A (ridiculously short and incomplete) introduction to INLA

Gianluca Baio

(Thanks to Håvard Rue)

University College London
Department of Statistical Science

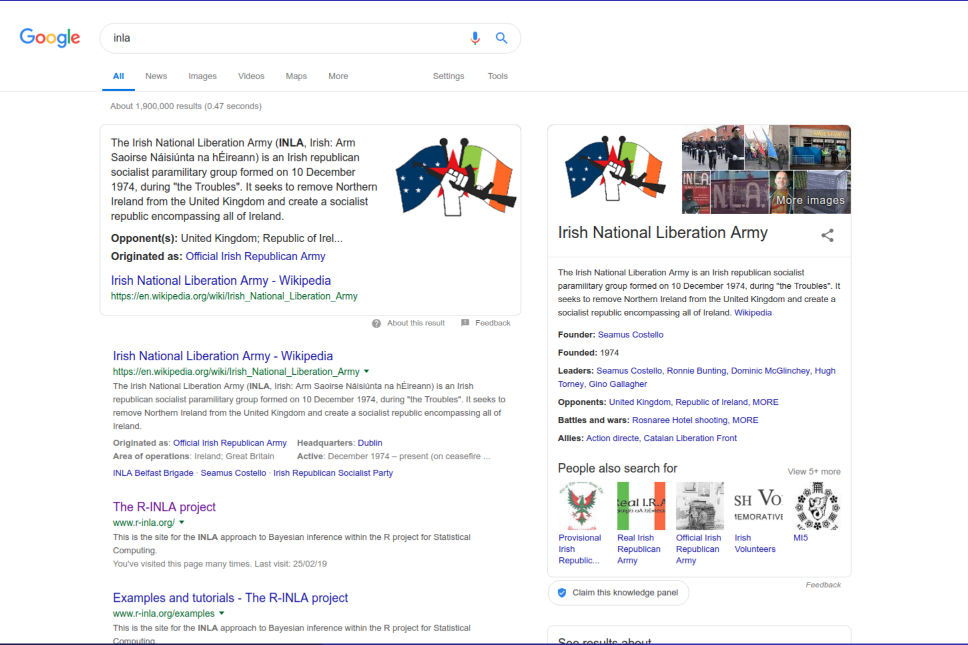
g.baio@ucl.ac.uk
http://www.ucl.ac.uk/statistics/research/statistics-health-economics/
http://www.statistica.it/gianluca
https://github.com/giabaio

Network of Applied Statisticians in Health Seminar Series
University College London

Wednesday 3 April 2019

Laplace's Liberation Army?

Know what you're looking for...



inla

All News Images Videos Maps More Settings Tools

About 1,900,000 results (0.47 seconds)

The Irish National Liberation Army (**INLA**, Irish: Arm Saoirse Náisiúnta na hÉireann) is an Irish republican socialist paramilitary group formed on 10 December 1974, during "the Troubles". It seeks to remove Northern Ireland from the United Kingdom and create a socialist republic encompassing all of Ireland.

Opponent(s): United Kingdom; Republic of Irel...

Originated as: Official Irish Republican Army

Irish National Liberation Army - Wikipedia
https://en.wikipedia.org/wiki/Irish_National_Liberation_Army

About this result Feedback

Irish National Liberation Army - Wikipedia
https://en.wikipedia.org/wiki/Irish_National_Liberation_Army ▾
The Irish National Liberation Army (**INLA**, Irish: Arm Saoirse Náisiúnta na hÉireann) is an Irish republican socialist paramilitary group formed on 10 December 1974, during "the Troubles". It seeks to remove Northern Ireland from the United Kingdom and create a socialist republic encompassing all of Ireland.

Originated as: Official Irish Republican Army **Headquarters**: Dublin
Area of operations: Ireland; Great Britain **Active**: December 1974 – present (on ceasefire ...

INLA Belfast Brigade · Seamus Costello · Irish Republican Socialist Party

The R-INLA project
www.r-inla.org/ ▾
This is the site for the **INLA** approach to Bayesian inference within the R project for Statistical Computing.
You've visited this page many times. Last visit: 25/02/19

Examples and tutorials - The R-INLA project
www.r-inla.org/examples ▾
This is the site for the **INLA** approach to Bayesian inference within the R project for Statistical Computing.

More images

Irish National Liberation Army

The Irish National Liberation Army is an Irish republican socialist paramilitary group formed on 10 December 1974, during "the Troubles". It seeks to remove Northern Ireland from the United Kingdom and create a socialist republic encompassing all of Ireland. Wikipedia

Founder: Seamus Costello

Founded: 1974

Leaders: Seamus Costello, Ronnie Bunting, Dominic McGlinchey, Hugh Torney, Gino Gallagher

Opponents: United Kingdom, Republic of Ireland, MORE

Battles and wars: Rosnaree Hotel shooting, MORE

Allies: Action directe, Catalan Liberation Front

People also search for View 5+ more

Provisional Irish Republic... · Real Irish Republican Army · Official Irish Republican Army · Irish Volunteers · MI5

Feedback

Claim this knowledge panel

Bayesian computation

- In a (**very** small!) nutshell, Bayesian inference boils down to the computation of **posterior** and/or **predictive** distributions

$$p(\boldsymbol{\theta} \mid \boldsymbol{y}) = \frac{p(\boldsymbol{y} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\boldsymbol{y} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}} \qquad p(y^* \mid \boldsymbol{y}) = \int p(y^* \mid \boldsymbol{\theta})p(\boldsymbol{\theta} \mid \boldsymbol{y})d\boldsymbol{\theta}$$

- In a (**very** small!) nutshell, Bayesian inference boils down to the computation of **posterior** and/or **predictive** distributions

$$p(\boldsymbol{\theta} \mid \boldsymbol{y}) = \frac{p(\boldsymbol{y} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\boldsymbol{y} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}} \qquad p(y^* \mid \boldsymbol{y}) = \int p(y^* \mid \boldsymbol{\theta})p(\boldsymbol{\theta} \mid \boldsymbol{y})d\boldsymbol{\theta}$$

- Since the advent of simulation-based techniques (notably MCMC), Bayesian computation has enjoyed incredible development
- This has certainly been helped by dedicated software (eg `BUGS` and then `WinBUGS`, `OpenBUGS`, `JAGS`)
- MCMC methods are very general and can effectively be applied to "any" model

- In a (**very** small!) nutshell, Bayesian inference boils down to the computation of **posterior** and/or **predictive** distributions

$$p(\boldsymbol{\theta} \mid \boldsymbol{y}) = \frac{p(\boldsymbol{y} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\boldsymbol{y} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}} \qquad p(y^* \mid \boldsymbol{y}) = \int p(y^* \mid \boldsymbol{\theta})p(\boldsymbol{\theta} \mid \boldsymbol{y})d\boldsymbol{\theta}$$

- Since the advent of simulation-based techniques (notably MCMC), Bayesian computation has enjoyed incredible development
- This has certainly been helped by dedicated software (eg `BUGS` and then `WinBUGS`, `OpenBUGS`, `JAGS`)
- MCMC methods are very general and can effectively be applied to "any" model
- However:
  - Even if **in theory**, MCMC can provide (nearly) exact inference, given perfect convergence and MC error $\rightarrow 0$, in practice, this has to be balanced with model complexity and running time
  - This is particularly an issue for problems characterised by large data or very complex structure (eg hierarchical models)

MCMC

- "Standard" MCMC sampler are generally easy-ish to program and are in fact implemented in readily available software
- However, depending on the complexity of the problem, their efficiency might be limited

MCMC

- "Standard" MCMC sampler are generally easy-ish to program and are in fact implemented in readily available software
- However, depending on the complexity of the problem, their efficiency might be limited

- Possible solutions
  1. More complex **model specification**
     - Blocking
     - Overparameterisation

MCMC

- "Standard" MCMC sampler are generally easy-ish to program and are in fact implemented in readily available software
- However, depending on the complexity of the problem, their efficiency might be limited
- Possible solutions
  1. More complex **model specification**
     - Blocking
     - Overparameterisation
  2. More complex **sampling schemes**
     - Hamiltonian Monte Carlo
     - No U-turn sampling (eg `stan`) — more on this later!

Pros & cons

- "Standard" MCMC sampler are generally easy-ish to program and are in fact implemented in readily available software
- However, depending on the complexity of the problem, their efficiency might be limited
- Possible solutions
  1. More complex **model specification**
     - Blocking
     - Overparameterisation
  2. More complex **sampling schemes**
     - Hamiltonian Monte Carlo
     - No U-turn sampling (eg `stan`) — more on this later!
  3. Alternative methods of inference
     - Approximate Bayesian Computation (ABC)
     - INLA

MCMC

- "Standard" MCMC sampler are generally easy-ish to program and are in fact implemented in readily available software
- However, depending on the complexity of the problem, their efficiency might be limited
- Possible solutions
  1. More complex **model specification**
     - Blocking
     - Overparameterisation
  2. More complex **sampling schemes**
     - Hamiltonian Monte Carlo
     - No U-turn sampling (eg `stan`) — more on this later!
  3. Alternative methods of inference
     - Approximate Bayesian Computation (ABC)
     - INLA — more on this now!

Basics of INLA

The basic ideas revolve around

- Formulating the model using a specific characterisation
  - All models that can be formulated in this way have certain features in common, which facilitate the computational aspects
  - The characterisation is still quite general and covers a wide range of possible models (more on that later!)
  - **NB**: This implies less flexibility with respect to MCMC — but in many cases this is not a huge limitation!

Basics of INLA

The basic ideas revolve around

- Formulating the model using a specific characterisation
  - All models that can be formulated in this way have certain features in common, which facilitate the computational aspects
  - The characterisation is still quite general and covers a wide range of possible models (more on that later!)
  - **NB**: This implies less flexibility with respect to MCMC — but in many cases this is not a huge limitation!

- Use some basic probability conditions to approximate the relevant distributions

Basics of INLA

The basic ideas revolve around

- Formulating the model using a specific characterisation
  - All models that can be formulated in this way have certain features in common, which facilitate the computational aspects
  - The characterisation is still quite general and covers a wide range of possible models (more on that later!)
  - **NB**: This implies less flexibility with respect to MCMC — but in many cases this is not a huge limitation!

- Use some basic probability conditions to approximate the relevant distributions

- Compute the relevant quantities typically using numerical methods

Basics of INLA

The basic ideas revolve around

- Formulating the model using a specific characterisation
  - All models that can be formulated in this way have certain features in common, which facilitate the computational aspects
  - The characterisation is still quite general and covers a wide range of possible models (more on that later!)
  - **NB**: This implies less flexibility with respect to MCMC — but in many cases this is not a huge limitation!

- Use some basic probability conditions to approximate the relevant distributions

- Compute the relevant quantities typically using numerical methods

For a longer, more structured (but older) version of this talk see:
`http://www.statistica.it/gianluca/Talks/INLA.pdf`

Latent Gaussian models (LGMs)

- The general problem of (parametric) inference is posited by assuming a probability model for the observed data, 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, 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)

$$\boldsymbol{\theta} \mid \boldsymbol{\psi} \sim \text{Normal}(\mathbf{0}, \boldsymbol{\Sigma}(\boldsymbol{\psi}))$$
$$\theta_l \perp\!\!\!\perp \theta_m \mid \boldsymbol{\theta}_{-lm} \Leftrightarrow \boldsymbol{Q}_{lm} = \boldsymbol{\Sigma}^{-1}_{lm} = 0$$

  where
  - The notation "$-lm$" indicates all the other elements of the parameters vector, excluding elements l and m
  - **NB**: Conditional independence implies that the precision matrix $\boldsymbol{Q}$ is sparse (simplify calculations!)
  - The covariance matrix $\boldsymbol{\Sigma}$ depends on some hyper-parameters $\boldsymbol{\psi}$

Latent Gaussian models (LGMs)

- The general problem of (parametric) inference is posited by assuming a probability model for the observed data, 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)

$$\boldsymbol{\theta} \mid \boldsymbol{\psi} \sim \text{Normal}(\mathbf{0}, \boldsymbol{\Sigma}(\boldsymbol{\psi}))$$
$$\theta_l \perp\!\!\!\perp \theta_m \mid \boldsymbol{\theta}_{-lm} \Leftrightarrow \boldsymbol{Q}_{lm} = \boldsymbol{\Sigma}^{-1}_{lm} = 0$$

  where
  - The notation "$-lm$" indicates all the other elements of the parameters vector, excluding elements l and m
  - **NB**: Conditional independence implies that the precision matrix $\boldsymbol{Q}$ is **sparse** (simplify calculations!)
  - The covariance matrix $\boldsymbol{\Sigma}$ depends on some hyper-parameters $\boldsymbol{\psi}$

- This kind of models is often referred to as **Latent Gaussian models**

LGMs as a general framework

- In general, we can partition $\boldsymbol\psi = (\boldsymbol\psi_1, \boldsymbol\psi_2)$ and re-express a LGM as

$$\begin{aligned}
\boldsymbol\psi &\sim p(\boldsymbol\psi) && \text{(“hyperprior”)}\\
\boldsymbol\theta \mid \boldsymbol\psi &\sim p(\boldsymbol\theta \mid \boldsymbol\psi) = \text{Normal}(0, \boldsymbol\Sigma(\boldsymbol\psi_1)) && \text{(“GMRF prior”)}\\
\boldsymbol y \mid \boldsymbol\theta, \boldsymbol\psi &\sim \prod_i p(y_i \mid \boldsymbol\theta, \boldsymbol\psi_2) && \text{(“data model”)}
\end{aligned}$$

i.e. $\boldsymbol\psi_1$ are the **hyper-parameters** and $\boldsymbol\psi_2$ are **nuisance parameters**

LGMs as a general framework

- In general, we can partition $\boldsymbol\psi = (\boldsymbol\psi_1, \boldsymbol\psi_2)$ and re-express a LGM as

$$\begin{aligned} \boldsymbol\psi &\sim p(\boldsymbol\psi) && \text{("hyperprior")} \\ \boldsymbol\theta \mid \boldsymbol\psi &\sim p(\boldsymbol\theta \mid \boldsymbol\psi) = \text{Normal}(0, \boldsymbol\Sigma(\boldsymbol\psi_1)) && \text{("GMRF prior")} \\ \boldsymbol y \mid \boldsymbol\theta, \boldsymbol\psi &\sim \prod_i p(y_i \mid \boldsymbol\theta, \boldsymbol\psi_2) && \text{("data model")} \end{aligned}$$

  i.e. $\boldsymbol\psi_1$ are the **hyper-parameters** and $\boldsymbol\psi_2$ are **nuisance parameters**

- The dimension of $\boldsymbol\theta$ can be very large (e.g. 10^2-10^5)
- Conversely, because of the conditional independence properties, the dimension of $\boldsymbol\psi$ needs to be generally small (e.g. 1-5)

LGMs as a general framework

- A very general way of specifying the problem is by modelling the mean for the i-th unit by means of an additive linear predictor, defined on a suitable scale (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;
  - $(\beta_1, \ldots, \beta_M)$ quantify the effect of $\boldsymbol{x} = (x_1, \ldots, x_M)$ on the response;
  - $\boldsymbol{f} = \{f_1(\cdot), \ldots, f_L(\cdot)\}$ is a set of functions defined in terms of some covariates $\boldsymbol{z} = (z_1, \ldots, z_L)$

  and then assume

$$\boldsymbol{\theta} = (\boldsymbol{\beta}, \boldsymbol{f}) \sim \text{GMRF}(\boldsymbol{\psi})$$

LGMs as a general framework

- A very general way of specifying the problem is by modelling the mean for the i-th unit by means of an additive linear predictor, defined on a suitable scale (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;
  - $(\beta_1, \ldots, \beta_M)$ quantify the effect of $\boldsymbol{x} = (x_1, \ldots, x_M)$ on the response;
  - $\boldsymbol{f} = \{f_1(\cdot), \ldots, f_L(\cdot)\}$ is a set of functions defined in terms of some covariates $\boldsymbol{z} = (z_1, \ldots, z_L)$

  and then assume

$$\boldsymbol{\theta} = (\boldsymbol{\beta}, \boldsymbol{f}) \sim \text{GMRF}(\boldsymbol{\psi})$$

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

LGMs as a general framework

Upon varying the form of the functions $f_l(\cdot)$, this formulation can accommodate a wide range of models

LGMs as a general framework

Upon varying the form of the functions $f_l(\cdot)$, this formulation can accommodate a wide range of models

- Standard regression
  - $f_l(\cdot) =$ NULL

LGMs as a general framework

Upon varying the form of the functions $f_l(\cdot)$, this formulation can accommodate a wide range of models

- Standard regression
  - $f_l(\cdot) =$ NULL
- Hierarchical models
  - $f_l(\cdot) \sim$ Normal$(0, \sigma^2_f)$ (Exchangeable)

    $\sigma^2_f \mid \psi \sim$ some common distribution

LGMs as a general framework

Upon varying the form of the functions $f_l(\cdot)$, this formulation can accommodate a wide range of models

- Standard regression
  - $f_l(\cdot) =$ NULL
- Hierarchical models
  - $f_l(\cdot) \sim$ Normal$(0, \sigma^2_f)$ (Exchangeable)
    $\sigma^2_f \mid \psi \sim$ some common distribution
- Spatial and spatio-temporal models
  - Two components: $f_1(\cdot) \sim$ CAR (Spatially structured effects)
    $f_2(\cdot) \sim$ Normal$(0, \sigma^2_{f_2})$ (Unstructured residual)

LGMs as a general framework

Upon varying the form of the functions $f_l(\cdot)$, this formulation can accommodate a wide range of models

- Standard regression
  - $f_l(\cdot) = $ NULL
- Hierarchical models
  - $f_l(\cdot) \sim$ Normal$(0, \sigma_f^2)$ (Exchangeable)
    $\sigma_f^2 \mid \psi \sim$ some common distribution
- Spatial and spatio-temporal models
  - Two components: $f_1(\cdot) \sim$ CAR (Spatially structured effects)
    $f_2(\cdot) \sim$ Normal$(0, \sigma_{f_2}^2)$ (Unstructured residual)
- Spline smoothing
  - $f_l(\cdot) \sim$ AR$(\phi, \sigma_\varepsilon^2)$

LGMs as a general framework

Upon varying the form of the functions $f_l(\cdot)$, this formulation can accommodate a wide range of models

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

- Hierarchical models
  - $f_l(\cdot) \sim \text{Normal}(0, \sigma^2_f)$ (Exchangeable)
    $\sigma^2_f \mid \boldsymbol\psi \sim$ some common distribution

- Spatial and spatio-temporal models
  - Two components: $f_1(\cdot) \sim \text{CAR}$ (Spatially structured effects)
    $f_2(\cdot) \sim \text{Normal}(0, \sigma^2_{f_2})$ (Unstructured residual)

- Spline smoothing
  - $f_l(\cdot) \sim \text{AR}(\phi, \sigma^2_\varepsilon)$

- Survival models / logGaussian Cox Processes
  - More complex specification in theory, but relatively easy to fit within the INLA framework!

LGMs as a general framework

Upon varying the form of the functions $f_l(\cdot)$, this formulation can accommodate a wide range of models

- Standard regression
  - $f_l(\cdot) = \text{NULL}$
- Hierarchical models
  - $f_l(\cdot) \sim \text{Normal}(0, \sigma^2_f)$ (Exchangeable)
    $\sigma^2_f \mid \psi \sim$ some common distribution
- Spatial and spatio-temporal models
  - Two components: $f_1(\cdot) \sim \text{CAR}$ (Spatially structured effects)
    $f_2(\cdot) \sim \text{Normal}(0, \sigma^2_{f_2})$ (Unstructured residual)
- Spline smoothing
  - $f_l(\cdot) \sim \text{AR}(\phi, \sigma^2_\varepsilon)$
- Survival models / logGaussian Cox Processes
  - More complex specification in theory, but relatively easy to fit within the INLA framework!
- ...

Integrated Nested Laplace Approximation (INLA)

- In a Bayesian LGM, the required distributions are

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

Integrated Nested Laplace Approximation (INLA)

- In a Bayesian LGM, the required distributions are

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

- Estimate

$$p(\boldsymbol{\psi} \mid \boldsymbol{y}) = \frac{p(\boldsymbol{\theta}, \boldsymbol{\psi} \mid \boldsymbol{y})}{p(\boldsymbol{\theta} \mid \boldsymbol{\psi}, \boldsymbol{y})} \approx \left. \frac{p(\boldsymbol{\psi}) p(\boldsymbol{\theta} \mid \boldsymbol{\psi}) p(\boldsymbol{y} \mid \boldsymbol{\theta})}{\tilde{p}(\boldsymbol{\theta} \mid \boldsymbol{\psi}, \boldsymbol{y})} \right|_{\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}(\boldsymbol{\psi})}$$

Integrated Nested Laplace Approximation (INLA)

- In a Bayesian LGM, the required distributions are

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

- Estimate

$$p(\boldsymbol{\psi} \mid \boldsymbol{y}) = \frac{p(\boldsymbol{\theta}, \boldsymbol{\psi} \mid \boldsymbol{y})}{p(\boldsymbol{\theta} \mid \boldsymbol{\psi}, \boldsymbol{y})} \approx \left. \frac{p(\boldsymbol{\psi}) p(\boldsymbol{\theta} \mid \boldsymbol{\psi}) p(\boldsymbol{y} \mid \boldsymbol{\theta})}{\tilde{p}(\boldsymbol{\theta} \mid \boldsymbol{\psi}, \boldsymbol{y})} \right|_{\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}(\boldsymbol{\psi})}$$

$$p(\theta_j \mid \boldsymbol{\psi}, \boldsymbol{y}) = \frac{p\left(\{\theta_j, \boldsymbol{\theta}_{-j}\} \mid \boldsymbol{\psi}, \boldsymbol{y}\right)}{p(\boldsymbol{\theta}_{-j} \mid \theta_j, \boldsymbol{\psi}, \boldsymbol{y})} \approx \left. \frac{p(\boldsymbol{\psi}) p(\boldsymbol{\theta} \mid \boldsymbol{\psi}) p(\boldsymbol{y} \mid \boldsymbol{\theta})}{\tilde{p}(\boldsymbol{\theta}_{-j} \mid \theta_j, \boldsymbol{\psi}, \boldsymbol{y})} \right|_{\boldsymbol{\theta}_{-j} = \hat{\boldsymbol{\theta}}_{-j}(\theta_j, \boldsymbol{\psi})}$$

where $\tilde{p}$ indicates the Laplace approximation and $\hat{\boldsymbol{\theta}}$ is the mode

  - Can do various forms of LA: "Simplified" (based on Taylor's expansion up to 3rd order) vs "Full" (more precise but more computationally expensive)

Integrated Nested Laplace Approximation (INLA)

- In a Bayesian LGM, the required distributions are

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

- Estimate

$$p(\boldsymbol{\psi} \mid \boldsymbol{y}) = \frac{p(\boldsymbol{\theta}, \boldsymbol{\psi} \mid \boldsymbol{y})}{p(\boldsymbol{\theta} \mid \boldsymbol{\psi}, \boldsymbol{y})} \approx \left. \frac{p(\boldsymbol{\psi}) p(\boldsymbol{\theta} \mid \boldsymbol{\psi}) p(\boldsymbol{y} \mid \boldsymbol{\theta})}{\tilde{p}(\boldsymbol{\theta} \mid \boldsymbol{\psi}, \boldsymbol{y})} \right|_{\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}(\boldsymbol{\psi})}$$

$$p(\theta_j \mid \boldsymbol{\psi}, \boldsymbol{y}) = \frac{p\left(\{\theta_j, \boldsymbol{\theta}_{-j}\} \mid \boldsymbol{\psi}, \boldsymbol{y}\right)}{p(\boldsymbol{\theta}_{-j} \mid \theta_j, \boldsymbol{\psi}, \boldsymbol{y})} \approx \left. \frac{p(\boldsymbol{\psi}) p(\boldsymbol{\theta} \mid \boldsymbol{\psi}) p(\boldsymbol{y} \mid \boldsymbol{\theta})}{\tilde{p}(\boldsymbol{\theta}_{-j} \mid \theta_j, \boldsymbol{\psi}, \boldsymbol{y})} \right|_{\boldsymbol{\theta}_{-j} = \hat{\boldsymbol{\theta}}_{-j}(\theta_j, \boldsymbol{\psi})}$$

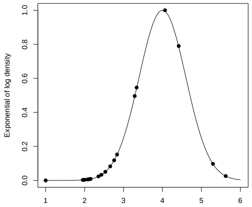
  where $\tilde{p}$ indicates the Laplace approximation and $\hat{\boldsymbol{\theta}}$ is the mode
  - Can do various forms of LA: "Simplified" (based on Taylor's expansion up to 3rd order) vs "Full" (more precise but more computationally expensive)

- Use numerical integration to obtain the marginals

INLA in practice

1. Select a grid of points for ψ_h^* and associated area weights Δ_h & interpolate to approximate the posterior

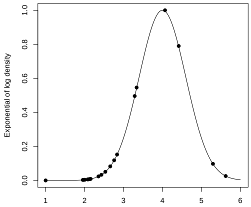
Posterior marginal for ψ : $p(\psi \mid \boldsymbol{y}) \propto \frac{p(\theta,\boldsymbol{y}|\psi)p(\psi)}{p(\theta|\boldsymbol{y},\psi)}$



INLA in practice

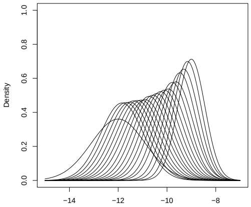
1. Select a grid of points for ψ_h^* and associated area weights Δ_h & interpolate to approximate the posterior

Posterior marginal for ψ : $p(\psi \mid \boldsymbol{y}) \propto \frac{p(\theta,\boldsymbol{y}|\psi)p(\psi)}{p(\theta|\boldsymbol{y},\psi)}$



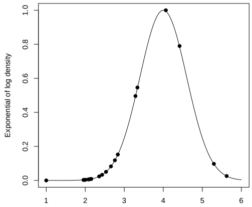
2. Weight the (conditional) marginal posteriors by the density associated with each ψ on the grid

Posterior marginal for θ, conditional on each $\{\psi_h^*\}$ (unweighted)



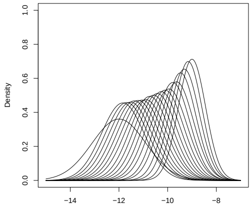
1. Select a grid of points for ψ^*_h and associated area weights Δ_h & interpolate to approximate the posterior

Posterior marginal for ψ : $p(\psi \mid \boldsymbol{y}) \propto \dfrac{p(\theta,\boldsymbol{y}\mid\psi)p(\psi)}{p(\theta\mid\boldsymbol{y},\psi)}$



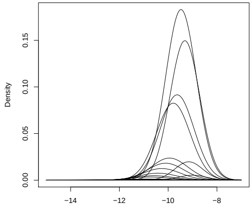
2. Weight the (conditional) marginal posteriors by the density associated with each ψ on the grid

Posterior marginal for θ, conditional on each $\{\psi^*_h\}$ (unweighted)



3. Weight the (conditional) marginal posteriors by the density associated with each ψ on the grid

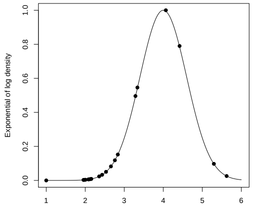
Posterior marginal for θ, conditional on each $\{\psi^*_h\}$ (weighted)



INLA in practice

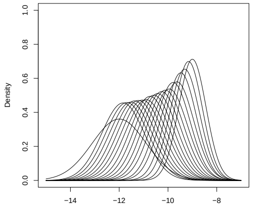
1. Select a grid of points for ψ^*_h and associated area weights Δ_h & interpolate to approximate the posterior

Posterior marginal for ψ : $p(\psi \mid \boldsymbol{y}) \propto \frac{p(\theta,\boldsymbol{y}|\psi)p(\psi)}{p(\theta|\boldsymbol{y},\psi)}$



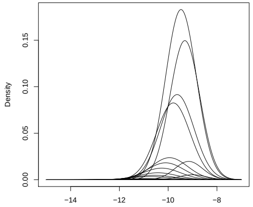
2. Weight the (conditional) marginal posteriors by the density associated with each ψ on the grid

Posterior marginal for θ, conditional on each $\{\psi^*_h\}$ (unweighted)



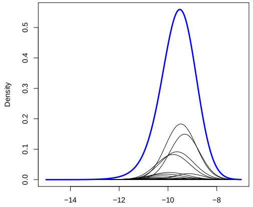
3. Weight the (conditional) marginal posteriors by the density associated with each ψ on the grid

Posterior marginal for θ, conditional on each $\{\psi^*_h\}$ (weighted)



4. (Numerically) sum over the conditional densities to get the marginal posterior for θ

Posterior marginal for θ : $p(\theta \mid \boldsymbol{y})$



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

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

where

- $\boldsymbol{x} = (x_1, x_2)$ are observed covariates for which we are assuming a linear effect on some function $g(\cdot)$ of the parameter θ_i
- $\boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2) \sim \text{Normal}(0, \tau_1^{-1})$ are unstructured ("fixed") effects
- z is an **index**. This can be used to include structured ("random"), spatial, spatio-temporal effect, etc.
- $f \sim \text{Normal}(0, \boldsymbol{Q}_f^{-1}(\tau_2))$ is a suitable function used to model the 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

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

where

- $\boldsymbol{x} = (x_1, x_2)$ are observed covariates for which we are assuming a linear effect on some function $g(\cdot)$ of the parameter θ_i
- $\boldsymbol{\beta} = (\beta_0, \beta_1, \beta_2) \sim \text{Normal}(0, \tau_1^{-1})$ are unstructured ("fixed") effects
- z is an **index**. This can be used to include structured ("random"), spatial, spatio-temporal effect, etc.
- $f \sim \text{Normal}(0, \boldsymbol{Q}_f^{-1}(\tau_2))$ is a suitable function used to model the 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, or lmer)

```
# Install the R-INLA package (see http://www.r-inla.org/)
INLA.repo="https://inla.r-inla-download.org/R/stable"
install.packages("INLA",dep=TRUE,repos=c(getOption("repos"),INLA=INLA.repo))

# Define a model "formula" (as you would in (g)lm)
formula = y ~ x1 + x2 + f(z, model=...)
```

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, or lmer)

```r
# Install the R-INLA package (see http://www.r-inla.org/)
INLA.repo="https://inla.r-inla-download.org/R/stable"
install.packages("INLA",dep=TRUE,repos=c(getOption("repos"),INLA=INLA.repo))

# Define a model "formula" (as you would in (g)lm)
formula = y ~ x1 + x2 + f(z, model=...)
```

- The f() function can account for several structured effects
- This is done by specifying a different model
  - iid, iid1d, iid2d, iid3d specify random effects
  - rw1, rw2, ar1 are smooth effect of covariates or time effects
  - seasonal specifies a seasonal effect
  - besag models spatially structured effects (CAR)
  - generic is a user-defined precision matrix

Step by step guide to using R-INLA

2. Call the function inla, specifying the data and options (more on this later), e.g.

```r
# Calls INLA to fit the model
m = inla(formula, data=data.frame(y,x1,x2,z),...)
```

Step by step guide to using R-INLA

2. Call the function inla, specifying the data and options (more on this later), e.g.

```
# Calls INLA to fit the model
m = inla(formula, data=data.frame(y,x1,x2,z),...)
```

- The data need to be included in a suitable data.frame
- R returns an object m in the class inla, which has some methods available
  - summary()
  - plot()
- The options let you specify the priors and hyperpriors, together with additional output

Example

- Logistic regression — data available in the brinla package (https://github.com/julianfaraway/brinla)

```
library(INLA)
# Load the data
data(lowbwt, package = "brinla")
head(lowbwt)

  LOW AGE LWT RACE SMOKE HT UI FTV
1   1  28 120    3     1  0  1   0
2   1  29 130    1     0  0  1   2
3   1  34 187    2     1  1  0   0
4   1  25 105    3     0  1  0   0
5   1  25  85    3     0  0  1   0
6   1  27 150    3     0  0  0   0

# Specify the model
formula = LOW ~ AGE + LWT + RACE + SMOKE + HT + UI + FTV

# Run INLA
m = inla(formula, data=lowbwt, family = "binomial", Ntrials = 1,
                     control.compute = list(dic = TRUE, cpo = TRUE))
```

Example

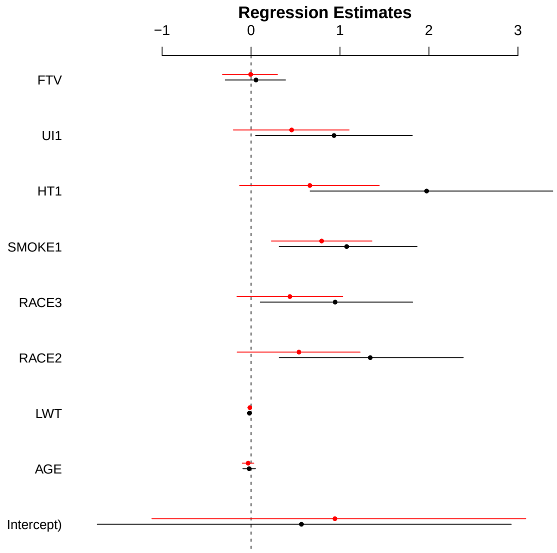
```
summary(m)
Time used:
 Pre-processing    Running inla Post-processing           Total
     0.21849060      0.07591939      0.03839827      0.33280826

Fixed effects:
               mean       sd 0.025quant 0.5quant 0.975quant     mode       kld
(Intercept)  0.56670 1.185563   -1.72706  0.55532   2.924583  0.53281 1.187e-07
AGE         -0.02068 0.035961   -0.09215 -0.02039   0.049074 -0.01980 2.311e-07
LWT         -0.01760 0.006853   -0.03167 -0.01739  -0.004754 -0.01696 1.425e-06
RACE2        1.34018 0.527674    0.31539  1.33639   2.385517  1.32888 3.483e-07
RACE3        0.94550 0.436262    0.10374  0.94057   1.815536  0.93082 3.124e-08
SMOKE1       1.07495 0.395395    0.31524  1.06945   1.866740  1.05857 4.189e-09
HT1          1.97339 0.694087    0.66337  1.95455   3.391165  1.91703 9.834e-07
UI1          0.93286 0.448558    0.05220  0.93284   1.812824  0.93284 9.883e-08
FTV          0.05591 0.171968   -0.28921  0.05852   0.386280  0.06369 3.120e-08

The model has no random effects
The model has no hyperparameters

Expected number of effective parameters(std dev): 8.999(0.00)
Number of equivalent replicates : 21.00

Deviance Information Criterion (DIC) ...............: 221.20
Deviance Information Criterion (DIC, saturated) ....: 221.20
```

Example

- By default, INLA uses "minimally informative" priors for the model parameters. But can modify in various ways
  - "Penalised Complexity" Prior — invariant to reparameterisations & linked to "objective", Jeffreys' priors
  - Standard distributions with fixed parameters

```
# Specify different values for (some of) the prior distributions
priors = list(mean.intercept=0, prec.intercept=10^(-2),
              mean=list(AGE=log(.5), SMOKE1=log(2), default=0), prec=.5^(-2))

# Re-run the model
m2 = inla(formula, data=lowbwt, family = "binomial", Ntrials = 1,
          control.compute = list(dic = TRUE, cpo = TRUE),control.fixed=priors)

# Shows the results
print(m2$summary.fixed,digits=4)

                 mean      sd 0.025quant  0.5quant 0.975quant      mode       kld
(Intercept)  0.942691 1.07101   -1.11482  0.927722   3.087720  0.898064 5.262e-09
AGE         -0.032257 0.03383   -0.09999 -0.031804   0.032901 -0.030904 4.904e-10
LWT         -0.012983 0.00628   -0.02586 -0.012790  -0.001199 -0.012406 1.352e-06
RACE2        0.538491 0.35252   -0.15720  0.539704   1.226768  0.542148 8.830e-07
RACE3        0.436564 0.30300   -0.15892  0.436762   1.030369  0.437185 8.350e-07
SMOKE1       0.793426 0.28788    0.23013  0.792763   1.359874  0.791464 1.013e-06
HT1          0.661096 0.39972   -0.12790  0.662528   1.441378  0.665414 8.627e-07
UI1          0.456288 0.33115   -0.19716  0.457404   1.102905  0.459653 8.929e-07
FTV         -0.004357 0.15674   -0.31955 -0.001754   0.296136  0.003412 8.787e-08
```

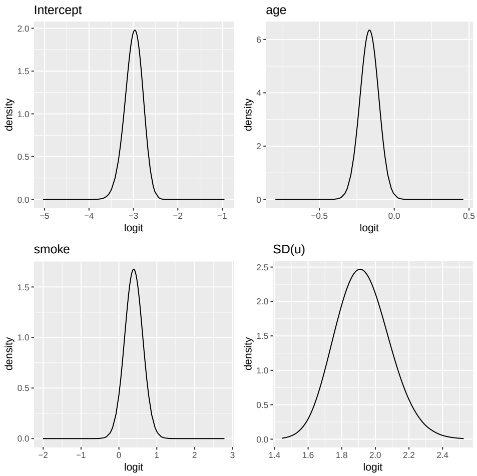
Example



-●- Default prior
-●- Modified prior

Example

```
# Load the data
data(ohio, package = "brinla")
# Specify the model including random effects by individual
formula = resp ~ age + smoke + f(id, model="iid")
# Run INLA
m = inla(formula, family="binomial", data=ohio, control.compute=list(config=TRUE))
```



Simulating from the posterior distributions

- Arguably, one of the main advantages of MCMC is that, given convergence, the output is given by samples from the **joint** posterior distribution of all parameters, $p(\boldsymbol{\theta} \mid y)$
  - Can obtain all marginal distributions $p(\theta_j \mid y)$ by simply selecting the relevant simulations
  - Can obtain simulations from the posterior distribution of any function $g(\theta_j)$ by simply applying the function to the simulations for θ_j

Simulating from the posterior distributions

- Arguably, one of the main advantages of MCMC is that, given convergence, the output is given by samples from the **joint** posterior distribution of all parameters, $p(\boldsymbol{\theta} \mid y)$
  - Can obtain all marginal distributions $p(\theta_j \mid y)$ by simply selecting the relevant simulations
  - Can obtain simulations from the posterior distribution of any function $g(\theta_j)$ by simply applying the function to the simulations for θ_j

- INLA is a bit more complicated
  - Can use Monte Carlo to obtain simulations from the posterior distributions
  - However, because of how it works, the estimates are for the **marginal** posterior distributions for each model parameter
  - Can use specialised functions based on copulæ to approximate the underlying joint posterior and then MC-simulate

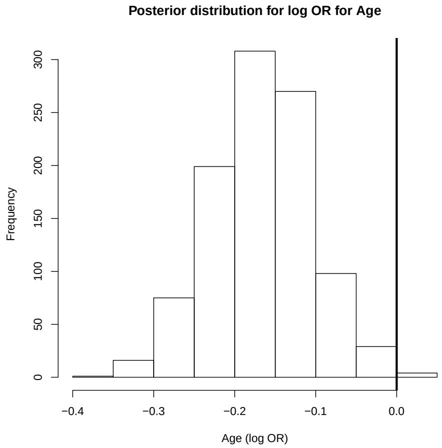
```
# Create an object with the simulations from the joint posterior
jpost = inla.posterior.sample(n=1000,m)

# Selects the positions in the resulting list where the values of the "fixed effects" are stored
pos = pmatch(rownames(m$summary.fixed),rownames(jpost[[1]]$latent))

# Select only the relevant simulated values and put them in a matrix with
# number of rows = nsim and number of columns=length(pos)
sim <- matrix(unlist(lapply(jpost,function(x) x$latent[pos,])),ncol=length(pos),byrow=T)
colnames(sim) <- m$names.fixed
```

Simulating from the posterior distributions

```
# Matrix with simulations from the joint posterior distribution
head(sim)

     (Intercept)        age     smoke
[1,]   -2.927550 -0.2021857 0.2968993
[2,]   -3.196615 -0.1806465 0.7236093
[3,]   -3.006732 -0.1474960 0.4901844
[4,]   -2.926754 -0.2207513 0.2648405
[5,]   -2.909527 -0.1479024 0.4605895
[6,]   -2.796352 -0.1622534 0.2770397

# Posterior probability that the "age effect" exceeds 0 (on logOR scale)
sum(sim[,"age"]>0)/nrow(sim)

[1] 0.004

# Posterior probability that the "smoke effect" exceeds 1 (on OR scale)
sum(exp(sim[,"smoke"])>1)/nrow(sim)

[1] 0.944
```

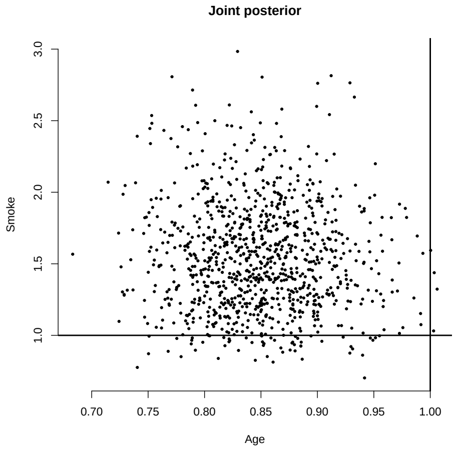
Simulating from the posterior distributions

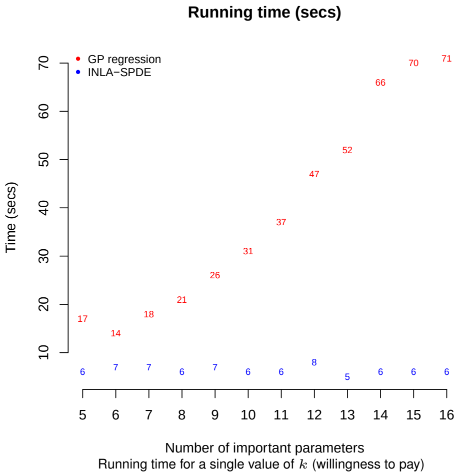
```r
# Histogram for the marginal posterior distribution of Age (logOR scale)
hist(sim[,"age"],xlab="Age (log OR)",main="Posterior distribution for log OR for Age")
abline(v=0,lwd=3)
```



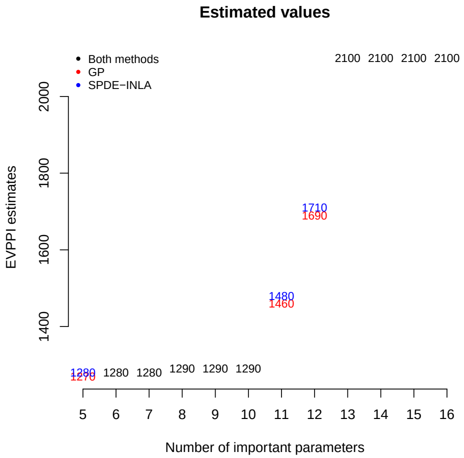
Simulating from the posterior distributions

```
# Scatterplot for the joint posterior distribution of Age & Smoke (OR scale)
plot(exp(sim[,"age"]),exp(sim[,"smoke"]),pch=20,cex=.7,xlab="Age",
     ylab="Smoke",main="Joint posterior",axes=F)
axis(1); axis(2)
abline(v=1,lwd=2); abline(h=1,lwd=2)
```





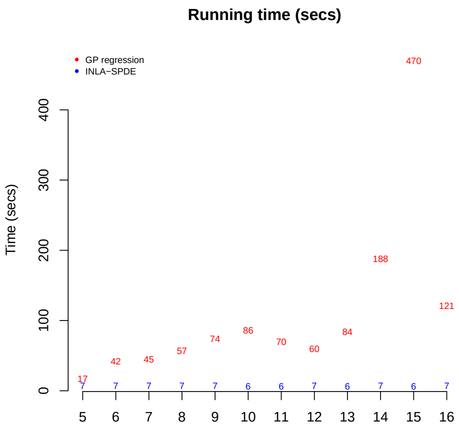
Running time for a single value of k (willingness to pay)



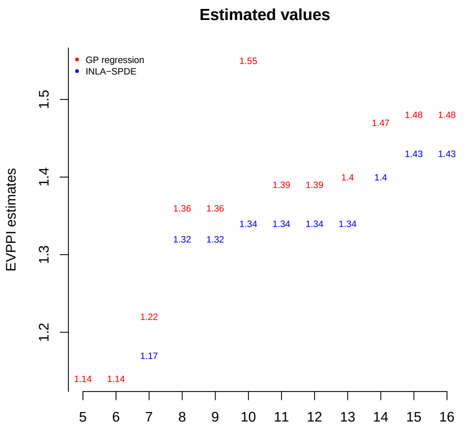
- Fictional decision tree model with correlated parameters
- 2 treatment options and overall 19 parameters
- Parameters simulated from multivariate Normal distribution, so can compute exact EVPPI

Heath et al *Statistics in Medicine*. 2016; **35(23)**: 4264-4280

Using INLA in practice. . .



Number of important parameters
Running time for a single value of k (willingness to pay)



Number of important parameters

- Cost-effectiveness model for influenza vaccine based on evidence synthesis
- 2 treatment options and overall 63 parameters
- Model not available in closed form (needs MCMC simulations)

Heath et al *Statistics in Medicine*. 2016; **35(23)**: 4264-4280

Using INLA in practice. . .

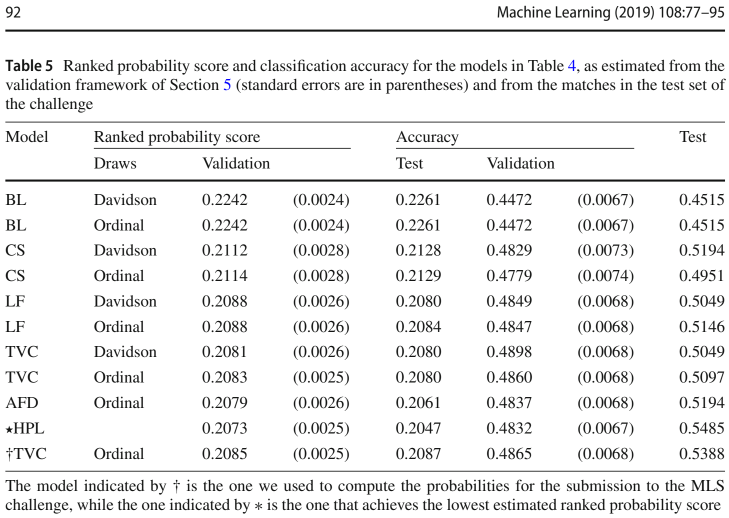


Table 5 Ranked probability score and classification accuracy for the models in Table 4, as estimated from the validation framework of Section 5 (standard errors are in parentheses) and from the matches in the test set of the challenge

Model	Ranked probability score			Accuracy			Test
	Draws	Validation		Test	Validation		
BL	Davidson	0.2242	(0.0024)	0.2261	0.4472	(0.0067)	0.4515
BL	Ordinal	0.2242	(0.0024)	0.2261	0.4472	(0.0067)	0.4515
CS	Davidson	0.2112	(0.0028)	0.2128	0.4829	(0.0073)	0.5194
CS	Ordinal	0.2114	(0.0028)	0.2129	0.4779	(0.0074)	0.4951
LF	Davidson	0.2088	(0.0026)	0.2080	0.4849	(0.0068)	0.5049
LF	Ordinal	0.2088	(0.0026)	0.2084	0.4847	(0.0068)	0.5146
TVC	Davidson	0.2081	(0.0026)	0.2080	0.4898	(0.0068)	0.5049
TVC	Ordinal	0.2083	(0.0025)	0.2080	0.4860	(0.0068)	0.5097
AFD	Ordinal	0.2079	(0.0026)	0.2061	0.4837	(0.0068)	0.5194
⋆HPL		0.2073	(0.0025)	0.2047	0.4832	(0.0067)	0.5485
†TVC	Ordinal	0.2085	(0.0025)	0.2087	0.4865	(0.0068)	0.5388

The model indicated by † is the one we used to compute the probabilities for the submission to the MLS challenge, while the one indicated by $*$ is the one that achieves the lowest estimated ranked probability score

Thank you!