
In general

— Ask questions!
— Discuss with us!
— If you have questions, you can use the google group or help@r-inla.org
Outline

Motivation

Bayesian hierarchical models

Latent Gaussian models

Deterministic inference

R and INLA
Why use INLA?

— Provides full Bayesian analysis
— Quick to write code, do not need to write a sampler
— Runs quickly
— Can be used for a flexible class of models (Latent Gaussian Models)
Example: Ski flying records

We have ski flying world records $y = (y_1, \ldots, y_n)$ and their dates $x_1, \ldots, x_n$, and want to fit a simple linear regression with Gaussian responses, where

$$E(y_i) = \mu + \beta x_i, \quad \text{Var}(y_i) = \tau^{-1}, \quad i = 1, \ldots, n$$
Frequentist analysis

```r
mod = lm(Length ~ Date, data = skiData)
summary(mod)
```

Estimates

\[ \mu: -3986 (66), \]
\[ \beta: 2.10 (0.03) \]
\[ \sigma = \frac{1}{\sqrt{\tau}}: 3.98 \]
Bayesian analysis

```r
res = inla(Length ~ Date, data = skiData)
res$summary.fixed[,1:2]; res$summary.hyperpar
```

- **mu**
  - Mean = -3986, SD = 65

- **beta**
  - Mean = 2.10, SD = 0.03

- **standard deviation**
  - Mean = 3.0, SD = 0.03

---

Real-world problems are typically more complicated!

Often we need to
  — include complicated dependency structures
  — stabilize the inference
Real-world problems are typically more complicated!

Often we need to
  — include complicated dependency structures
  — stabilize the inference

Can be achieved with hierarchical Bayesian modelling, but...

Two main challenges:
  — Need computationally efficient methods to calculate posteriors.
  — Select priors in a sensible way
Bayesian hierarchical models

\textbf{INLA} can analyze Bayesian hierarchical models specified in three stages:

\textbf{Stage 1:} What is the distribution of the responses?
Bayesian hierarchical models

INLA can analyze Bayesian hierarchical models specified in three stages:

**Stage 1:** What is the distribution of the responses?

**Stage 2:** What is the distribution of the underlying unobserved components?
Bayesian hierarchical models

**INLA** can analyze Bayesian hierarchical models specified in three stages:

**Stage 1:** What is the distribution of the responses?

**Stage 2:** What is the distribution of the underlying unobserved components?

**Stage 3:** What are our prior beliefs about the parameters controlling the components in the model?
Bayesian hierarchical models

INLA can analyze Bayesian hierarchical models specified in three stages:

**Stage 1:** What is the distribution of the responses?

**Stage 1.5:** How is the mean/variance/probability of the response linked to the underlying unobserved components?

**Stage 2:** What is the distribution of the underlying unobserved components?

**Stage 3:** What are our prior beliefs about the parameters controlling the components in the model?
Stage 1

How is the data \( y \) generated from the underlying components \( x \) and hyperparameters \( \theta \) in the model:

— Gaussian response?
— Count data? (E.g. Poisson, negative binomial)
— Zero-inflation?
— Point pattern? (E.g. Log-Gaussian cox process)
— Binary data?

The response distribution is connected to \( x \) and \( \theta \) through the likelihood \( \pi(y|x, \theta) \)
Stage 1.5

In INLA Stage 1 and Stage 2 must be connected through linear predictors by

\[ \pi(y|x, \theta) = \prod_{i=1}^{n} \pi(y_i|\eta_i, \theta), \]

where each \( \eta_i \) is a linear combination of the model components \( x \).

For example, \( \eta_i = \mu + \beta x_i \) can be combined with

- **Gaussian**: \( \eta_i = \mu_i \)
- **Poisson**: \( \eta_i = \log(\mu_i) \)
- **Binomial**: \( \eta_i = \logit(\rho_i) \)
Stage 2

The underlying **unobserved components** $x$ are called **latent components** and can be:

— Covariates
— Unstructured random effects (individual effects, group effects)
— Structured random effects (AR(1), regional effects, continuously indexed spatial effects)

The distribution of the **model components** are specified by $\pi(x|\theta)$
Stage 3

The likelihood and the latent model typically have hyperparameters that control their behavior. The hyperparameters $\theta$ can include:

- Variance of unstructured effects
- Range and variance of spatial effects
- Autocorrelation parameter
- Variance of observation noise
- Probability of a zero (zero-inflated models)

The a priori beliefs about these parameters are placed in the prior $\pi(\theta)$
Statistical jargon

It can be phrased with equations as

Stage 1:  \( y \mid x, \theta \sim \pi(y \mid x, \theta) = \prod_{i=1}^{n} \pi(y_i \mid \eta_i, \theta) \)

Stage 1.5: Each \( \eta_i \) is a linear combination of elements of \( x \)

Stage 2:  \( x \mid \theta \sim \pi(x \mid \theta) \)

Stage 3:  \( \theta \sim \pi(\theta) \)
Example: Disease mapping in Germany

We have observed larynx cancer mortality counts for males in 544 district of Germany from 1986 to 1990 and want to make a model.

Information given:

\( y_i \): The count at location \( i \).

\( E_i \): An offset; expected number of cases in district \( i \).

\( c_i \): A covariate (level of smoking consumption) at location \( i \).

\( s_i \): spatial location \( i \) (here, district).
Stage 1: The data

First we decide on the likelihood for our data $y$

- Need a distribution for counts
- We decide to model our responses as

$$y_i \mid \eta_i \sim \text{Poisson}(E_i \exp(\eta_i)),$$

$\eta_i$ is a linear function of the latent components
Stage 2: The latent model

We choose four components

— Intercept $\mu$
— Spatially structured effect $f_s$ and unstructured effect $u$
— Covariate effect $f(c_i)$ of the exposure covariate $c_i$

Combine with linear predictor $\eta_i = \mu + f_s(s_i) + f(c_i) + u_i$, and the full latent field $\mathbf{x} = (\mu, \{f_s(\cdot)\}, \{f(\cdot)\}, u_1, u_2, \ldots, u_n)$
Stage 3: Hyperparameters

The structured and unstructured spatial effect as well as the smooth covariate effect are each controlled by one parameter \( \tau_c, \tau_f, \tau_\eta \): The precisions (inverse variances) of the covariate effect, spatial effect and unstructured effect, respectively.

Hyperparameters \( \theta = (\tau_c, \tau_f, \tau_\eta) \) must be given a prior \( \pi(\tau_c, \tau_f, \tau_\eta) \).
Quantities of interest

Median of structured spatial effect
\[ \exp(f_s(s_i)) \]

Covariate effect \( f(c_i) \)
Latent Gaussian models

A key feature of the example is that it is contained in the very flexible and useful class of models called Latent Gaussian models.
Latent Gaussian models

A key feature of the example is that it is contained in the very flexible and useful class of models called Latent Gaussian models.

- The characteristic property is that the latent part of the hierarchical model is Gaussian, $x|\theta \sim \mathcal{N}(0, Q^{-1})$
  - The expected value is 0
  - The *precision* matrix (inverse covariance matrix) is $Q$
Latent Gaussian models

A key feature of the example is that it is contained in the very flexible and useful class of models called **Latent Gaussian models**

— The characteristic property is that the latent part of the hierarchical model is **Gaussian**, $\mathbf{x}|\theta \sim \mathcal{N}(0, \mathbf{Q}^{-1})$

  - The expected value is $\mathbf{0}$
  - The *precision* matrix (inverse covariance matrix) is $\mathbf{Q}$

Together with the linear predictor restriction this defines the class of models **INLA** can handle
The general set-up

The class contains GLMs, GLMMs, GAMs, GAMMs, and more. Can be constructed by connecting the mean $\mu_i$ to the linear predictor, $\eta_i$, through a link function $g$,

$$\eta_i = g(\mu_i) = \alpha + z_i^T \beta + u_i + \sum_{\gamma} w_{\gamma,i}, \quad i = 1, 2, \ldots, n$$

where
The general set-up

The class contains GLMs, GLMMs, GAMs, GAMMs, and more. Can be constructed by connecting the mean $\mu_i$ to the linear predictor, $\eta_i$, through a link function $g$,

$$\eta_i = g(\mu_i) = \alpha + \sum_{\gamma} w_{\gamma,i},$$

where

- $\alpha$ : Intercept
- $\beta$ : Fixed effects of covariates
- $u_i$ : Unstructured error terms
- $\{f_\gamma(c_\gamma,i)\}$ : Non-linear/smooth effects of covariates
- $\{w_{\gamma,i}\}$ : Known weights defined for each observed data point

$i = 1, 2, \ldots, n$
The general set-up

The class contains GLMs, GLMMs, GAMs, GAMMs, and more. Can be constructed by connecting the mean $\mu_i$ to the linear predictor, $\eta_i$, through a link function $g$,

$$
\eta_i = g(\mu_i) = \alpha + z_i^T \beta + u_i + \sum c_{w_i} \gamma(i), \quad i = 1, 2, \ldots, n
$$

where

- $\alpha$: Intercept
- $\beta$: Fixed effects of covariates $z$
The general set-up

The class contains GLMs, GLMMs, GAMs, GAMMs, and more. Can be constructed by connecting the mean $\mu_i$ to the linear predictor, $\eta_i$, through a link function $g$,

$$
\eta_i = g(\mu_i) = \alpha + z_i^T \beta + u_i + \sum \gamma_{\gamma, i}, \quad i = 1, 2, \ldots, n
$$

where

$\alpha$ : Intercept

$\beta$ : Fixed effects of covariates $z$

$u$ : Unstructured error terms
The general set-up

The class contains GLMs, GLMMs, GAMs, GAMMMs, and more. Can be constructed by connecting the mean \( \mu_i \) to the linear predictor, \( \eta_i \), through a link function \( g \),

\[
\eta_i = g(\mu_i) = \alpha + z_i^T \beta + u_i + \sum_{\gamma} f_{\gamma}(c_{\gamma,i}), \quad i = 1, 2, \ldots, n
\]

where

\( \alpha \) : Intercept
\( \beta \) : Fixed effects of covariates \( z \)
\( u \) : Unstructured error terms
\( \{f_{\gamma}(\cdot)\} \) : Non-linear/smooth effects of covariates \( c \)
The general set-up

The class contains GLMs, GLMMs, GAMs, GAMMs, and more. Can be constructed by connecting the mean $\mu_i$ to the linear predictor, $\eta_i$, through a link function $g$,

$$
\eta_i = g(\mu_i) = \alpha + z_i^T \beta + u_i + \sum_{\gamma} w_{\gamma,i} f_\gamma(c_{\gamma,i}), \quad i = 1, 2, \ldots, n
$$

where

- $\alpha$: Intercept
- $\beta$: Fixed effects of covariates $z$
- $u$: Unstructured error terms
- $\{f_\gamma(\cdot)\}$: Non-linear/smooth effects of covariates $c$
- $\{w_{\gamma,i}\}$: Known weights defined for each observed data point
Flexibility through $f$-functions

The functions $\{f_\gamma\}$ provides very different types of random effects
- $f(\text{time})$: E.g., an AR(1) process, RW1 or RW2
- $f(\text{spatial location})$: E.g., a Matérn field
- $f(\text{covariate})$: E.g., a RW1 or RW2 on the covariate values
- $f(\text{time}, \text{spatial location})$: spatio-temporal effect
- And much more
Additivity

— One of the most useful features of the framework is the additivity
— Effects can easily be removed and added without difficulty
— Each component might adds a new latent part and might add new hyperparameters, but the modelling framework and computations stay the same
Example: Smoothing binary time-series

— Observed the sequence $y_1, y_2, \ldots, y_n$ of 0s and 1s
— Each time $t$ has an associated covariate $x_i$
— We want to smooth the time series by inferring the sequence $p_t$, for $t = 1, 2, \ldots, n$, of probabilities for 1s at each time step
Example: Smoothing time series

Stage 1: Bernoulli distribution for the responses

\[ y_t | \eta_t \sim \text{Bernoulli} \left( \frac{\exp(\eta_t)}{1 + \exp(\eta_t)} \right) \]
Example: Smoothing time series

Stage 1: Bernoulli distribution for the responses

\[ y_t | \eta_t \sim \text{Bernoulli} \left( \frac{\exp(\eta_t)}{1 + \exp(\eta_t)} \right) \]

Stage 2: Covariates, AR(1) component and random noise are connected to likelihood by

\[ \eta_t = \beta_0 + \beta_1 x_t + a_t + v_t \]
Example: Smoothing time series

Stage 1: Bernoulli distribution for the responses

\[ y_t | \eta_t \sim \text{Bernoulli} \left( \frac{\exp(\eta_t)}{1 + \exp(\eta_t)} \right) \]

Stage 2: Covariates, AR(1) component and random noise are connected to likelihood by

\[ \eta_t = \beta_0 + \beta_1 x_t + a_t + v_t \]

Stage 3: \( \rho \): Dependence parameter in AR(1) process
\( \sigma_a^2 \): Marginal variance in AR(1) process
\( \sigma_v^2 \): Variance of the unstructured error
Loads of examples

— Dynamic linear models
— Stochastic volatility models
— Generalised linear (mixed) models
— Generalised additive (mixed) models
— Measurement error models
— Spline smoothing
— Semi-parametric regression
— Disease mapping
— Log-Gaussian Cox-processes
— Spatio-temporal models
— Survival analysis
— And more!
Compuations

Now we have a modelling framework.

But how are the calculations actually done?
Computations

Now we have a modelling framework.

But how are the calculations actually done?

It depends on what you want to compute!
What are we interested in?

— Quantiles for the fixed effects
What are we interested in?

— Quantiles for the fixed effects
— A linear combination of elements from the latent field (e.g. the average over an area of a spatial effect, the difference of two effects)
What are we interested in?

— Quantiles for the fixed effects
— A linear combination of elements from the latent field (e.g. the average over an area of a spatial effect, the difference of two effects)
— A single hyperparameter (e.g. the range)
What are we interested in?

— Quantiles for the fixed effects
— A linear combination of elements from the latent field (e.g. the average over an area of a spatial effect, the difference of two effects)
— A single hyperparameter (e.g. the range)
— A non-linear combination of hyperparameters (breeding values for livestock)
What are we interested in?

- Quantiles for the fixed effects
- A linear combination of elements from the latent field (e.g. the average over an area of a spatial effect, the difference of two effects)
- A single hyperparameter (e.g. the range)
- A non-linear combination of hyperparameters (breeding values for livestock)
- Predictions at unobserved locations
What do we need to compute?

Often interested in the marginal posteriors the latent field

\[ \pi(x_i|y) \]

or the marginal posteriors the hyperparameters

\[ \pi(\theta_i|y) \]

or the posterior of another statistics

\[ \pi(f(x, \theta)|y) \]
What do we need to compute?

Often interested in the marginal posteriors the latent field

\[ \pi(x_i|y) \]

or the marginal posteriors the hyperparameters

\[ \pi(\theta_i|y) \]

or the posterior of another statistics

\[ \pi(f(x, \theta)|y) \]

However, these can almost never be computed analytically.
Traditional approach with MCMC

Construct Markov chains with the target posterior distribution as the stationary distribution.

— Extensively used for Bayesian inference since the 1980’s
— It is flexible and general
— There are generic tools such as JAGS or OpenBUGS
— Or more specific tools for more specific models, e.g. BayesX
Alternative approach

— MCMC “works” for everything, but it can be incredibly slow
— Is it possible to make a quicker, more specialized inference scheme which only needs to work for this limited class of models?
Our model framework

Latent Gaussian models:

Stage 1: \( y|x, \theta \sim \prod_i \pi(y_i|\eta_i, \theta) \)

Stage 2: \( x|\theta \sim \pi(x|\theta) \sim \mathcal{N}(0, Q(\theta)^{-1}) \) Gaussian!

Stage 3: \( \theta \sim \pi(\theta) \)

where the precision matrix \( Q(\theta) \) is sparse. Generally these “sparse” Gaussian distributions are called Gaussian Markov random fields (GMRFs).

The sparseness can be exploited for very quick computations for the Gaussian part of the model through numerical algorithms for sparse matrices.
The INLA idea

Directly approximate the posterior marginals

\[ \pi(x_i \mid y) \quad \text{and} \quad \pi(\theta_j \mid y) \]

using the posterior distribution

\[ \pi(x, \theta \mid y) \propto \pi(\theta)\pi(x \mid \theta)\pi(y \mid x, \theta). \]
Toy example: Smoothing

Observations

\[ y_i = m(i) + \epsilon_i, \quad i = 1, \ldots, n \]

- \( \epsilon_i \) is i.i.d. Gaussian noise with *known* precision, \( \tau_0 \)
- \( m(i) \) is an *unknown* smooth function wrt \( i \)

---

```r
n = 50
idx = 1:n
fun = 100*((idx-n/2)/n)^3
y = fun + rnorm(n, mean =0, sd=1)
plot(idx, y)
```
Assumed hierarchical model

1. Data: Gaussian observations with known precision

\[ y_i \mid x_i, \theta \sim \mathcal{N}(x_i, \tau_0) \]
Assumed hierarchical model

1. **Data**: Gaussian observations with known precision

   \[ y_i \mid x_i, \theta \sim \mathcal{N}(x_i, \tau_0) \]

2. **Latent model**: A second-order random walk for the underlying smooth function\(^1\)

   \[
   \pi(x \mid \theta) \propto \theta^{(n-2)/2} \exp\left( -\frac{\theta}{2} \sum_{i=3}^{n} (x_i - 2x_{i-1} + x_{i-2})^2 \right)
   \]

\(^1\)model="rw2"
Assumed hierarchical model

1. **Data**: Gaussian observations with known precision
   \[ y_i \mid x_i, \theta \sim \mathcal{N}(x_i, \tau_0) \]

2. **Latent model**: A second-order random walk for the underlying smooth function\(^1\)
   \[
   \pi(x \mid \theta) \propto \theta^{(n-2)/2} \exp \left( -\frac{\theta}{2} \sum_{i=3}^{n} (x_i - 2x_{i-1} + x_{i-2})^2 \right)
   \]

3. **Hyperparameter**: The smoothing parameter \(\theta\) is assigned a \(\Gamma(a, b)\) prior
   \[
   \pi(\theta) \propto \theta^{a-1} \exp(-b\theta), \quad \theta > 0
   \]

\(^1\)model="rw2"
Derivation of posterior marginals (I)

Since

\[ \mathbf{x}, \mathbf{y} \mid \theta \sim \mathcal{N}(\cdot, \cdot) \]

(derived using \( \pi(\mathbf{x}, \mathbf{y} \mid \theta) \propto \pi(\mathbf{y} \mid \mathbf{x}, \theta) \pi(\mathbf{x} \mid \theta) \)),

we can compute (numerically) all marginals, using that
Derivation of posterior marginals (I)

Since

$$\mathbf{x}, \mathbf{y} \mid \theta \sim \mathcal{N}(\cdot, \cdot)$$

(derived using \(\pi(\mathbf{x}, \mathbf{y} \mid \theta) \propto \pi(\mathbf{y} \mid \mathbf{x}, \theta) \pi(\mathbf{x} \mid \theta))\),

we can compute (numerically) all marginals, using that

$$\pi(\theta \mid \mathbf{y}) \propto \frac{\pi(\mathbf{x}, \mathbf{y} \mid \theta) \pi(\theta)}{\pi(\mathbf{x} \mid \mathbf{y}, \theta)}$$
Posterior marginal for hyperparameter

Posterior marginal for theta

Unnormalised density

Posterior marginal for theta

Density
Derivation of posterior marginals (II)

From

\[ x \mid y, \theta \sim \mathcal{N}(\cdot, \cdot) \]
Derivation of posterior marginals (II)

From

\[
\mathbf{x} \mid \mathbf{y}, \theta \sim \mathcal{N}(\cdot, \cdot)
\]

we can compute

\[
\pi(x_i \mid \mathbf{y}) = \int \pi(x_i \mid \theta, \mathbf{y}) \pi(\theta \mid \mathbf{y}) \, d\theta
\]

\[
\approx \sum_k \pi(x_i \mid \theta_k, \mathbf{y}) \pi(\theta_k \mid \mathbf{y}) \Delta_k
\]

where \(\theta_k, k = 1, \ldots, K\), correspond to representative points of \(\theta \mid \mathbf{y}\) and \(\Delta_k\) are the corresponding weights.
Posterior marginal for latent parameters

Posterior marginal for $x_1$ for each theta (unweighted)
Posterior marginal for latent parameters

Posterior marginal for $x_1$ for each theta (weighted)
Posterior marginal for latent parameters

Posterior marginal for $x_1$
Fitted spline

The posterior marginals are used to calculate summary statistics, like means, variances and credible intervals:
Comparison with maximum likelihood

The red line is the Bayesian posterior, the blue line is the “posterior” using the MLE of $\theta$, and the vertical line is the observed value $y_1$.
Extensions

This is the simple basic idea behind INLA
Extensions

This is the simple basic idea behind INLA

However, we need to extend this basic idea to deal with
— More than one hyperparameter
— Non-Gaussian observations
Extension: More than one hyperparameter

**Step 1** Explore $\pi(\theta|y)$
- Locate the mode
- Use the Hessian to construct new variables
- Grid-search

**Step 2** Approximate marginals based on these integration points
Extension: More than one hyperparameter

**Step 1** Explore $\pi(\theta|y)$
- Locate the mode
- Use the Hessian to construct new variables
- Grid-search

**Step 2** Approximate marginals based on these integration points
Extension: More than one hyperparameter

**Step 1** Explore $\pi(\theta|y)$
- Locate the mode
- Use the Hessian to construct new variables
- Grid-search

**Step 2** Approximate marginals based on these integration points
Extension: More than one hyperparameter

Step 1  
Explore $\pi(\theta | y)$
- Locate the mode
- Use the Hessian to construct new variables
- Grid-search

Step 2  
Approximate marginals based on these integration points
Non-Gaussian observations

— $\pi(x|y, \theta)$ is often very close to a Gaussian distribution even with a non-Gaussian likelihood, and can be replaced with a Laplace approximation

— All the difficult high-dimensional integrals w.r.t. the latent field are easy, and only the integrals w.r.t. the hyperparameters remain

— The integrals can be done efficiently numerically when the number of hyperparameters is low
Limitations

— The dimension of the latent field $x$ can be large ($10^2$–$10^6$)
— But the dimension of the hyperparameters $\theta$ must be small ($\leq 9$)

In other words, each random effect can be big, but there cannot be too many random effects unless they share parameters.
How to use INLA?

INLA is implemented through the package \texttt{INLA} in the \texttt{R} software which
— is the most popular computing language in applied statistics
— is open source and \textit{free}
— has a lot of packages that extend the base functionality
— has a very user friendly \texttt{formula} interface

\begin{verbatim}
linear_model <- lm(weight ~ group)
\end{verbatim}

Fits the linear model

\[ \text{weight}_i = \mu + \text{group}_i + \epsilon_i \]
Summary of INLA

Three main ingredients in INLA
- Latent Gaussian models
- Laplace approximations
- Gaussian Markov random fields

These ingredients lead to a very nice tool for Bayesian inference:
- fast
- accurate
- scales well for moderate sizes