Session 2: Bayesian thinking and computation using INLA

Learning Objectives

After this lecture you should be able to

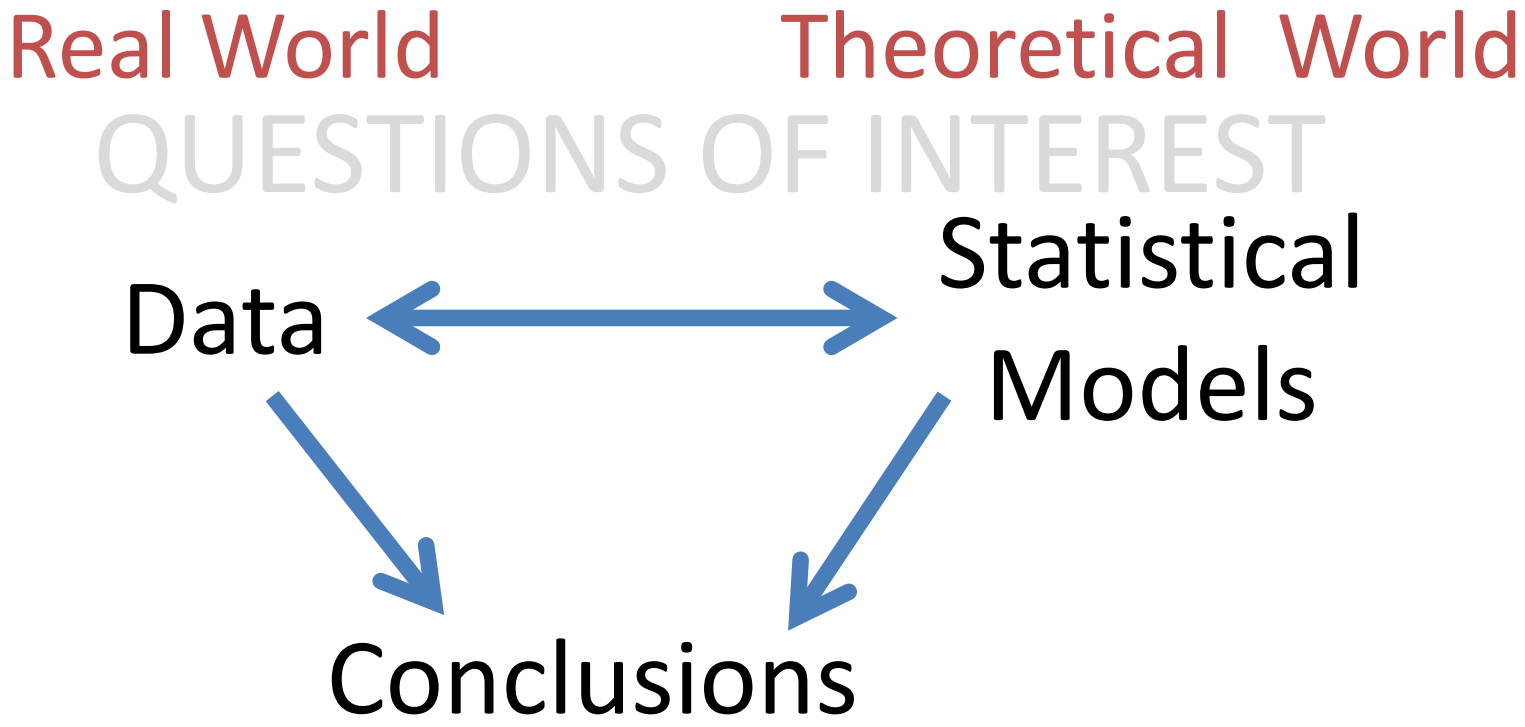
- Introduce Bayesian way of thinking
- Present Bayes theorem and Bayesian inference
- Describe computational methods commonly used to perform Bayesian inference
- Introduce INLA and the R package R-INLA

The topics treated in this lecture are presented in Chapter 3-4 of the "Spatial and Spatio-Temporal Bayesian Models with R-INLA"

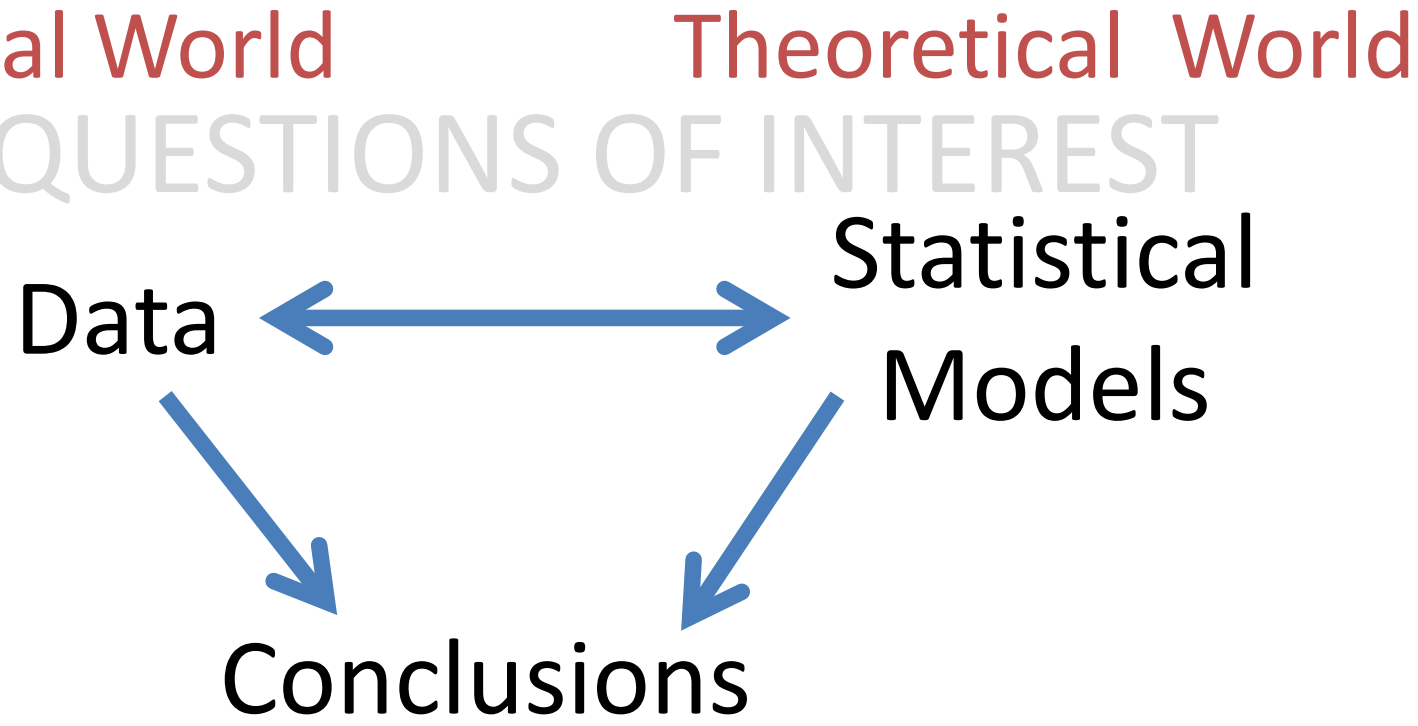
Outline

QUESTIONS OF INTEREST

Statistics - The 'Big Picture'



Statistics - The 'Big Picture'



- Several different ways of formulating statistical models and computing inferences from these models and data
- Can be grouped into two broad approaches:
  - Frequentist
  - Bayesian

Everyday thought process

- At the start of the football season I formed a view about the chance that the team I support will be relegated, based on
  - performance last season
  - summer transfers
  - etc.
- The first match is played
- I re-assesses the probability of relegation upwards, because
  - they lose
  - their main striker limps off

Everyday thought process

- At the start of the football season I formed a view about the chance that the team I support will be relegated, based on
  - performance last season
  - summer transfers
  - etc.

PRIOR view - before season begins

- The first match is played

- I re-assesses the probability of relegation upwards, because
  - they lose
  - their main striker limps off

Everyday thought process

- At the start of the football season I formed a view about the chance that the team I support will be relegated, based on
  - performance last season
  - summer transfers
  - etc.

PRIOR view - before season begins

- The first match is played

Now I have some information from current season (DATA)

- I re-assesses the probability of relegation upwards, because
  - they lose
  - their main striker limps off

Everyday thought process

- At the start of the football season I formed a view about the chance that the team I support will be relegated, based on
  - performance last season
  - summer transfers
  - etc.

PRIOR view - before season begins

- The first match is played

Now I have some information from current season (DATA)

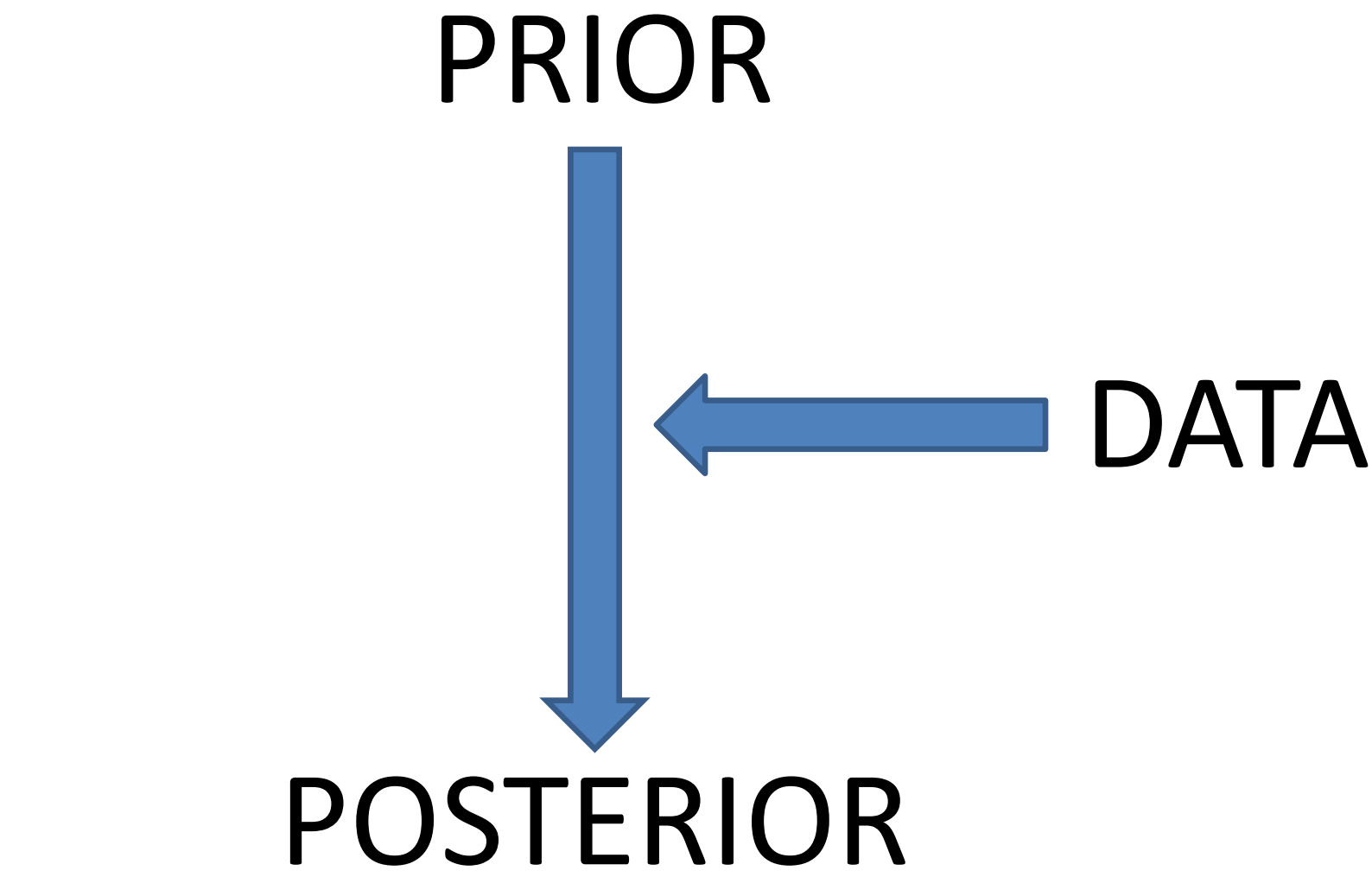
- I re-assesses the probability of relegation upwards, because
  - they lose
  - their main striker limps off

Prior view updated with data to give POSTERIOR view

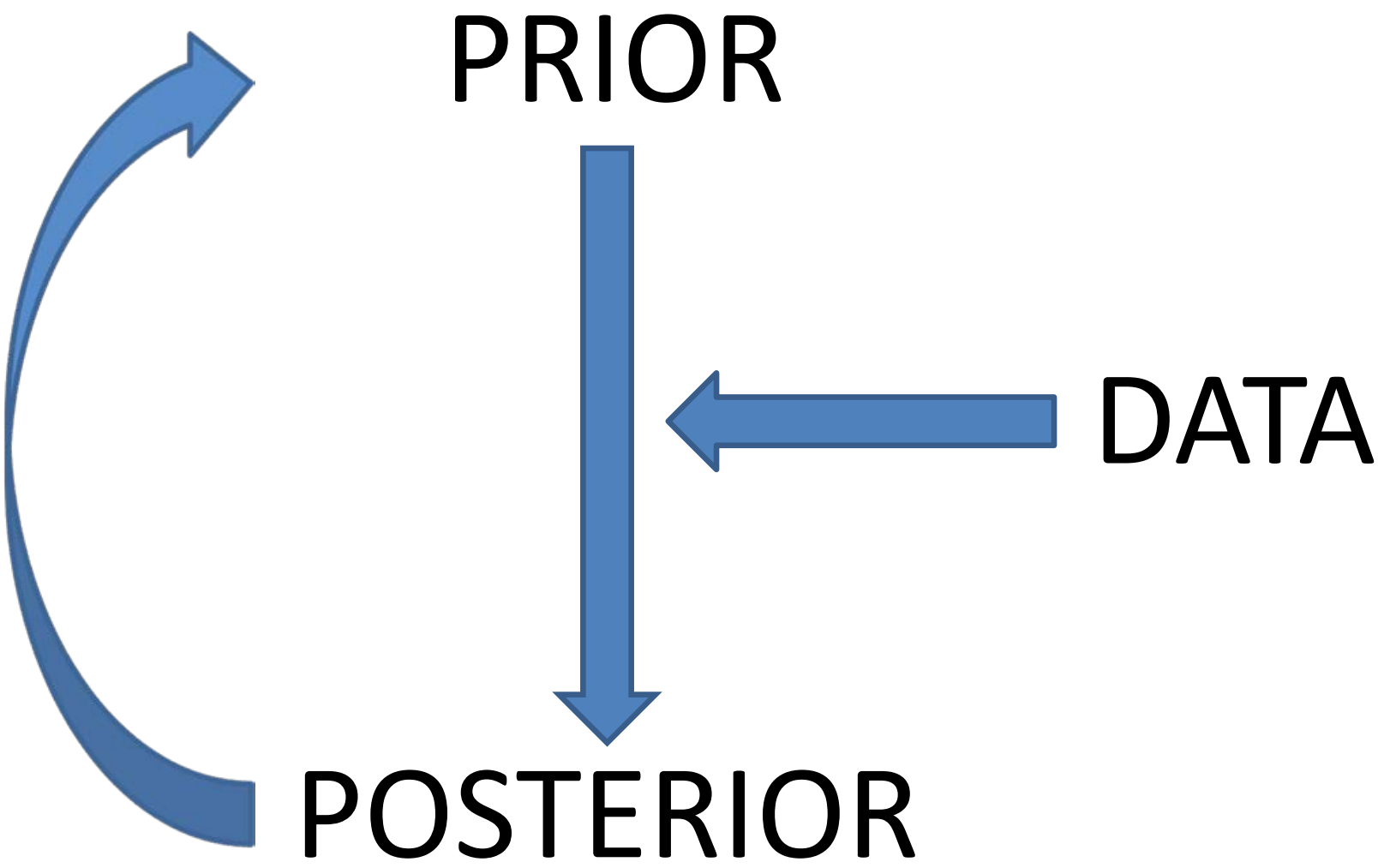
Thought process of a physician

- A patient presents with a set of symptoms, concerned that they might have a certain disease
- The physician assesses the chance that the patient has this disease, based on
  - symptoms
  - family history
  - alternative explanations of symptoms
  - prevalence of disease
- The physician sends the patient for a diagnostic test
- The physician re-assesses the chance that the patient has this disease, taking account of
  - results of diagnostic test
  - reliability of diagnostic test
- The physician may send the patient for further diagnostic tests

Bayesian thinking



Bayesian thinking



Why Bayesian methods?

- Bayesian methods have been widely applied in many areas:
  - medicine / epidemiology
  - genetics
  - ecology
  - environmental sciences
  - social and political sciences
  - finance
  - archaeology
  - .....
- Motivations for adopting Bayesian approach vary:
  - natural and coherent way of thinking about science and learning
  - pragmatic choice that is suitable for the problem in hand

Outline

Components of a Bayesian analysis

A clinical trial is carried out to collect evidence about an unknown 'treatment effect'. The Bayesian analyst needs to explicitly state

- a reasonable opinion concerning the plausibility of different values of the treatment effect *excluding* the evidence from the trial (the prior distribution)
- the support for different values of the treatment effect based *solely* on data from the trial (the likelihood),

and to combine these two sources to produce

- a final opinion about the treatment effect (the posterior distribution)

The final combination is done using Bayes theorem (and only simple rules of probability), which essentially weights the likelihood from the trial with the relative plausibilities defined by the prior distribution

One can view the Bayesian approach as a formalisation of the process of learning from experience

Bayesian inference: the posterior distribution

Posterior distribution forms basis for all inference — can be summarised to provide

- point and interval estimates of Quantities of Interest (QOI), e.g. treatment effect, small area estimates, ...
- point and interval estimates of any function of the parameters
- probability that QOI (e.g. treatment effect) exceeds a critical threshold
- prediction of QOI in a new unit
- prior information for future experiments, trials, surveys, ...
- inputs for decision making
- ...

Bayes theorem and its link with Bayesian inference

Bayes' theorem

- Provable from probability axioms
- Let A and B be events, then

$$p(A|B) = \frac{p(A \cap B)}{p(B)} = \frac{p(B|A)p(A)}{p(B)}$$

- If A_i is a set of mutually exclusive and exhaustive events (*i.e.* $A_i \cap A_j = \emptyset$, $p(\bigcup_i A_i) = \sum_i p(A_i) = 1$), then

$$p(A_i|B) = \frac{p(B|A_i)p(A_i)}{p(B)} = \frac{p(B|A_i)p(A_i)}{\sum_j p(B|A_j)p(A_j)}$$

Example: use of Bayes theorem in diagnostic testing

A new HIV test is claimed to have "95% sensitivity and 98% specificity".
In a population with an HIV prevalence of 1/1000, what is the chance that a patient testing positive actually has HIV?

- Let A be the event that patient is truly HIV positive, $\overline{A}$ be the event that they are truly HIV negative. We have that $p(A) = 0.001$ and $p(\overline{A}) = 0.999$.
- Let B be the event that they test positive:
  - "95% sensitivity" (the test is positive given that a person has HIV) means that $p(B|A) = .95$
  - "98% specificity" (the test is negative given that a person has not HIV) means that $p(B|\overline{A}) = .02$

- We want $p(A|B)$. Now Bayes theorem says

$$p(A|B) = \frac{p(B|A)p(A)}{p(B|A)p(A) + p(B|\overline{A})p(\overline{A})} = \frac{.95 \times .001}{.95 \times .001 + .02 \times .999} = .045$$

- Thus over 95% of those testing positive will, in fact, not have HIV.

Comments

- Our intuition is poor when processing probabilistic evidence
- The vital issue is *how should this test result change our belief that patient is HIV positive?*
- The disease prevalence can be thought of as a *'prior'* probability ($p = 0.001$)
- Observing a positive result causes us to modify this probability to $p = 0.045$. This is our *'posterior'* probability that patient is HIV positive.
- Bayes theorem applied to *observables* (as in diagnostic testing) is uncontroversial and established
- More controversial is the use of Bayes theorem in general statistical analyses, where *parameters* are the unknown quantities, and their prior distribution needs to be specified — this is Bayesian inference

Bayesian inference

Makes fundamental distinction between

- Observable quantities y, i.e. the data
- Unknown quantities θ
  - θ can be statistical parameters, missing data, mismeasured data ...

  $\rightarrow$ parameters are treated as random variables

  $\rightarrow$ in the Bayesian framework, we make probability statements about model parameters

! in the frequentist framework, parameters are fixed non-random quantities and the probability statements concern the data

As with any statistical analysis, we start building a model which specifies $p(y \mid \theta)$

This is the likelihood, which relates all variables into a 'full probability model'

Bayesian inference [continued]

From a Bayesian point of view

- θ is unknown so should have a probability distribution reflecting our uncertainty about it before seeing the data

  $\rightarrow$ need to specify a prior distribution $p(\theta)$

- y is known so we should condition on it

  $\rightarrow$ use Bayes theorem to obtain conditional probability distribution for unobserved quantities of interest given the data:

$$p(\theta \mid y) = \frac{p(\theta, y)}{p(y)} = \frac{p(\theta)\, p(y \mid \theta)}{\int p(\theta)\, p(y \mid \theta)\, d\theta} \propto p(\theta)\, p(y \mid \theta)$$

  This is the posterior distribution

The prior distribution $p(\theta)$, expresses our uncertainty about θ before seeing the data

The posterior distribution $p(\theta \mid y)$, expresses our uncertainty about θ after seeing the data

Outline

How to obtain the posterior distribution?

- When the prior and posterior come from the same family of distributions the prior is said to be **conjugate** to the likelihood $\rightarrow$ the posterior is a known distribution.
- In real life it is (almost) impossible to use conjugacy... we need to resort to simulative approaches or approximations:
  - Monte Carlo methods
  - Markov Chain Monte Carlo
  - INLA

Monte Carlo Simulation

- This approach is based on the idea that if you have a large random sample from a certain distribution, the statistics that you can calculate in this sample (mean, SD, percentiles...) will be very similar to the corresponding theoretical values in the distribution.

- If you have a complicated mathematical expression for a distribution and you cannot calculate algebraically important parameters, you could get the computer to generate a large random sample from such a distribution

- By calculating the mean of that parameter in the sample you could estimate the mean in the original distribution with great precision

Monte Carlo approach to approximate log-odds

- We start with a Binomial likelihood

$$y \mid \theta \sim \text{Binomial}(\theta, n)$$

combined with a

$$\text{Beta}(a, b)$$

as prior for the probability of success θ.

- We are interested in the log-odds function of θ defined as

$$\log\left(\frac{\theta}{1-\theta}\right)$$

- The integral

$$\int_0^1 \log\left(\frac{\theta}{1-\theta}\right) p(\theta \mid y) \mathrm{d}\theta$$

cannot be computed analytically; we resort to Monte Carlo approximation

Example of MC: in practice

- We simulate m independent values $\left\{\theta^{(1)}, \ldots, \theta^{(m)}\right\}$ from the Beta$(a_1 = y + a, b_1 = n - y + b)$ posterior distribution.
- We apply the log-odds transformation to each value obtaining the set of values

$$\left\{\log\left(\frac{\theta^{(1)}}{1-\theta^{(1)}}\right), \ldots, \log\left(\frac{\theta^{(m)}}{1-\theta^{(m)}}\right)\right\}$$

- Finally, we compute the sample mean

$$\frac{\sum_{i=1}^{m} \log\left(\frac{\theta^{(i)}}{1-\theta^{(i)}}\right)}{m}$$

which is the Monte Carlo approximation to

$$\log\left(\frac{\theta}{1-\theta}\right)$$

Example of MC: R code

In R:

```
a <- 1
b <- 1
theta <- rbeta(1,a,b)
n <- 1000
y <- rbinom(1, size=n, p=theta)
a1 <- a + y
b1 <- n - y + b
```

Example of MC: R code

- With this setting the exact posterior distribution of θ is

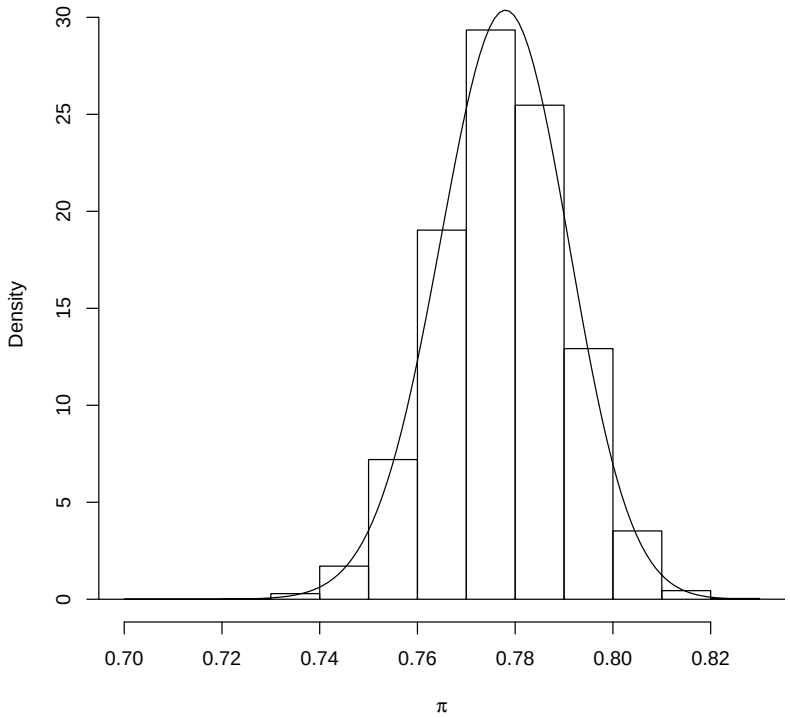
$$\text{Beta}(a_1 = a + y, b_1 = n - y + b)$$

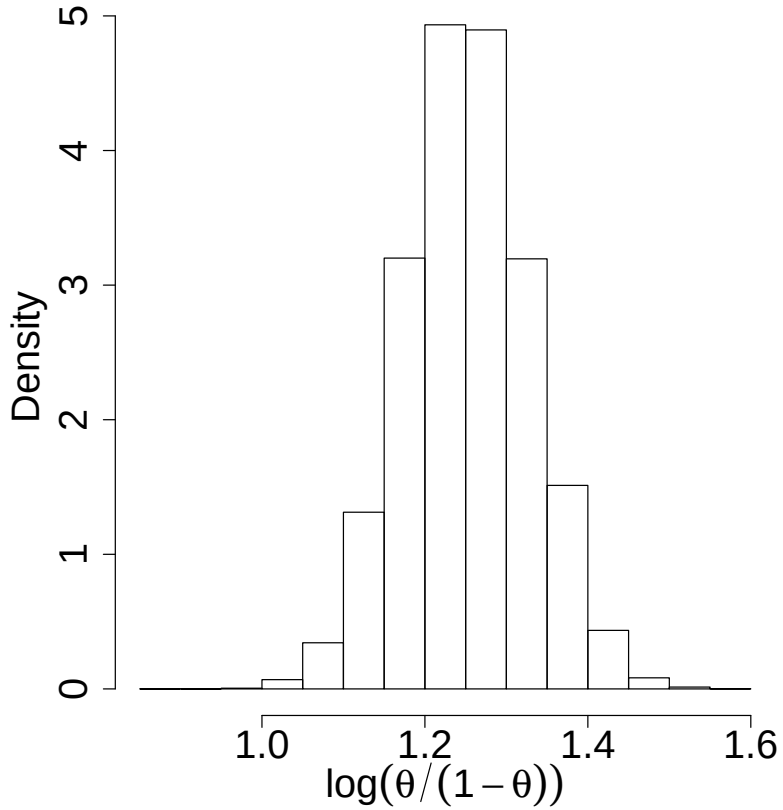
- To approximate the log-odds, we simulate $m = 50000$ values from this Beta posterior distribution using the `rbeta` function.

```
sim <- rbeta(n=50000, shape1=a1, shape2=b1)
logodds <- log(sim/(1-sim))
```

Results and comparison with the theoretical distribution

The empirical distribution of the Monte Carlo sample is plotted below together with the exact posterior distribution of θ.





Why Markov Chain Monte Carlo?

- For all but trivial examples it will be difficult to draw an iid Monte Carlo sample directly from the posterior distribution.
  - This happens, for example, when the dimension of the parameter vector $\boldsymbol{\theta}$ is high
  - Also to use MC methods we must have a known form for the posterior distribution
- Alternatively *correlated* values (via MCMC) can be (more easily) drawn to approximate the posterior distribution of the parameter of interest
- Instead of simulating independent values from the posterior distribution we draw a sample by running a Markov chain whose stationary distribution is the posterior density $p(\theta \mid y)$
- A sequence of values $\left\{\theta^{(1)}, \ldots, \theta^{(m)}\right\}$ generated from a Markov chain that has reached its stationary distribution (i.e. has converged) can be considered as an approximation to the posterior distribution and can be used to compute all the summaries of interest.

MCMC — Gibbs sampling

The **Gibbs sampling** (GS) is one of the most popular schemes for MCMC. Consider the case of a generic J dimensional parameter set $(\theta_1, \theta_2, \ldots, \theta_J)$:

1. Select a set of initial values $(\theta_1^{(0)}, \theta_2^{(0)}, \ldots, \theta_J^{(0)})$
2. Sample $\theta_1^{(1)}$ from the conditional distribution $p(\theta_1 \mid \theta_2^{(0)}, \theta_3^{(0)}, \ldots, \theta_J^{(0)}, y)$
   Sample $\theta_2^{(1)}$ from the conditional distribution $p(\theta_2 \mid \theta_1^{(1)}, \theta_3^{(0)}, \ldots, \theta_J^{(0)}, y)$
   ...
   Sample $\theta_J^{(1)}$ from the conditional distribution $p(\theta_J \mid \theta_1^{(1)}, \theta_2^{(1)}, \ldots, \theta_{J-1}^{(1)}, y)$

MCMC — Gibbs sampling

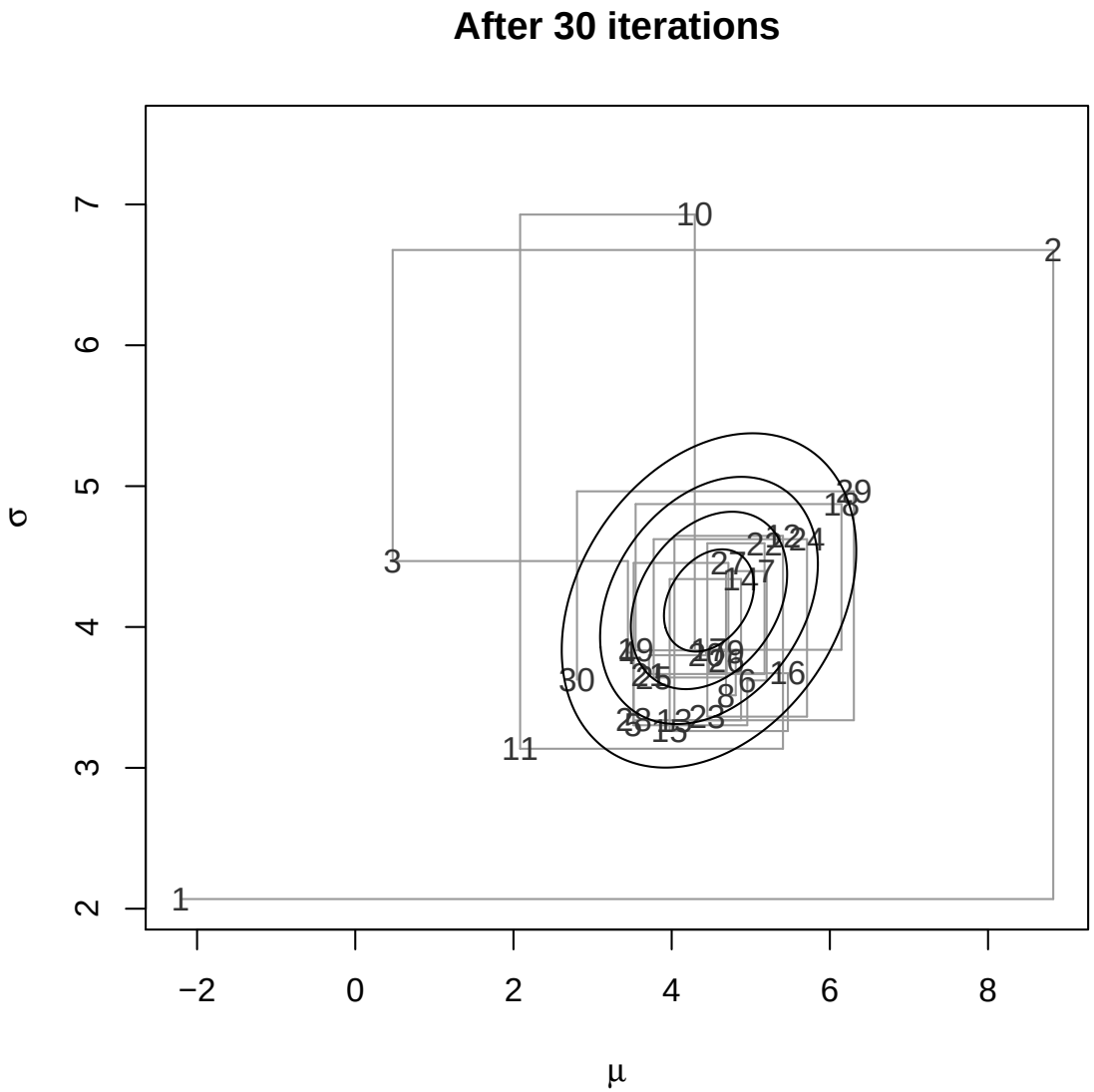
The **Gibbs sampling** (GS) is one of the most popular schemes for MCMC. Consider the case of a generic J dimensional parameter set $(\theta_1, \theta_2, \ldots, \theta_J)$:

1. Select a set of initial values $(\theta_1^{(0)}, \theta_2^{(0)}, \ldots, \theta_J^{(0)})$
2. Sample $\theta_1^{(1)}$ from the conditional distribution $p(\theta_1 \mid \theta_2^{(0)}, \theta_3^{(0)}, \ldots, \theta_J^{(0)}, y)$
   Sample $\theta_2^{(1)}$ from the conditional distribution $p(\theta_2 \mid \theta_1^{(1)}, \theta_3^{(0)}, \ldots, \theta_J^{(0)}, y)$
   ...
   Sample $\theta_J^{(1)}$ from the conditional distribution $p(\theta_J \mid \theta_1^{(1)}, \theta_2^{(1)}, \ldots, \theta_{J-1}^{(1)}, y)$
3. Repeat step 2. for S times until convergence is reached to the target distribution $p(\boldsymbol{\theta} \mid y)$
4. Use the sample from the target distribution to compute all relevant statistics: (posterior) mean, variance, credibility intervals, etc.

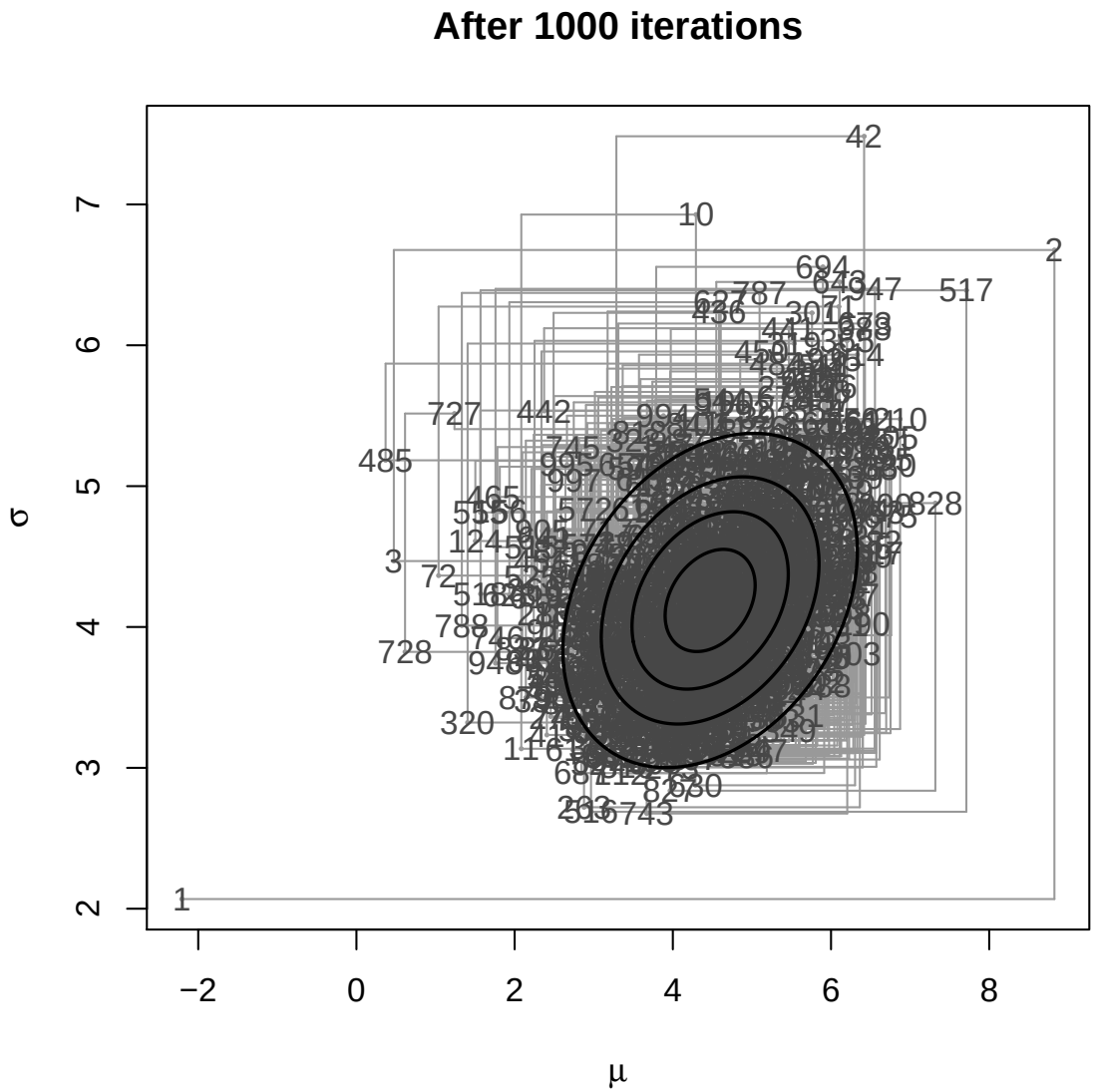
If the *full conditionals* are not readily available, they need to be estimated (eg via Metropolis-Hastings) before applying the GS

MCMC — convergence

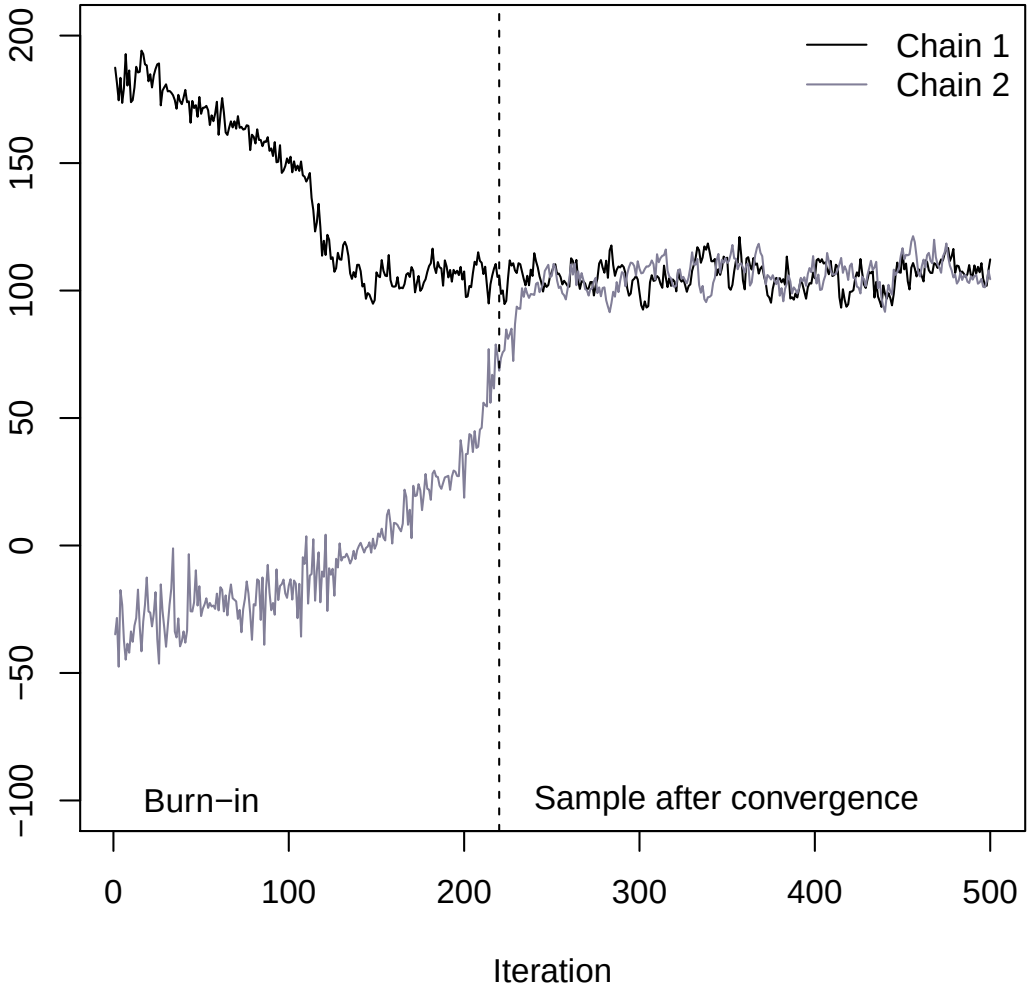
MCMC — convergence



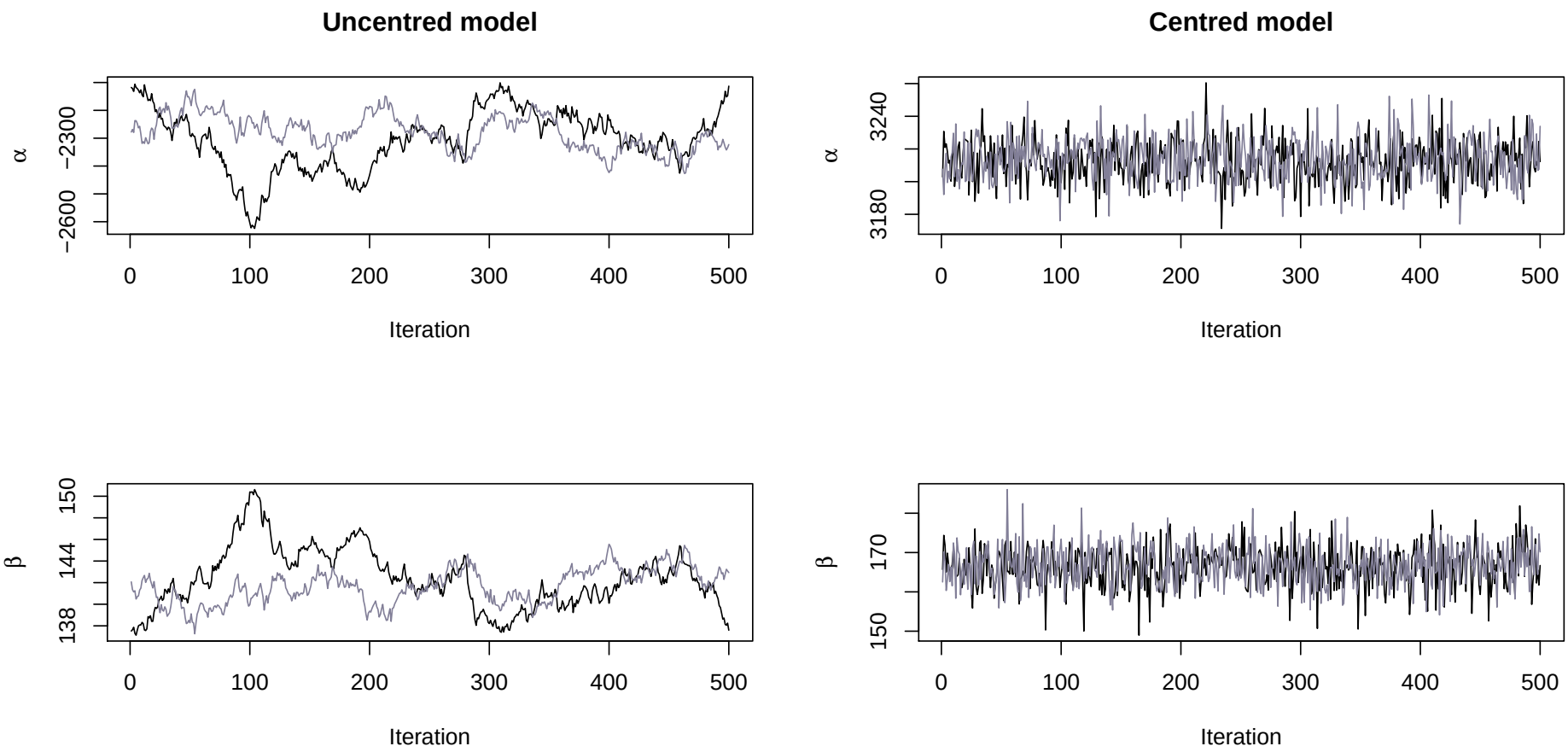
MCMC — convergence



MCMC — convergence



MCMC — convergence



MCMC — pros & cons

- "Standard" MCMC sampler are generally easy-ish to program and are in fact implemented in readily available software
- MCMC methods are flexible and able to deal with virtually any type of data and model, but they involve computationally- and time- intensive simulations to obtain the posterior distribution for the parameters. For this reason the complexity of the model and the database dimension often remain fundamental issues.

MCMC — pros & cons

- "Standard" MCMC sampler are generally easy-ish to program and are in fact implemented in readily available software
- MCMC methods are flexible and able to deal with virtually any type of data and model, but they involve computationally- and time- intensive simulations to obtain the posterior distribution for the parameters. For this reason the complexity of the model and the database dimension often remain fundamental issues.
- The INLA algorithm proposed by *Rue et al. (2009)* is a *deterministic* algorithm for Bayesian inference and it represents an alternative to MCMC which is instead a simulation based algorithm.

MCMC — pros & cons

- "Standard" MCMC sampler are generally easy-ish to program and are in fact implemented in readily available software
- MCMC methods are flexible and able to deal with virtually any type of data and model, but they involve computationally- and time- intensive simulations to obtain the posterior distribution for the parameters. For this reason the complexity of the model and the database dimension often remain fundamental issues.
- The INLA algorithm proposed by *Rue et al. (2009)* is a *deterministic* algorithm for Bayesian inference and it represents an alternative to MCMC which is instead a simulation based algorithm.
- The INLA algorithm is designed for the class of *latent Gaussian models* and compared to MCMC it provides (as) accurate results in a shorter time.

MCMC — pros & cons

- "Standard" MCMC sampler are generally easy-ish to program and are in fact implemented in readily available software
- MCMC methods are flexible and able to deal with virtually any type of data and model, but they involve computationally- and time- intensive simulations to obtain the posterior distribution for the parameters. For this reason the complexity of the model and the database dimension often remain fundamental issues.
- The INLA algorithm proposed by *Rue et al. (2009)* is a *deterministic* algorithm for Bayesian inference and it represents an alternative to MCMC which is instead a simulation based algorithm.
- The INLA algorithm is designed for the class of *latent Gaussian models* and compared to MCMC it provides (as) accurate results in a shorter time.
- INLA has become very popular amongst statisticians and applied researchers and in the past few years the number of papers reporting usage and extensions of the INLA method has increased considerably.

Outline

INLA website and community

http://www.r-inla.org/

- The website contains source code, examples, papers and reports discussing the theory and applications of INLA.
- There is also a discussion forum where users can post queries and requests of help.
- There are INLA-related scientific meetings (see http://www.r-inla.org).

Latent Gaussian models (LGMs)

Latent Gaussian models (LGMs)

- The general problem of (parametric) inference is posited by assuming a probability model for the observed data $\boldsymbol{y} = (y_1, \ldots, y_n)$, as a function of some relevant parameters

$$\boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{\psi} \sim p(\boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{\psi}) = \prod_{i=1}^{n} p(y_i \mid \boldsymbol{\theta}, \boldsymbol{\psi})$$

Latent Gaussian models (LGMs)

- The general problem of (parametric) inference is posited by assuming a probability model for the observed data $\boldsymbol{y} = (y_1, \ldots, y_n)$, as a function of some relevant parameters

$$\boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{\psi} \sim p(\boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{\psi}) = \prod_{i=1}^{n} p(y_i \mid \boldsymbol{\theta}, \boldsymbol{\psi})$$

- Often (in fact for a surprisingly large range of models!), we can assume that the parameters are described by a **Gaussian Markov Random Field** (GMRF)

$$\boldsymbol{\theta} \mid \boldsymbol{\psi} \sim \text{Normal}(\mathbf{0}, \boldsymbol{Q}^{-1}(\boldsymbol{\psi}))$$
$$\theta_i \perp\!\!\!\perp \theta_j \mid \boldsymbol{\theta}_{-i,j} \iff Q_{ij}(\boldsymbol{\psi}) = 0$$

  where
  - The precision matrix $\boldsymbol{Q}$ depends on some hyperparameters $\boldsymbol{\psi}$.
  - The notation "$-i, j$" indicates all the other elements of the parameters vector, excluding elements i and j
  - The components of $\boldsymbol{\theta}$ are supposed to be *conditionally independent* with the consequence that $\boldsymbol{Q}$ is a sparse precision matrix.

LGMs as a general framework

- In general

$$
\begin{aligned}
\boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{\psi} &\sim p(\boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{\psi}) = \prod_i p(y_i \mid \boldsymbol{\theta}, \boldsymbol{\psi}) && (\text{“data model”}) \\
\boldsymbol{\theta} \mid \boldsymbol{\psi} &\sim p(\boldsymbol{\theta} \mid \boldsymbol{\psi}) = \text{Normal}(0, \boldsymbol{Q}^{-1}(\boldsymbol{\psi})) && (\text{“GMRF prior”}) \\
\boldsymbol{\psi} &\sim p(\boldsymbol{\psi}) && (\text{“hyperprior”})
\end{aligned}
$$

LGMs as a general framework

- In general

$$
\begin{aligned}
\boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{\psi} &\sim p(\boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{\psi}) = \prod_i p(y_i \mid \boldsymbol{\theta}, \boldsymbol{\psi}) && (\text{“data model”}) \\
\boldsymbol{\theta} \mid \boldsymbol{\psi} &\sim p(\boldsymbol{\theta} \mid \boldsymbol{\psi}) = \text{Normal}(0, \boldsymbol{Q}^{-1}(\boldsymbol{\psi})) && (\text{“GMRF prior”}) \\
\boldsymbol{\psi} &\sim p(\boldsymbol{\psi}) && (\text{“hyperprior”})
\end{aligned}
$$

- The dimension of $\boldsymbol{\theta}$ can be very large (eg 10^2-10^5).
- Conversely, the dimension of $\boldsymbol{\psi}$ must be relatively small (less than 20 is recommended) to avoid an exponential increase in the computational costs of the model.

LGMs as a general framework

- A very general way of specifying the problem is specifying a distribution for y_i characterized by a parameter ϕ_i (usually the mean) defined as a function of a structured additive predictor η_i , defined on a suitable scale, such that $g(\phi_i) = \eta_i$ (e.g. logistic for binomial data):

$$\eta_i = \beta_0 + \sum_{m=1}^{M} \beta_m x_{mi} + \sum_{l=1}^{L} f_l(z_{li})$$

where
  - β_0 is the intercept;
  - $\boldsymbol{\beta} = \{\beta_1, \ldots, \beta_M\}$ quantify the effect of the covariates $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M)$ on the response;
  - $\boldsymbol{f} = \{f_1(\cdot), \ldots, f_L(\cdot)\}$ is a set of functions defined in terms of some covariates $\boldsymbol{z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_L)$

and then assume

$$\boldsymbol{\theta} = \{\beta_0, \boldsymbol{\beta}, \boldsymbol{f}\} \sim \text{Normal}(\boldsymbol{0}, \boldsymbol{Q}^{-1}(\boldsymbol{\psi})) = \text{GMRF}(\boldsymbol{\psi})$$

LGMs as a general framework

- A very general way of specifying the problem is specifying a distribution for y_i characterized by a parameter ϕ_i (usually the mean) defined as a function of a structured additive predictor η_i , defined on a suitable scale, such that $g(\phi_i) = \eta_i$ (e.g. logistic for binomial data):

$$\eta_i = \beta_0 + \sum_{m=1}^{M} \beta_m x_{mi} + \sum_{l=1}^{L} f_l(z_{li})$$

where
  - β_0 is the intercept;
  - $\boldsymbol{\beta} = \{\beta_1, \ldots, \beta_M\}$ quantify the effect of the covariates $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M)$ on the response;
  - $\boldsymbol{f} = \{f_1(\cdot), \ldots, f_L(\cdot)\}$ is a set of functions defined in terms of some covariates $\boldsymbol{z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_L)$

and then assume

$$\boldsymbol{\theta} = \{\beta_0, \boldsymbol{\beta}, \boldsymbol{f}\} \sim \text{Normal}(\boldsymbol{0}, \boldsymbol{Q}^{-1}(\boldsymbol{\psi})) = \text{GMRF}(\boldsymbol{\psi})$$

- **NB**: This of course implies some form of Normally-distributed marginals for $\beta_0, \boldsymbol{\beta}$ and $\boldsymbol{f}$

LGMs as a general framework — examples

Upon varying the form of the functions $f_l(\cdot)$, this formulation can accommodate a wide range of models (see Martins et al. (2012) for a review)

LGMs as a general framework — examples

Upon varying the form of the functions $f_l(\cdot)$, this formulation can accommodate a wide range of models (see Martins et al. (2012) for a review)

- Standard regression
  - $f_l(\cdot) = \text{NULL}$

LGMs as a general framework — examples

Upon varying the form of the functions $f_l(\cdot)$, this formulation can accommodate a wide range of models (see Martins et al. (2012) for a review)

- Standard regression
  - $f_l(\cdot) = \text{NULL}$
- Hierarchical models
  - $f_l(\cdot) \sim \text{Normal}(0, \sigma_f^2)$
    $\sigma_f^2 \mid \boldsymbol{\psi} \sim$ some common distribution

LGMs as a general framework — examples

Upon varying the form of the functions $f_l(\cdot)$, this formulation can accommodate a wide range of models (see Martins et al. (2012) for a review)

- Standard regression
  - $f_l(\cdot) = \text{NULL}$
- Hierarchical models
  - $f_l(\cdot) \sim \text{Normal}(0, \sigma_f^2)$
    $\sigma_f^2 \mid \boldsymbol{\psi} \sim$ some common distribution
- Spatial and spatio-temporal models
  - Areal data: $f_1(\cdot) \sim \text{CAR}$ (Spatially structured effects)
    (Unstructured residual)
  - Geostatistical data: $f(\cdot) \sim$ Gaussian field
  - Temporal component: $f(\cdot) \sim \text{RW}$

LGMs as a general framework — examples

Upon varying the form of the functions $f_l(\cdot)$, this formulation can accommodate a wide range of models (see Martins et al. (2012) for a review)

- Standard regression
  - $f_l(\cdot) = \text{NULL}$
- Hierarchical models
  - $f_l(\cdot) \sim \text{Normal}(0, \sigma_f^2)$
    $\sigma_f^2 \mid \boldsymbol{\psi} \sim$ some common distribution
- Spatial and spatio-temporal models
  - Areal data: $f_1(\cdot) \sim \text{CAR}$ (Spatially structured effects)
    (Unstructured residual)
  - Geostatistical data: $f(\cdot) \sim$ Gaussian field
  - Temporal component: $f(\cdot) \sim \text{RW}$
- Spline smoothing
  - $f_l(\cdot) \sim \text{AR}(\phi, \sigma_\varepsilon^2)$

LGMs as a general framework — examples

Upon varying the form of the functions $f_l(\cdot)$, this formulation can accommodate a wide range of models (see Martins et al. (2012) for a review)

- Standard regression
  - $f_l(\cdot) = \text{NULL}$
- Hierarchical models
  - $f_l(\cdot) \sim \text{Normal}(0, \sigma_f^2)$
    $\sigma_f^2 \mid \boldsymbol{\psi} \sim$ some common distribution
- Spatial and spatio-temporal models
  - Areal data: $f_1(\cdot) \sim \text{CAR}$ (Spatially structured effects)
    (Unstructured residual)
  - Geostatistical data: $f(\cdot) \sim$ Gaussian field
  - Temporal component: $f(\cdot) \sim \text{RW}$
- Spline smoothing
  - $f_l(\cdot) \sim \text{AR}(\phi, \sigma_\varepsilon^2)$
- Survival models, logGaussian Cox Processes, etc.

The INLA approach

INLA ingredients

- The first "ingredient" of the INLA approach is the definition of conditional probability, which holds for any pair of variables (x, z) — and, technically, provided $p(z) > 0$

$$p(x \mid z) =: \frac{p(x, z)}{p(z)} \rightarrow p(x, z) = p(x \mid z)p(z)$$

$p(x \mid z)$ can be re-written as

$$p(z) = \frac{p(x, z)}{p(x \mid z)}$$

INLA ingredients

- The first "ingredient" of the INLA approach is the definition of conditional probability, which holds for any pair of variables (x, z) — and, technically, provided $p(z) > 0$

$$p(x \mid z) =: \frac{p(x, z)}{p(z)} \rightarrow p(x, z) = p(x \mid z)p(z)$$

  $p(x \mid z)$ can be re-written as

$$p(z) = \frac{p(x, z)}{p(x \mid z)}$$

- The second "ingredient" is **Laplace approximation.**. Main idea: approximate a generic function $f(x)$ using a quadratic function by means of a Taylor's series expansion around the mode $x^* = \text{argmax}_x \log f(x)$. Thus, under LA, $f(x) \approx \text{Normal}(x^*, \sigma^{2^*})$.

INLA objectives

- In a Bayesian LGM, the required distributions are

$$
\begin{aligned}
p(\theta_i \mid \boldsymbol{y}) &= \int p(\theta_i, \boldsymbol{\psi} \mid \boldsymbol{y}) d\boldsymbol{\psi} = \int p(\boldsymbol{\psi} \mid \boldsymbol{y}) p(\theta_i \mid \boldsymbol{\psi}, \boldsymbol{y}) d\boldsymbol{\psi} \\
p(\psi_k \mid \boldsymbol{y}) &= \int p(\boldsymbol{\psi} \mid \boldsymbol{y}) d\boldsymbol{\psi}_{-k}
\end{aligned}
$$

INLA objectives

- In a Bayesian LGM, the required distributions are

$$p(\theta_i \mid \boldsymbol{y}) = \int p(\theta_i, \boldsymbol{\psi} \mid \boldsymbol{y}) d\boldsymbol{\psi} = \int p(\boldsymbol{\psi} \mid \boldsymbol{y}) p(\theta_i \mid \boldsymbol{\psi}, \boldsymbol{y}) d\boldsymbol{\psi}$$

$$p(\psi_k \mid \boldsymbol{y}) = \int p(\boldsymbol{\psi} \mid \boldsymbol{y}) d\boldsymbol{\psi}_{-k}$$

- Thus we need to estimate/approximate:

  (1.) $p(\boldsymbol{\psi} \mid \boldsymbol{y})$, from which also all the relevant marginals $p(\psi_k \mid \boldsymbol{y})$ can be obtained;

INLA objectives

- In a Bayesian LGM, the required distributions are

$$p(\theta_i \mid \boldsymbol{y}) = \int p(\theta_i, \boldsymbol{\psi} \mid \boldsymbol{y}) d\boldsymbol{\psi} = \int p(\boldsymbol{\psi} \mid \boldsymbol{y}) p(\theta_i \mid \boldsymbol{\psi}, \boldsymbol{y}) d\boldsymbol{\psi}$$

$$p(\psi_k \mid \boldsymbol{y}) = \int p(\boldsymbol{\psi} \mid \boldsymbol{y}) d\boldsymbol{\psi}_{-k}$$

- Thus we need to estimate/approximate:

  (1.) $p(\boldsymbol{\psi} \mid \boldsymbol{y})$, from which also all the relevant marginals $p(\psi_k \mid \boldsymbol{y})$ can be obtained;

  (2.) $p(\theta_i \mid \boldsymbol{\psi}, \boldsymbol{y})$, which is needed to compute the marginal posterior for the parameters

Integrated Nested Laplace Approximation (INLA)

(1.) can be easily estimated as

$$\begin{aligned} p(\psi \mid \boldsymbol{y}) &= \ldots \\ &\approx \left. \frac{p(\psi)p(\boldsymbol{\theta} \mid \psi)p(\boldsymbol{y} \mid \boldsymbol{\theta})}{\tilde{p}(\boldsymbol{\theta} \mid \psi, \boldsymbol{y})} \right|_{\boldsymbol{\theta}=\boldsymbol{\theta}^*(\psi)} =: \tilde{p}(\psi \mid \boldsymbol{y}) \end{aligned}$$

where

- $\tilde{p}(\boldsymbol{\theta} \mid \psi, \boldsymbol{y})$ is the Laplace approximation of $p(\boldsymbol{\theta} \mid \psi, \boldsymbol{y})$
- $\boldsymbol{\theta} = \boldsymbol{\theta}^*(\psi)$ is its mode for a given ψ

Integrated Nested Laplace Approximation (INLA)

(2.) is slightly more complex, because in general there will be more elements in $\boldsymbol{\theta}$ than there are in $\boldsymbol{\psi}$ and thus this computation is more expensive.

Integrated Nested Laplace Approximation (INLA)

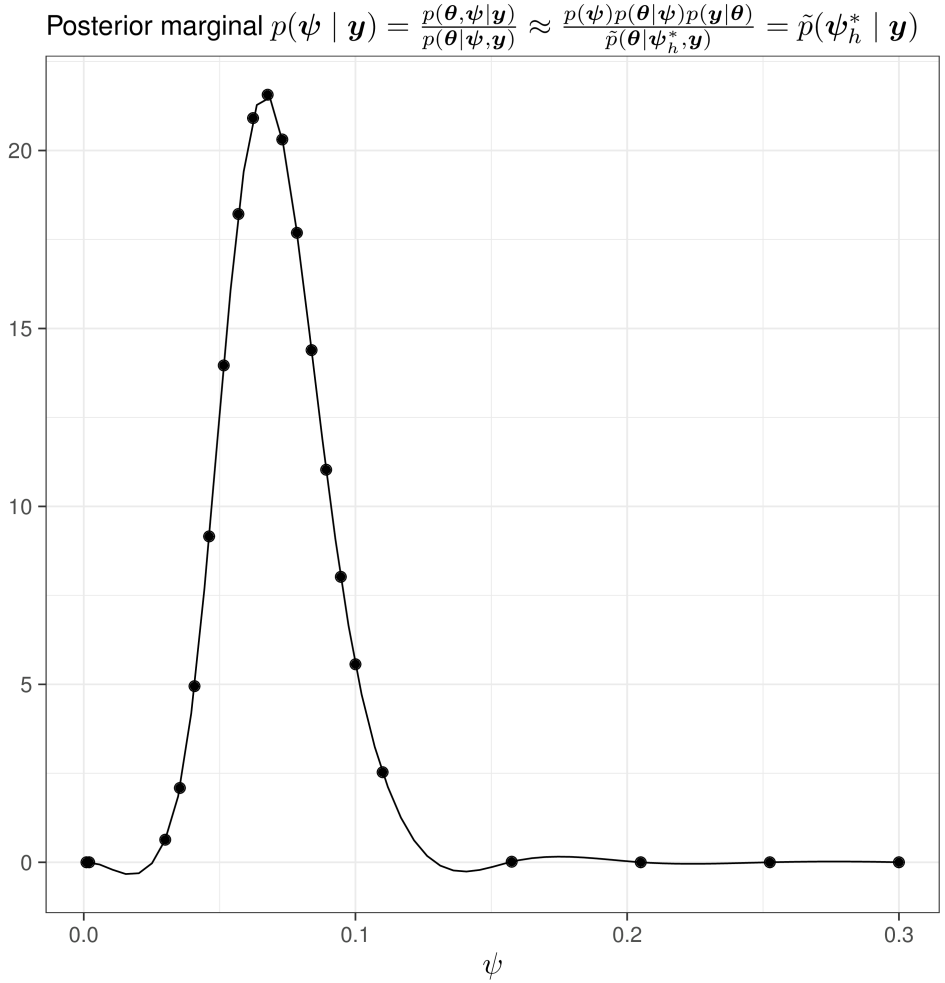
(2.) is slightly more complex, because in general there will be more elements in $\boldsymbol{\theta}$ than there are in $\boldsymbol{\psi}$ and thus this computation is more expensive.

Three possible approaches to approximate $p(\theta_i \mid \boldsymbol{\psi}, \boldsymbol{y})$:

- **Gaussian Approximation**
- **Full Laplace Approximation**
- **Simplified Laplace Approximation** (implemented by default by `R-INLA`)

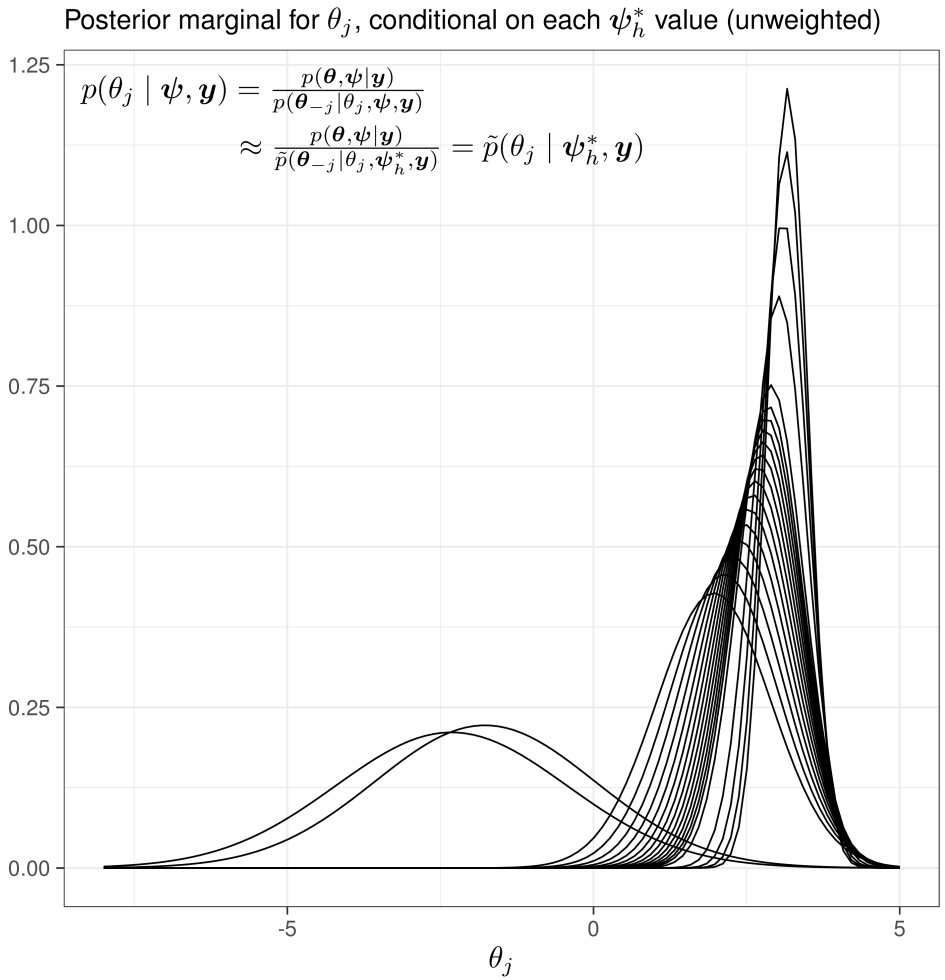
In a nutshell

1. Select a grid of H points $\{\boldsymbol{\psi}_h^*\}$ and area weights $\{\Delta_h\}$; interpolate the density to approximate to the posterior



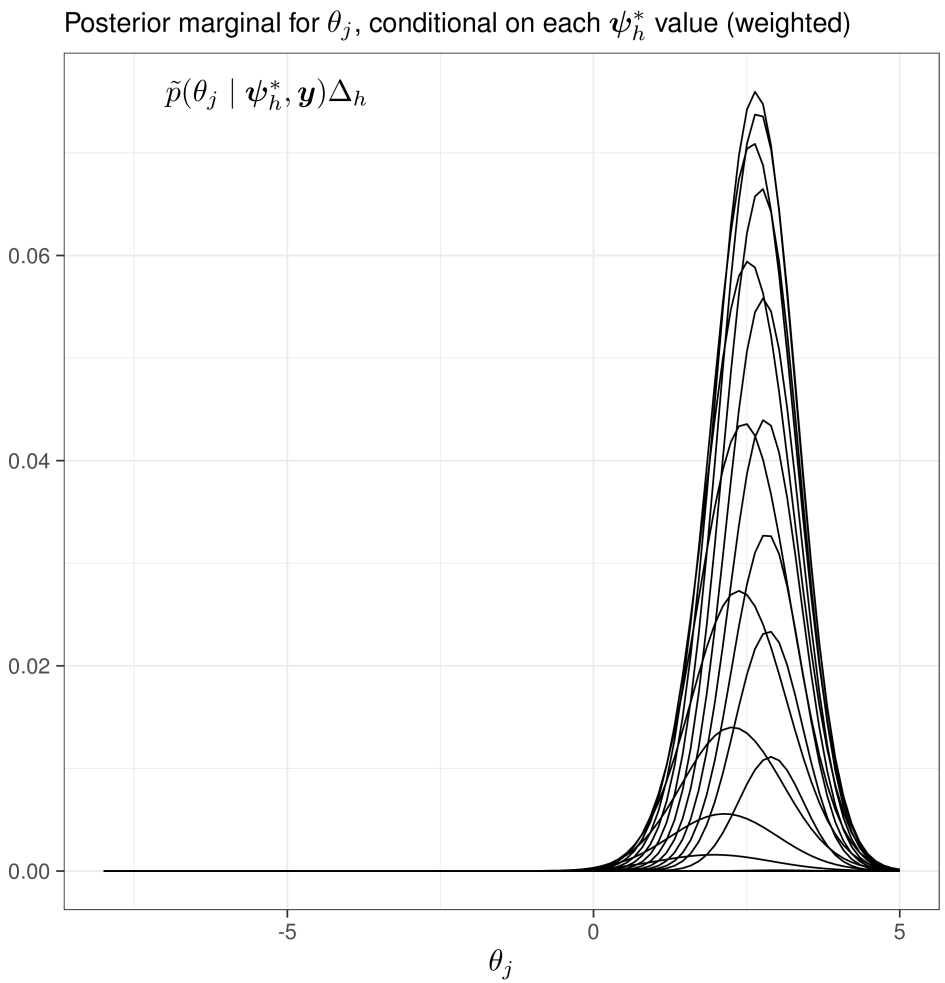
In a nutshell

2. Approximates the conditional posterior of each θ_j, given $\boldsymbol{\psi}, \boldsymbol{y}$ on the H dimensional grid



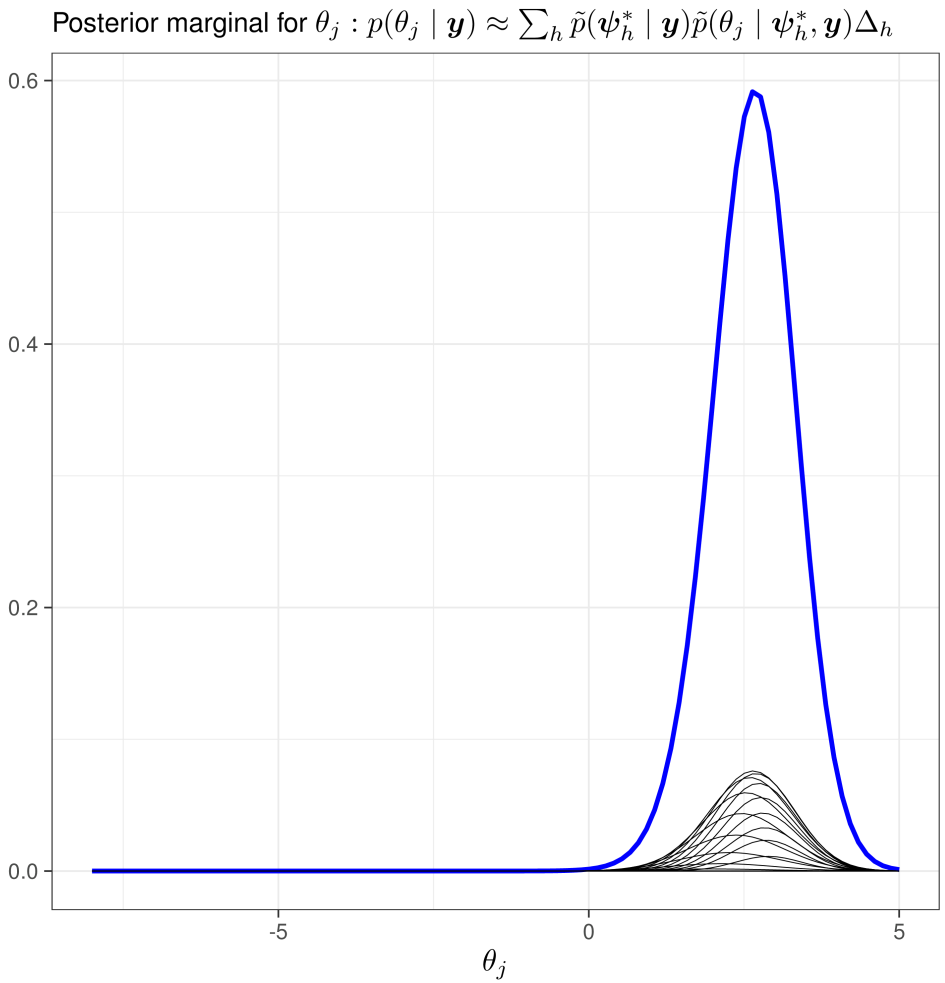
In a nutshell

3. Weight the conditional marginal posteriors by the density associated with each ψ_h^*



In a nutshell

4. (Numerically) sum over all the conditional densities to obtain the marginal posterior for θ_j



Summary so far

- The INLA approach is not a rival/competitor/replacement to/of MCMC, just a better option for the class of LGMs.
- The basic idea behind the INLA procedure is simple:
  - Repeatedly use Laplace approximation and take advantage of computational simplifications due to the structure of the model;
  - Use numerical integration to compute the required posterior marginal distributions.
- Complications are mostly computational and occur when:
  - Extending to a large number of hyperparameters;
  - Markedly non-Gaussian observations.

The R-INLA package

The INLA package for R

Good news is that all the procedures needed to perform INLA are implemented in a R package. This is effectively made by two components

The INLA package for R

Good news is that all the procedures needed to perform INLA are implemented in a R package. This is effectively made by two components

1. The GMRFLib library
   - A C library for fast and exact simulation of GMRFs

The INLA package for R

Good news is that all the procedures needed to perform INLA are implemented in a R package. This is effectively made by two components

1. The GMRFLib library
   - A C library for fast and exact simulation of GMRFs
2. The inla program
   - A standalone C program build upon the GMRFLib library (it performs the relevant computation and returns the results in a standardised way)

The INLA package for R

Good news is that all the procedures needed to perform INLA are implemented in a R package. This is effectively made by two components

1. The GMRFLib library
   - A C library for fast and exact simulation of GMRFs
2. The inla program
   - A standalone C program build upon the GMRFLib library (it performs the relevant computation and returns the results in a standardised way)

NB: Because the package R-INLA relies on a standalone C program (and other reasons...), it is not available directly from CRAN.

R-INLA runs natively under Linux, Windows and Mac and it is possible to do multi-threading using OpenMP

The INLA package for R — Installation

- From R, installation of the stable version is performed typing

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

- Later, you can upgrade the package by typing

```
library(INLA)
inla.upgrade()
```

- A test-version (which may contain unstable updates/new functions) can be obtained by typing

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

- Type `inla.version()` to find out the installed version

The INLA package for R — Installation

- From R, installation of the stable version is performed typing

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

- Later, you can upgrade the package by typing

```
library(INLA)
inla.upgrade()
```

- A test-version (which may contain unstable updates/new functions) can be obtained by typing

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

- Type `inla.version()` to find out the installed version

Step by step guide to using R-INLA

1. The first thing to do is to **specify the model**

- For example, assume we have a generic model

$$\begin{aligned} y_i &\overset{iid}{\sim} p(y_i \mid \theta_i) \\ \eta_i &= \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + f(z_i) \end{aligned}$$

where

  - $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2)$ are observed covariates
  - $\boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2) \sim \text{Normal}(0, \tau_1^{-1})$ are unstructured ("fixed") effects
  - $\boldsymbol{z}$ is an **index**. This can be used to include structured ("random"), spatial, spatio-temporal effect, etc.
  - $f \sim \text{Normal}(\boldsymbol{0}, \boldsymbol{Q}_f^{-1}(\tau_2))$ is a suitable function used to model the random (i.e. structured) effects

Step by step guide to using R-INLA

1. The first thing to do is to **specify the model**

- For example, assume we have a generic model

$$\begin{aligned} y_i &\overset{iid}{\sim} p(y_i \mid \theta_i) \\ \eta_i &= \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + f(z_i) \end{aligned}$$

where

  - $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2)$ are observed covariates
  - $\boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2) \sim \text{Normal}(0, \tau_1^{-1})$ are unstructured ("fixed") effects
  - $\boldsymbol{z}$ is an **index**. This can be used to include structured ("random"), spatial, spatio-temporal effect, etc.
  - $f \sim \text{Normal}(\boldsymbol{0}, \boldsymbol{Q}_f^{-1}(\tau_2))$ is a suitable function used to model the random (i.e. structured) effects

- As mentioned earlier, this formulation can actually be used to represent quite a wide class of models!

Step by step guide to using R-INLA

- The model is translated in R code using a formula
- This is sort of standard in R (you would do pretty much the same for calls to functions such as lm or glm)

```
formula = y ~ 1+ x1 + x2 + f(name=z, model="...",hyper=...)
```

Step by step guide to using R-INLA

- The model is translated in R code using a formula
- This is sort of standard in R (you would do pretty much the same for calls to functions such as lm or glm)

```
formula = y ~ 1+ x1 + x2 + f(name=z, model="...",hyper=...)
```

- The f() function can account for several structured nonlinear effects. We have for example:
  - iid specify independent random effects
  - rw1, rw2, ar1 are smooth effect of covariates or time effects
  - besag models spatially structured effects (CAR)
  - generic is a user-defined precision matrix
- Type inla.list.models("latent") for the complete list and find descriptions at https://www.rinla.org/documentation
- Some options can be specified for the f() term: for example hyper is used to specify the prior on the hyperparameters (more later).

Step by step guide to using R-INLA

- The model is translated in R code using a formula
- This is sort of standard in R (you would do pretty much the same for calls to functions such as lm or glm)

```
formula = y ~ 1+ x1 + x2 + f(name=z, model="...",hyper=...)
```

- The f() function can account for several structured nonlinear effects. We have for example:
  - iid specify independent random effects
  - rw1, rw2, ar1 are smooth effect of covariates or time effects
  - besag models spatially structured effects (CAR)
  - generic is a user-defined precision matrix
- Type inla.list.models("latent") for the complete list and find descriptions at https://www.rinla.org/documentation
- Some options can be specified for the f() term: for example hyper is used to specify the prior on the hyperparameters (more later).
- It is possible to include in the formula several f() terms specifying them separately, e.g.

```
formula <- y ~ x1 + x2 + f(z1,...)  + f(z2,...)  +
f(z3,...)
```

Step by step guide to using R-INLA

2. Call the function `inla` to fit the model, specifying the data and options (more on this later), eg

```
m = inla(formula, family="...", data=data.frame(y,x1,x2,z))
```

Step by step guide to using R-INLA

2. Call the function `inla` to fit the model, specifying the data and options (more on this later), eg

```
m = inla(formula, family="...", data=data.frame(y,x1,x2,z))
```

- The data need to be included in a suitable `data.frame`
- The distribution of the data (i.e. the likelihood) is specified with the `family` option.
  - Type `inla.list.models("likelihood")` for the complete list of likelihood function (we have `poisson, binomial, gamma, beta, gaussian` and many others).
  - Visit `https://www.r-inla.org/documentation` for the complete description

Step by step guide to using R-INLA

The `control.xxx=list(...)` statements in the `inla` function control various part of the INLA program:

- `control.compute`: for computing measures of fit (eg DIC)
- `control.predictor`: for specifying the *Observation matrix* $\boldsymbol{A}$ which links the latent field to the data
- `control.family`: for changing the prior distribution of the likelihood hyperparameters
- `control.fixed`: for changing the prior distribution of the fixed effects
- `control.inla`: for changing the strategy to use for the approximations ('gaussian', 'simplified.laplace' (default) or 'laplace') or the grid exploration strategy
- and many others for expert use.

Step by step guide to using R-INLA

R returns an object `m` in the class `inla`, which has some methods available as for example `summary()` and `plot()`.

Name	Description
`summary.fitted.values`	Summary statistics of fitted values.
`summary.fixed`	Summary statistics of fixed effects.
`summary.random`	Summary statistics of random effects.
`summary.linear.predictor`	Summary statistics of linear predictors.
`marginals.fitted.values`	Posterior marginals of fitted values.
`marginals.fixed`	Posterior marginals of fixed effects.
`marginals.random`	Posterior marginals of random effects.
`marginals.linear.predictor`	Posterior marginals of linear predictors.
`mlik`	Estimate of marginal likelihood.

Source: Gomez-Rubio 2020, Section 2.3.1

Toy example

We consider the `iris` dataset included in the `R` datasets (see `?iris`), regarding the measurements in centimeters of the variables *sepal length* and *width* and *petal length* and *width*, respectively, for 50 flowers from each of 3 species of iris. The species are Iris setosa, versicolor, and virginica. See Section 2.6 of the INLA book.

```
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa    :50
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
```

We specify a simple regression model with `Petal.length` and `Petal.width` as dependent and independent variables, respectively.

$$\begin{aligned} \text{Petal.length}_i &\sim \text{Normal}(\eta_i, 1/\sigma_e^2) \\ \eta_i &= \beta_0 + \beta_1 \, \text{Petal.width}_i \end{aligned}$$

Toy example: run INLA + summary

```
> library(INLA)
> formula <- Petal.Length ~  1 + Petal.Width
> output <- inla(formula, family="gaussian", data=iris)
> summary(output)

Call:
"inla(formula = formula, family = \"gaussian\", data = iris)"

Time used:
 Pre-processing    Running inla Post-processing           Total
         0.2829          0.1971          0.0863          0.5664

Fixed effects:
              mean     sd 0.025quant 0.5quant 0.975quant   mode kld
(Intercept) 1.0836 0.0727     0.9407   1.0836     1.2263 1.0836   0
Petal.Width 2.2299 0.0512     2.1293   2.2299     2.3305 2.2299   0

The model has no random effects

Model hyperparameters:
                                          mean     sd 0.025quant 0.5quant 0.975quant  mode
Precision for the Gaussian observations 4.43 0.5117      3.495    4.408      5.501 4.368

Expected number of effective parameters(std dev): 2.011(0.001)
Number of equivalent replicates : 74.59

Marginal Likelihood:  -119.59
```

Exploring the output: fixed effects

```
> output$summary.fixed
               mean     sd 0.025quant 0.5quant 0.975quant   mode kld
(Intercept) 1.0836 0.0727     0.9407   1.0836     1.2263 1.0836   0
Petal.Width 2.2299 0.0512     2.1293   2.2299     2.3305 2.2299   0
```

- For each unstructured ("fixed") effect, R-INLA reports a set of summary statistics from the posterior distribution.

Exploring the output: fixed effects

```
> output$summary.fixed
               mean     sd 0.025quant 0.5quant 0.975quant   mode kld
(Intercept) 1.0836 0.0727     0.9407   1.0836     1.2263 1.0836   0
Petal.Width 2.2299 0.0512     2.1293   2.2299     2.3305 2.2299   0
```

- For each unstructured ("fixed") effect, `R-INLA` reports a set of summary statistics from the posterior distribution.
- The value of the Kullback-Leibler divergence (`kld`) describes the difference between the Gaussian approximation and the Simplified Laplace Approximation (SLA) to the marginal posterior densities:
  - Small values indicate that the posterior distribution is well approximated by a Normal distribution
  - If so, the more sophisticated SLA gives a "good" error rate and therefore there is no need to use the more computationally intensive "full" Laplace approximation.

Exploring the output: hyperparameters

```
> output$summary.hyperpar
                                                mean     sd 0.025quant 0.5quant 0.975quant   mode
Precision for the Gaussian observations 4.4304 0.5117     3.4952   4.4076     5.5014 4.3681
```

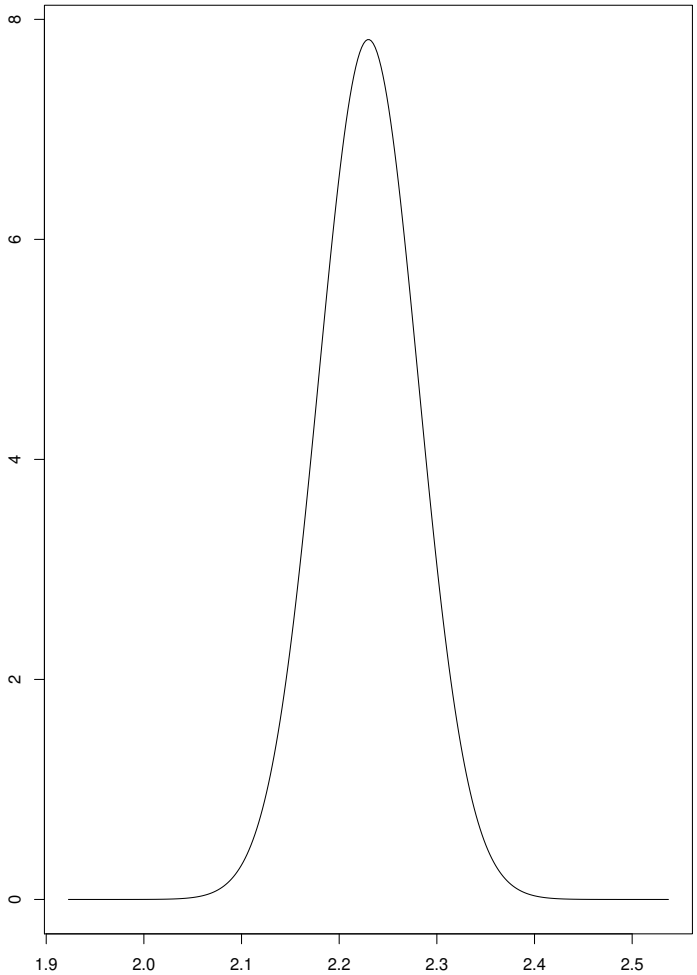
- For each hyperparameter the summary statistics are reported to describe the posterior distribution.
- **NB**: `INLA` reports results on the **precision** scale (more on this later).

Manipulating the marginals from R-INLA: fixed effects

```
> names(output$marginals.fixed)
[1] "(Intercept)" "Petal.Width"

> beta1_post <- output$marginals.fixed[[1]]
> plot(inla.smarginal(beta1_post),t="l")
```

Marginal posterior:

$p(\beta_1 \mid \boldsymbol{y})$

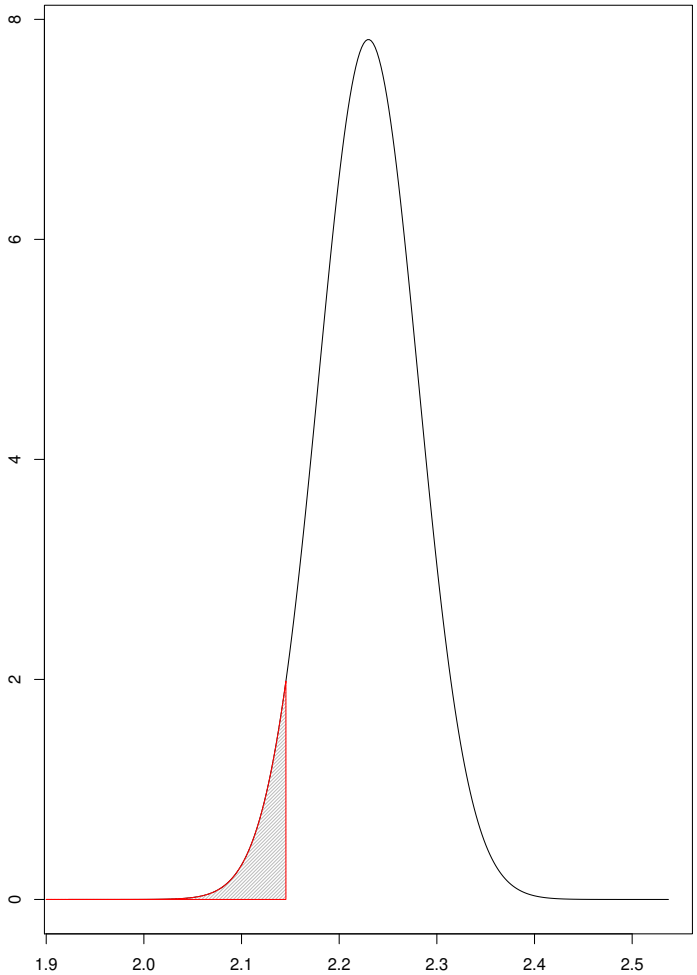


Manipulating the marginals from R-INLA: fixed effects

```
> names(output$marginals.fixed)
[1] "(Intercept)" "Petal.Width"

> beta1_post <- output$marginals.fixed[[1]]
> plot(inla.smarginal(beta1_post),t="l")


> inla.qmarginal(0.05,beta1_post)
[1] 2.145447
```

Marginal posterior:

$p(\beta_1 \mid \boldsymbol{y})$

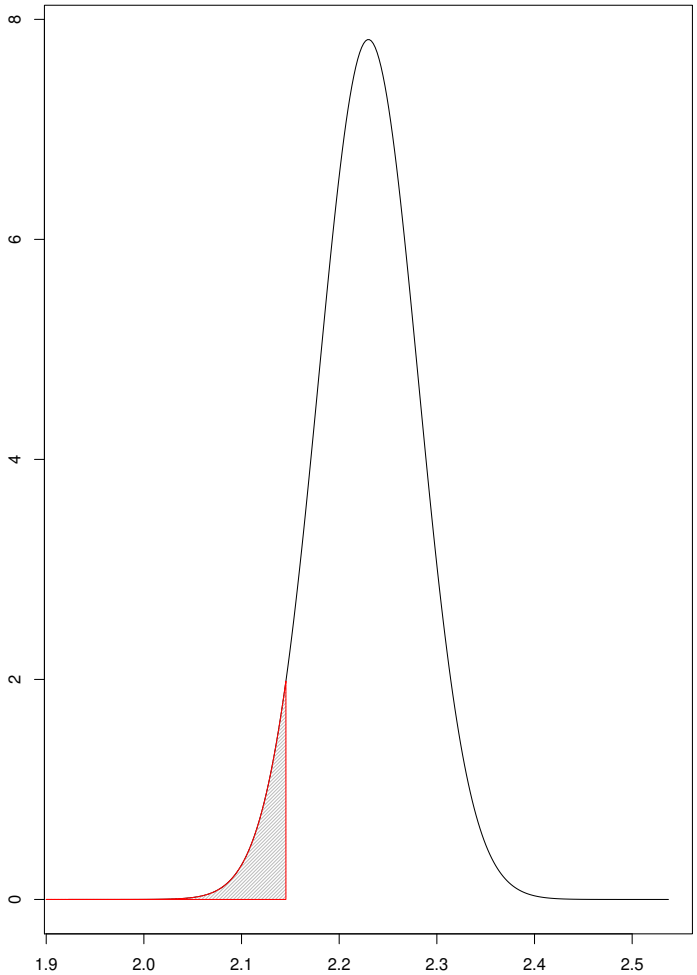


Manipulating the marginals from R-INLA: fixed effects

```
> names(output$marginals.fixed)
[1] "(Intercept)" "Petal.Width"

> beta1_post <- output$marginals.fixed[[1]]
> plot(inla.smarginal(beta1_post),t="l")


> inla.qmarginal(0.05,beta1_post)
[1] 2.145447


> inla.pmarginal(2.145447,beta1_post)
[1] 0.04999947
```

Marginal posterior:

$p(\beta_1 \mid \boldsymbol{y})$

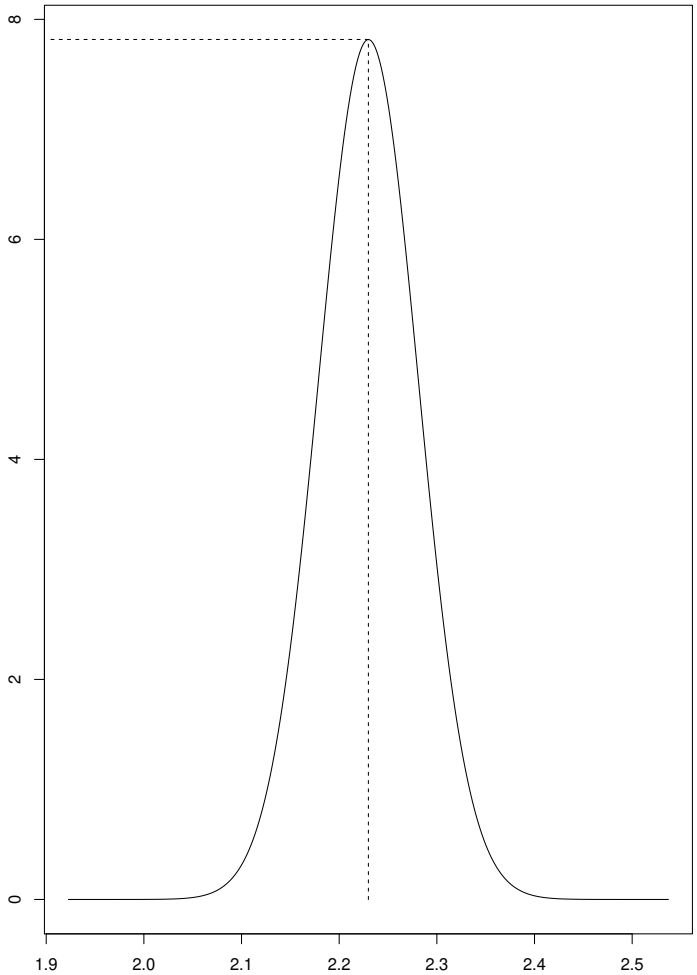


Manipulating the marginals from R-INLA: fixed effects

```
> names(output$marginals.fixed)
[1] "(Intercept)" "Petal.Width"

> beta1_post <- output$marginals.fixed[[1]]
> plot(inla.smarginal(beta1_post),t="l")


> inla.qmarginal(0.05,beta1_post)
[1] 2.145447


> inla.pmarginal(2.145447,beta1_post)
[1] 0.04999947


> inla.dmarginal(2.2299,beta1_post)
[1] 7.817155
```

Marginal posterior:

$p(\beta_1 \mid \boldsymbol{y})$



Manipulating the marginals from R-INLA: fixed effects

```
> names(output$marginals.fixed)
[1] "(Intercept)" "Petal.Width"

> beta1_post <- output$marginals.fixed[[1]]
> plot(inla.smarginal(beta1_post),t="l")


> inla.qmarginal(0.05,beta1_post)
[1] 2.145447


> inla.pmarginal(2.145447,beta1_post)
[1] 0.04999947


> inla.dmarginal(2.2299,beta1_post)
[1] 7.817155


> inla.rmarginal(4,beta1_post)
[1] 2.212335 2.102212 2.170217 2.201038
```

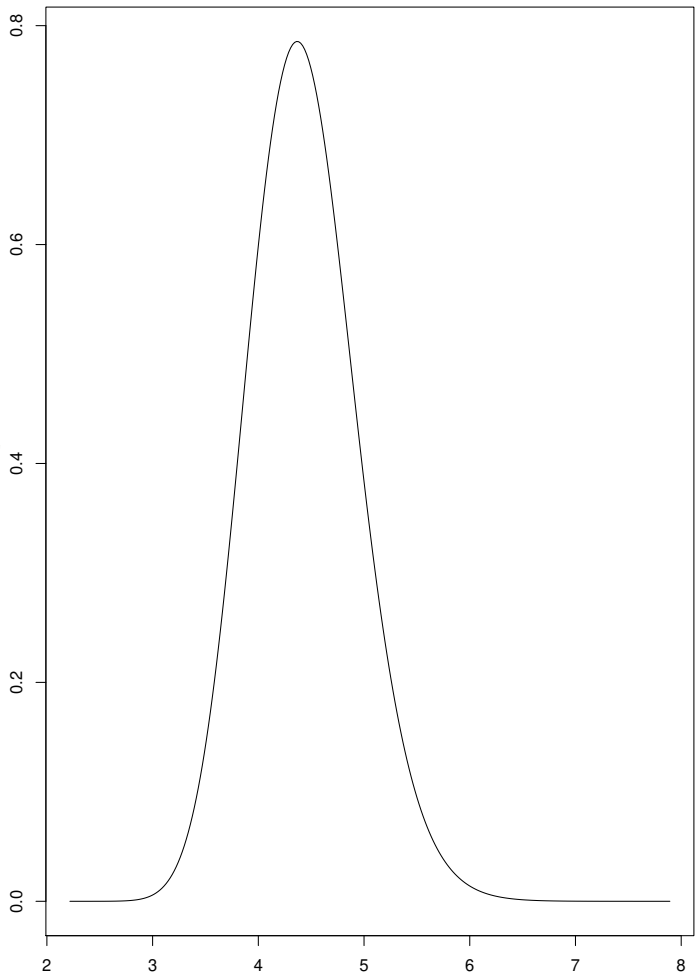
Manipulating the marginals from R-INLA: hyperparameters

INLA works by default with **precisions**

```
> names(output$marginals.hyperpar)
[1] "Precision for the Gaussian observations"
```

```
> prec_post = output$marginals.hyperpar[[1]]
> plot(inla.smarginal(prec_post),t="l")
```

Marginal posterior: $p(1/\sigma_e^2 \mid \boldsymbol{y})$



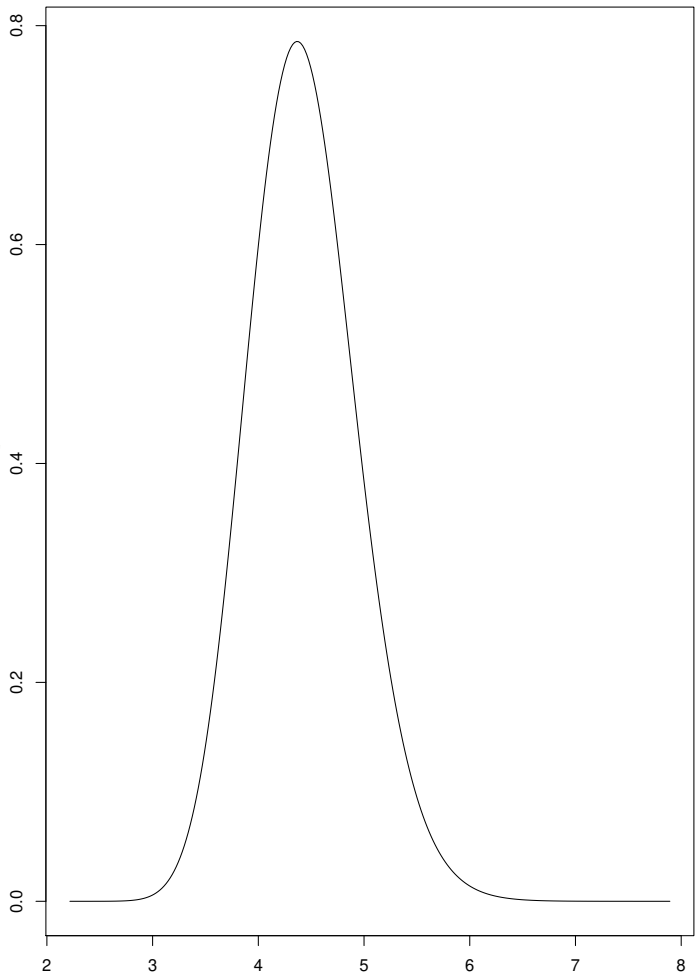
Manipulating the marginals from R-INLA: hyperparameters

INLA works by default with **precisions**

```
> names(output$marginals.hyperpar)
[1] "Precision for the Gaussian observations"
```

```
> prec_post = output$marginals.hyperpar[[1]]
> plot(inla.smarginal(prec_post),t="l")
```

Marginal posterior: $p(1/\sigma_e^2 \mid \boldsymbol{y})$



Problem: usually, we want to make inference on more interpretable parameters, eg **standard deviations**

Manipulating the marginals from R-INLA: hyperparameters

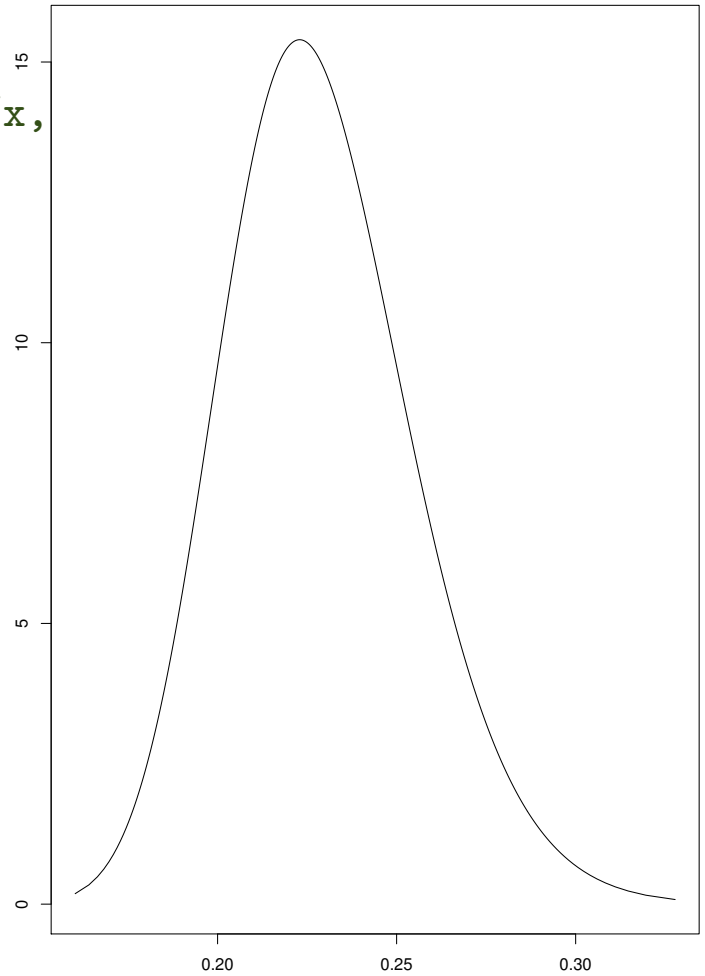
Using some built-in INLA functions it is possible to compute the variability, on the variance scale.

```
> var_post = inla.tmarginal(fun=function(x) 1/x,
mar=prec_post)
> plot(var_post,type="l")
```

```
> inla.emarginal(fun=function(x) 1/x,
marg=prec_post)
[1] 0.2287442
```

```
> m1 = inla.emarginal(function(x) 1/x,
prec_post)
> m2 = inla.emarginal(function(x) (1/x)^2,
 prec_post)
> sqrt(m2 - m1^2)
[1] 0.02660592
```

Marginal posterior: $p(\sigma_e^2 \mid \boldsymbol{y})$



Summary

- The INLA approach is not a rival/competitor/replacement to/of MCMC, just a better option for the class of LGMs.
- The basic idea behind the INLA procedure is simple
  - Repeatedly use Laplace approximation and take advantage of computational simplifications due to the structure of the model
  - Use numerical integration to compute the required posterior marginal distributions
- Complications are mostly computational and occur when
  - Extending to a large number of hyperparameters
  - Markedly non-Gaussian observations