Una primera aproximación a la Estadística Bayesiana

Asunción M. Mayoral y Javier Morales. IUI CIO-UMH

Noviembre 2022

Contents

1 Contexto **5**

1.1 Objetivos de aprendizaje 5

1.2 Contenidos propuestos 6

1.3 Contenidos definitivos 6

1.4 INLA 6

2 Regresión lineal **9**

2.1 Introducción 9

2.2 Variables y relaciones 10

2.3 Verosimilitud 11

2.4 Hiperparámetros 12

2.5 Efectos fijos 13

2.6 Resultados 14

2.7 Distribuciones posteriores 18

2.8 Simulación de la posterior 21

2.9 Regresión lineal múltiple con INLA 24

2.10 Conclusiones 32

3 Modelo de ANOVA **35**

3.1 Introducción 35

3.2 El modelo de ANOVA 35

3.3 Anova de una vía 36

3.4 Anova de varias vías 43

3.5 Análisis de ANCOVA 47
3.6 Efectos aleatorios 57
3.7 Modelos mixtos 64
3.8 Conclusiones 79

4 Modelos lineales generalizados **81**

4.1 Modelos jerárquicos bayesianos 82
4.2 Regresión logística 83
4.3 Regresión de Poisson 91

Chapter 1

Contexto

El curso "Una primera aproximación a la Estadística Bayesiana", con una duración de 10 horas, está ofertado a alumnado de doctorado con una formación científica en el ámbito BIO.

El curso se desarrollará íntegramente online a través de videoconferencia síncrona, durante un total de 10 horas, seccionadas en 4 sesiones de dos horas y media cada una de ellas, los días 8, 10, 15 y 17 de noviembre de 2022, de 16 a 18:30h.

Se presentarán los contenidos a través de ejemplos prácticos programados en R, para que el estudiantado pueda ir generando resultados y comentando las interpretaciones derivadas del análisis.

La evaluación es continua basada en la interacción con el profesorado durante las sesiones de trabajo.

1.1 Objetivos de aprendizaje

- Conocer los conceptos básicos en el planteamiento bayesiano de la Estadística.
- Identificar la relevancia de la información previa y de expertos, y la proporcionada por los datos.
- Conocer los procedimientos básicos para conjugar la información disponible.
- Aplicar los conocimientos básicos en problemas sencillos.
- Descubrir las dificultades computacionales en la inferencia bayesiana.

1.2 Contenidos propuestos

1. De probabilidad va la historia: la relevancia de las probabilidades condicionadas y el teorema de Bayes.
2. La jerga base: incertidumbre, a priori, a posteriori y verosímil.
3. Manos en la masa 1: ¿con qué probabilidad ocurrió?
4. Manos en la masa 2: ¿con qué abundancia ocurrió?
5. Curioseando para saber más: manuales y software.

1.3 Contenidos definitivos

1. SESIÓN 1: Probabilidades, Bayes y las proporciones.
2. SESIÓN 2: INLA y la regresión.
3. SESIÓN 3: INLA y el ANOVA.
4. SESIÓN 4: INLA y los GLM.

1.4 INLA

INLA es una librería de R que aproxima la inferencia Bayesiana para modelos gausianos latentes (LGM). Sus siglas provienen de Integrated Nested Laplace Approximation (INLA), que es un método para aproximar las inferencias bayesianas a través de la aproximación de Laplace.

Aunque la metodología INLA se ha desarrollado sobre modelos que se pueden expresar como campos aleatorios markovianos gausianos (*Gaussian Markov random fields, GMRF*), es viable para una gran familia de modelos habituales en la práctica estadística.

Disponemos de referencias múltiples y documentación de esta librería en la web r-inla.org, y en particular en el manual de referencia de Gómez-Rubio (2021) titulado Bayesian inference with INLA, también publicado por Chapman & Hall-CRC Press.

1.4.1 Instalación

Para instalar la librería INLA hemos de ejecutar, desde R, el comando

```
install.packages("INLA", repos=c(getOption("repos"),
                INLA="https://inla.r-inla-download.org/R/stable"),
                dep=TRUE)
# y a continuación la cargamos con:
library(INLA)
```

Para instalar actualizaciones, basta con ejecutar

```
options(repos = c(getOption("repos"),
                  INLA="https://inla.r-inla-download.org/R/testing"))
update.packages("INLA", dep=TRUE)
```

Las descargas y documentación completa sobre INLA está disponible en R-INLA home.

Ya desde R, para pedir ayuda sobre funciones en INLA, basta usar el comando `inla.doc()`, especificando dentro y entrecomillada, la función/objeto sobre el que se solicita ayuda. Por ejemplo, `inla.doc("ar1")` o `inla.doc("loggamma")`.

También utilizaremos una librería accesoria de INLA, `brinla`, desarrollada por Faraway, Yue y Wan, 2022.

```
install.packages("remotes")
remotes::install_github("julianfaraway/brinla")
```

1.4.2 Fundamentos

INLA está basado en la resolución de integrales vía la aproximación de Laplace, que aproxima el integrando a través de una expansión de Taylor de segundo grado que permite calcular la integral analíticamente.

$$\begin{aligned} I_n &= \int_x exp[nf(x)]dx \\ &\approx \int_x exp[n(f(x_0) + 1/2(x - x_0)^2 f''(x_0))]dx \\ &= exp[nf(x_0)] \cdot \sqrt{\frac{2\pi}{-nf''(x_0)}} \end{aligned}$$

Evita así los largos tiempos de simulación de las cadenas de Markov Monte Carlo. Cuando las distribuciones a integrar son Gausianas, Laplace da órdenes buenos de aproximación. Y este es el principio que usa para modelizar la mayoría de los modelos habituales, que se integran dentro de la amplia clase de los modelos gausianos latentes, en los que se aplica INLA.

INLA se ha aplicado en mapeo estadístico, modelos de cohorte multidimensionales, modelos de asociación espacial, genética, análisis medioambientales, salud y epidemiología, dinámicas de infecciones, estudios agronómicos, meta-análisis, impacto del cambio climático y muchos más ámbitos (Rue et al, 2017).

Chapter 2

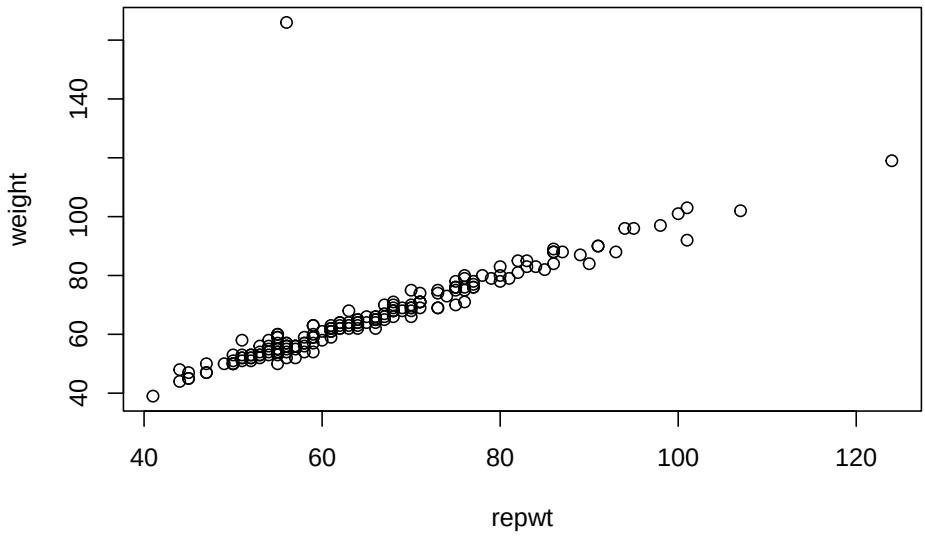
Regresión lineal

2.1 Introducción

Una vez presentados los fundamentos de INLA vamos a utilizarlo para trabajar progresivamente desde los modelos más sencillos a los más sofisticados. Empezamos aquí con el modelo de regresión lineal simple, para continuar generalizando con el de regresión lineal múltiple.

Partimos de la base de datos Davis (en la librería `carData`), que contiene 200 registros de 5 variables relacionadas con un estudio sobre habituación de hombres y mujeres a la realización de ejercicio físico de forma regular. Las variables que se registraron son sexo, peso y altura (reales y reportados). Vamos a indagar la relación entre el peso real (`weight`) y el reportado (`repwt`) a través de un análisis de regresión lineal simple.

```
data(Davis,package="carData")
plot(weight ~ repwt, data=Davis)
```




```
# Excluimos el valor outlier y los NA
davis=Davis %>%
  filter(weight<160) %>%
  slice(which(!is.na(repwt)))
```

2.2 Variables y relaciones

Entendemos como variable respuesta `y=weight`, de tipo numérico (continua), y como variable explicativa o covariable, `x=repwt`, también numérica.

La especificación de respuesta y y predictores $x_1, x_2, ...$, en INLA se registra en una fórmula del tipo:

```
formula= y ~ 1 + x_1 + x_2 +...
```

en la que podemos prescindir del 1 que identifica la interceptación, pues el ajuste, por defecto, siempre se resolverá con su estimación, salvo que en su lugar se escriba un '-1'.

En nuestro problema tendríamos pues:

```
formula = weight ~ repwt
```

A continuación es procedente elegir el modelo sobre la respuesta, o lo que es lo mismo, la verosimilitud.

2.3 Verosimilitud

En principio es razonable asumir normalidad en la respuesta, además de independencia entre todas las observaciones. Así, el modelo propuesto para la respuesta es:

$$(y_i|\mu_i, \sigma^2) \sim N(\mu_i, \sigma^2), i = 1, ..., n$$

donde `y=weight` y `x=repwt`, e $i = 1, ..., n$ es un subíndice que identifica a cada uno de los registros disponibles en el banco de datos. La media esperada contiene la relación lineal entre covariable y respuesta, $\mu_i = \theta + \beta x_i$, donde θ es la interceptación de la recta de regresión y β el coeficiente que explica la relación lineal entre x e y. El vector (θ, β) identifica los **efectos latentes**, en cuya inferencia posterior estamos interesados, y que están involucrados directamente en la media o predictor lineal. El modelo, o lo que es lo mismo, la verosimilitud, depende también de un parámetro de dispersión σ^2 sobre el que también querremos inferir.

Veamos cómo especificar este modelo con INLA. La función `names(inla.models())` proporciona un listado de todos los tipos de modelos posibles, tanto para los datos (`likelihood`), para los efectos latentes (`latent`), los parámetros (`prior`), y otras opciones que de momento no nos interesan. El listado completo de todas las distribuciones disponibles para cada uno de los tipos de modelos lo obtenemos con el comando `inla.list.models()`. En particular, si ejecutamos `names(inla.models()$likelihood)`, obtenemos todas las distribuciones disponibles para modelizar los datos.

La distribución `gaussian` identifica la distribución normal que hemos planteado en nuestro modelo de regresión lineal. Para obtener información sobre cómo está parametrizada y cuáles son las priors por defecto, basta consultar la documentación *gaussian* con el comando:

```
# documentación (parametrización y priors) del modelo normal
inla.doc("gaussian")
```

Para ajustar un modelo sencillo en INLA hay que echar mano de la función `inla`, en la que introducimos en primer lugar la `formula`, con la relación entre las variables, a continuación el modelo, en el argumento `family`, y después el banco de datos sobre el que trabajamos. Si no especificamos el argumento `family`, la función `inla` interpreta por defecto la opción `gaussian`, esto es, normalidad para los datos, de modo que podríamos excluir dicha especificación cuando modelizamos datos normales.

```
# Asumiendo datos normales
fit=inla(formula,family="gaussian", data)
```

```
# equivalente a
fit=inla(formula, data)
```

Adelantamos pues un paso más en nuestro problema, añadiendo la verosimilitud normal y la base de datos.

```
# ajuste del modelo
formula = weight ~ 1+ repwt
fit=inla(formula,family="gaussian",data=davis)
```

2.4 Hiperparámetros

INLA identifica como hiperparámetros todos aquellos parámetros en el modelo que no se corresponden con efectos latentes, esto es, relacionados con el predictor o respuesta esperada. En nuestro modelo, el único hiperparámetro es la varianza σ^2, sobre la que es preciso especificar una distribución a priori. Para la varianza σ^2 es habitual asumir una gamma inversa difusa, con media y varianza grandes.

En INLA, en lugar de asignar distribuciones a priori sobre las varianzas, se hace sobre el logaritmo de las precisiones, para facilitar el cálculo del máximo de la log-posterior (obtenida de la log-verosimilitud y la log-prior). Así, asumir una gamma inversa difusa $GaI(\alpha, \beta)$ para la varianza es equivalente a una Gamma difusa $Ga(\alpha, \beta)$ para la precisión $\tau = 1/\sigma^2$, y una log-gamma difusa $Log - Gamma(\alpha, \beta)$ para la log-precisión $log(\tau)$.

Por defecto, como ya verificamos en la documentación de la verosimilitud gaussiana, (con `inla.doc("gaussian")`), la distribución a priori por defecto sobre la log-precisión (θ_1 en la ayuda) es la log-gamma difusa $LogGa(1, 5 \cdot 10^{-5})$, lo que da un valor esperado para la precisión τ de 20000 y una varianza de $4 \cdot 10^8$.

Para modificar la distribución a priori de un parámetro podemos utilizar cualquiera de las distribuciones que ofrece INLA en su listado `names(inla.models()$prior)` (siempre buscando coherencia con la información sobre el parámetro en cuestión).

Para definir una prior para los parámetros o hiperparámetros en INLA hay que definir los siguientes argumentos:

- prior, el nombre de la distribución a priori (para hiperparámetros, alguna de las opciones en `names(inla.models()$prior)`)
- param, los valores de los parámetros de la prior
- initial, el valor inicial para el hiperparámetro
- fixed, variable booleana para decir si el hiperparámetro es fijo o aleatorio.

La modificación la haremos con el argumento `control.family=list(hyper=list(...))` en la función `inla`, al que le proporcionaremos una lista con el nombre de los parámetros (el *short name* con el que los identifica INLA en la documentación), que apunta a una lista con la distribución (*prior*) y los parámetros (*param*) a utilizar.

En nuestro problema, si tenemos información previa sobre la precisión del modelo, reconocida como `prec` (short name), y queremos especificar una a priori $Ga(1, 0.001)$ para la precisión, habremos de utilizar la siguiente sintaxis:

```
prec.info = list(prior="loggamma", param =c(1,0.001))
fit2=inla(formula,family="gaussian",data=davis,
      control.family = list(hyper = list(prec = prec.info)))
```

Si nos conformamos con la previa por defecto de INLA, el modelo que estamos asumiendo en nuestro problema de regresión lineal simple será:

$$\begin{aligned} (y_i|\mu_i, \sigma^2) &\sim N(\mu_i, \sigma^2), i = 1, ..., n \\ \tau = 1/\sigma^2 &\sim Ga(1, 0.00005) \end{aligned}$$

2.5 Efectos fijos

Una variable explicativa entra en el modelo como **efecto fijo** cuando se piensa que afecta a todas las observaciones del mismo modo, y que su efecto es de interés primario en el estudio.

En un modelo de regresión lineal todos los efectos latentes en el predictor lineal, interceptación y coeficientes de covariables, son efectos fijos. Ante ausencia de información, las priors para los efectos fijos, esto es, (θ, β) en nuestro caso, se asumen normales centradas en cero y con varianzas grandes. En INLA la interceptación β tiene por defecto una prior gausiana con media y precisión igual a cero (una distribución plana objetiva, que no integra 1), y los coeficientes β también tienen una prior gausiana con media cero y precisión igual a 0.001. Estos valores por defecto se pueden consultar con el comando `inla.set.control.fixed.default()`, que da la siguiente información:

- *mean=0* y *mean.intercept=0* son las medias de la distribución normal para los coeficientes β y la interceptación θ respectivamente
- *prec=0.001* y *prec.intercept=0* son las precisiones respectivas de las normales para β y θ.

Con todo, los parámetros de las priors sobre los efectos fijos se pueden modificar a través del argumento `control.fixed=list(...)` en la función `inla`,

utilizando siempre los nombres que atribuye INLA a los diferentes parámetros e hiperparámetros (*short name*). Por ejemplo, si queremos modificar la precisión de los efectos fijos para hacerla igual a 0.001 (esto es, varianza 1000), utilizamos la siguiente sintaxis:

```
formula = weight ~ 1+ repwt
fit=inla(formula,family="gaussian",data=davis,
         control.fixed=list(prec=0.001,prec.intercept=0.001))
```

Volvemos sobre nuestro ejemplo, y asumiendo las a priori por defecto de INLA tendremos:

$$\begin{aligned}(y_i|\theta,\beta,\sigma^2) &\sim N(\theta+\beta x_i,\sigma^2), i=1,...,n\\ \theta &\sim N(0,\infty)\\ \beta &\sim N(0,1000)\\ \tau=1/\sigma^2 &\sim Ga(1,0.00005)\end{aligned}$$

2.6 Resultados

Para mostrar una descriptiva de los resultados del ajuste obtenido con `fit=inla(...)`, utilizamos la sintaxis siguiente:

- `summary(fit)` proporciona una descriptiva del ajuste
- `names(fit$marginals.fixed)` lista los nombres de todos los efectos fijos
- `fit$summary.fixed` resume la inferencia posterior sobre los efectos fijos
- `names(fit$marginals.hyperpar)` lista los nombres de todos los hiperparámetros
- `fit$summary.hyperpar` da un resumen de la inferencia posterior de los parámetros e hiperparámetros
- `fit$summary.fitted.values` resume la inferencia posterior sobre los valores ajustados
- `fit$mlik` da la estimación de la log-verosimilitud marginal, útil para evaluar y comparar modelos.

Veamos los resultados para nuestro problema de regresión.

```
# ajuste del modelo
formula = weight ~ 1+ repwt
fit=inla(formula,family="gaussian",data=davis)
summary(fit)
#>
#> Call:
```

```
#>    c("inla.core(formula = formula, family = family,
#>    contrasts = contrasts, ", " data = data, quantiles =
#>    quantiles, E = E, offset = offset, ", " scale =
#>    scale, weights = weights, Ntrials = Ntrials, strata =
#>    strata, ", " lp.scale = lp.scale, link.covariates =
#>    link.covariates, verbose = verbose, ", " lincomb =
#>    lincomb, selection = selection, control.compute =
#>    control.compute, ", " control.predictor =
#>    control.predictor, control.family = control.family,
#>    ", " control.inla = control.inla, control.fixed =
#>    control.fixed, ", " control.mode = control.mode,
#>    control.expert = control.expert, ", " control.hazard
#>    = control.hazard, control.lincomb = control.lincomb,
#>    ", " control.update = control.update,
#>    control.lp.scale = control.lp.scale, ", "
#>    control.pardiso = control.pardiso, only.hyperparam =
#>    only.hyperparam, ", " inla.call = inla.call, inla.arg
#>    = inla.arg, num.threads = num.threads, ", "
#>    blas.num.threads = blas.num.threads, keep = keep,
#>    working.directory = working.directory, ", " silent =
#>    silent, inla.mode = inla.mode, safe = FALSE, debug =
#>    debug, ", " .parent.frame = .parent.frame)")
#> Time used:
#>     Pre = 2.45, Running = 0.183, Post = 0.019, Total = 2.66
#> Fixed effects:
#>              mean    sd 0.025quant 0.5quant 0.975quant mode
#> (Intercept) 2.734 0.814      1.135    2.734      4.333   NA
#> repwt       0.958 0.012      0.935    0.958      0.982   NA
#>             kld
#> (Intercept)   0
#> repwt         0
#>
#> Model hyperparameters:
#>                                           mean    sd
#> Precision for the Gaussian observations 0.199 0.021
#>                                         0.025quant 0.5quant
#> Precision for the Gaussian observations      0.161    0.198
#>                                         0.975quant mode
#> Precision for the Gaussian observations      0.242   NA
#>
#> Marginal log-Likelihood:  -426.72
#>  is computed
#> Posterior summaries for the linear predictor and the fitted values are computed
#> (Posterior marginals needs also 'control.compute=list(return.marginals.predictor=TRUE)')
fit$mlik
```

```
#>                                                [,1]
#> log marginal-likelihood (integration) -426.7295
#> log marginal-likelihood (Gaussian)    -426.7159
```

Obtenemos pues la salida con un descriptivo de la distribución posterior para los efectos fijos interceptación, θ, y el coeficiente del regresor `repwt`, β, con la media, desviación típica y cuantiles con los que podemos evaluar la región creíble al 95%.

También muestra a continuación una tabla con los descriptivos de la distribución posterior para la precisión $\tau = 1/\sigma^2$ de los datos.

Si queremos obtener los nombres con los que INLA reconoce los efectos fijos (interceptación y coeficiente del regresor) e hiperparámetros (precisión de los datos), llamamos a

```
names(fit$marginals.fixed)
#> [1] "(Intercept)" "repwt"
names(fit$marginals.hyperpar)
#> [1] "Precision for the Gaussian observations"
```

Y podemos pedir descriptivos específicos de las distribuciones de los efectos fijos y de los hiperparámetros.

```
# descriptivos efectos fijos
fit$summary.fixed
#>                   mean         sd 0.025quant  0.5quant
#> (Intercept) 2.7338110 0.81444704  1.1347636 2.7338109
#> repwt       0.9583742 0.01213756  0.9345438 0.9583742
#>             0.975quant mode          kld
#> (Intercept)  4.3328591   NA 6.375081e-10
#> repwt        0.9822045   NA 6.375718e-10
# descriptivos varianza
fit$summary.hyperpar
#>                                                mean         sd
#> Precision for the Gaussian observations 0.199083 0.02088159
#>                                                0.025quant
#> Precision for the Gaussian observations  0.1607375
#>                                                 0.5quant
#> Precision for the Gaussian observations 0.1983754
#>                                                0.975quant mode
#> Precision for the Gaussian observations  0.2417988   NA
# medias de los efectos fijos
fit$summary.fixed$mean
#> [1] 2.7338110 0.9583742
```

```
# descriptivos primer efecto fijo
fit$summary.fixed[1,]
#>                  mean       sd 0.025quant 0.5quant
#> (Intercept) 2.733811 0.814447   1.134764 2.733811
#>              0.975quant mode          kld
#> (Intercept)   4.332859   NA 6.375081e-10
```

Para describir la marginal posterior sobre cada uno de los datos ajustados:

```
head(fit$summary.fitted.values)
#>                            mean        sd 0.025quant 0.5quant
#> fitted.Predictor.001 76.52862 0.2162862   76.10402 76.52862
#> fitted.Predictor.002 51.61089 0.2441680   51.13155 51.61089
#> fitted.Predictor.003 54.48602 0.2190235   54.05604 54.48602
#> fitted.Predictor.004 69.82000 0.1750487   69.47635 69.82000
#> fitted.Predictor.005 59.27789 0.1856153   58.91349 59.27789
#> fitted.Predictor.006 75.57025 0.2087832   75.16037 75.57025
#>                       0.975quant mode
#> fitted.Predictor.001   76.95323   NA
#> fitted.Predictor.002   52.09024   NA
#> fitted.Predictor.003   54.91599   NA
#> fitted.Predictor.004   70.16365   NA
#> fitted.Predictor.005   59.64228   NA
#> fitted.Predictor.006   75.98012   NA
```

Si queremos hacer un **análisis de sensibilidad** sobre las distribuciones a priori, reajustamos el modelo con otras priors y comparamos los resultados.

```
# ajuste del modelo
formula = weight ~ 1+ repwt
fit2=inla(formula,family="gaussian",data=davis,
          control.fixed = list(mean.intercept = 100,
                               prec.intercept = 0.001,
                               prec = 0.001),
             control.family = list(hyper = list(
               prec = list(prior="loggamma", param =c(1,0.001)))))
fit2$summary.fixed
#>                   mean         sd 0.025quant  0.5quant
#> (Intercept) 2.7982706 0.81407251  1.2009560 2.7979292
#> repwt       0.9574339 0.01213213  0.9336006 0.9574389
#>              0.975quant mode          kld
#> (Intercept)   4.397522   NA 6.249540e-10
#> repwt         0.981239   NA 6.245646e-10
fit2$summary.hyperpar
```

```
#>                                                 mean
#> Precision for the Gaussian observations 0.1990996
#>                                                   sd
#> Precision for the Gaussian observations 0.02085016
#>                                            0.025quant
#> Precision for the Gaussian observations  0.1609806
#>                                             0.5quant
#> Precision for the Gaussian observations 0.1983598
#>                                            0.975quant mode
#> Precision for the Gaussian observations  0.2416834   NA
```

2.7 Distribuciones posteriores

Para obtener la distribución marginal de los valores ajustados y predichos necesitamos incorporar a la función `inla` el argumento `control.compute=list(return.marginals.predictor`

Tras conseguir un ajuste con `inla`, podemos acceder a todas las distribuciones marginales posteriores y predictivas a través de:

- `fit$marginals.fixed` da las distribuciones posteriores marginales de los efectos fijos
- `fit$marginals.fixed$xx` da la distribución del efecto fijo `xx`, y también se puede seleccionar con su ordinal en el conjunto de efectos fijos `fit$marginals.fixed[[i]]`
- `fit.marginals.hyperpar` da las distribuciones posteriores marginales de los parámetros e hiperparámetros
- `fit$marginals.fitted.values` da las distribuciones posteriores marginales para los valores ajustados

Con estas distribuciones, reconocidas como `marginal`, podemos hacer cálculos y gráficos de interés a través de estas funciones que operan sobre las distribuciones y que podemos consultar con `?inla.marginal`:

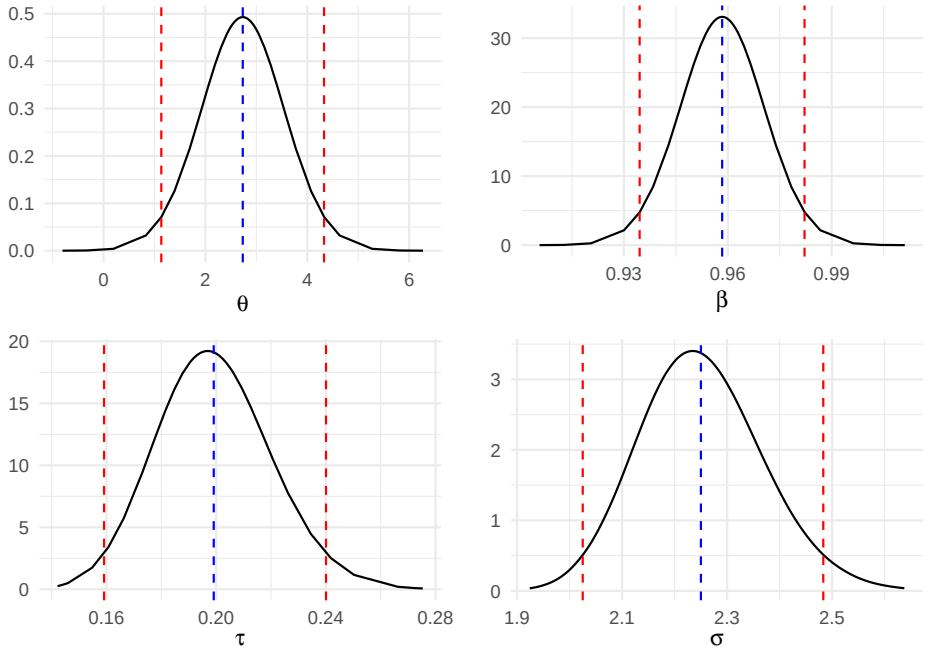
- `inla.dmarginal(x, marginal, ...)` da la densidad en x
- `inla.pmarginal(q, marginal, ...)` da las probabilidades o función de distribución
- `inla.qmarginal(p, marginal,...)` da los cuantiles
- `inla.rmarginal(n, marginal)` permite obtener n simulaciones
- `inla.hpdmarginal(p, marginal,...)` da la región HPD
- `inla.smarginal(marginal, ...)` da un suavizado con splines de la distribución marginal
- `inla.emarginal(fun, marginal, ...)` calcula el valor esperado de una función

- `inla.mmarginal(marginal,...)` calcula la moda posterior
- `inla.tmarginal(fun, marginal,...)` transforma la distribución marginal de una función del parámetro
- `inla.zmarginal(marginal,...)` calcula descriptivos de la marginal.

Veamos algunos ejemplos sobre nuestro problema. Vamos a mostrar a continuación, en un único gráfico, las distribuciones posteriores de los efectos fijos y el parámetro de dispersión de los datos $ *, con líneas verticales que marquen el valor esperado posterior (en azul) y el HPD95% en rojo.

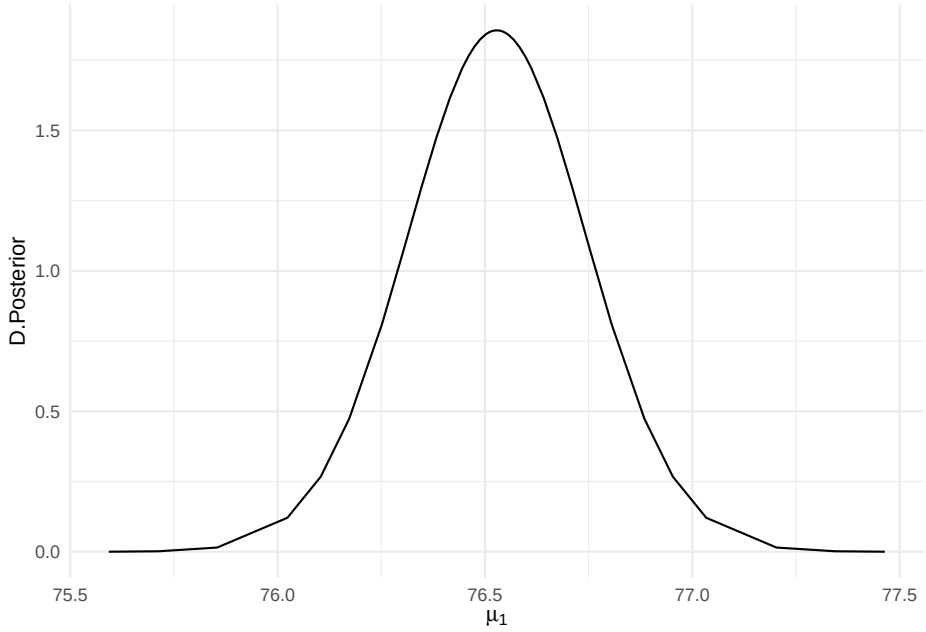
```
# library(gridExtra)
# library(ggplot2)

g=list() # lista en que almacenamos los gráficos con d.posteriores
# Efectos fijos
names.fixed=names(fit$marginals.fixed)
n.fixed=length(names.fixed)
names=c(expression(theta),expression(beta))
for (i in 1:n.fixed){
 g[[i]] = ggplot(as.data.frame(fit$marginals.fixed[[i]])) +
  geom_line(aes(x = x, y = y)) +
  labs(x=names[i],y="")+
   geom_vline(xintercept=inla.hpdmarginal(0.95,fit$marginals.fixed[[i]]),
              linetype="dashed",color="red")+
  geom_vline(xintercept=inla.emarginal(function(x) x,fit$marginals.fixed[[i]]),
              linetype="dashed",color="blue")
}


# Parámetros
g[[3]]= ggplot(as.data.frame(fit$marginals.hyperpar[[1]])) +
  geom_line(aes(x = x, y = y)) +
  labs(x=expression(tau),y="")+
  geom_vline(xintercept=inla.hpdmarginal(0.95,fit$marginals.hyperpar[[1]]),
              linetype="dashed",color="red")+
  geom_vline(xintercept=inla.emarginal(function(x) x,fit$marginals.hyperpar[[1]]),
              linetype="dashed",color="blue")

# Transformamos la posterior en tau para obtener la posterior de sigma
sigma.post=inla.tmarginal(function(tau) tau^(-1/2),
  fit$marginals.hyperpar[[1]])
# y la pintamos
g[[4]]= ggplot(as.data.frame(sigma.post)) +
  geom_line(aes(x = x, y = y)) +
  labs(x=expression(sigma),y="")+
```

```
  geom_vline(xintercept=inla.hpdmarginal(0.95,sigma.post),
             linetype="dashed",color="red")+
  geom_vline(xintercept=inla.emarginal(function(x) x,sigma.post),
             linetype="dashed",color="blue")

library(gridExtra)
grid.arrange(g[[1]],g[[2]],g[[3]],g[[4]],ncol=2)
```



Si queremos la distribución posterior de alguna de las medias $\mu_i = \theta + \beta x_i$, necesitamos añadir el argumento `control.compute=list(return.marginals.predictor=TRUE)` en `inla`. Así podremos representar, por ejemplo, la distribución posterior sobre el peso esperado para el individuo que aparece en el registro 1 de la base de datos, $\mu_1 = \theta + \beta x_1$:

```
fit=inla(formula,family="gaussian",data=davis,
         control.compute=list(return.marginals.predictor=TRUE))
ggplot(as.data.frame(fit$marginals.fitted.values[[1]]),aes(x=x,y=y))+
  geom_line()+
  labs(x=expression(mu[1]),y="D.Posterior")
```



2.8 Simulación de la posterior

Cuando queremos inferir sobre funciones de los efectos latentes e hiperparámetros que no proporciona `INLA` por defecto , podemos recurrir a simular de las distribuciones posteriores de los efectos involucrados, y con ellas evaluar la función que nos interesa, para conseguir una muestra de su distribución posterior.

Para ello es preciso que al ajustar el modelo con `inla` hayamos incluido el argumento `control.compute=list(config=TRUE)`.

Para obtener simulaciones de las correspondientes distribuciones posteriores, utilizamos las funciones:

- `inla.posterior.sample(n, fit,selection)`, para simular los efectos latentes, donde n es el número de simulaciones pretendido, `fit` es el ajuste obtenido con `inla` y `selection` es una lista con el nombre de las componentes (efectos latentes) a simular.

- `inla.hyperpar.sample(n,fit,improve.marginals=TRUE)` para simular de parámetros e hiperparámetros.

Para describir las distribuciones de los nuevos parámetros que queremos evaluar con dichas simulaciones, utilizaremos:

- `inla.posterior.eval()` para funciones sobre los efectos fijos
- `inla.hyperpar.eval()` para funciones sobre los hiperparámetros

Imaginemos que queremos obtener la distribución posterior de $(\theta + \beta \cdot 50)$, que correspondería con el peso real de un sujeto que ha declarado un peso de 50kg. Hemos de simular pues, de las distribuciones posteriores de θ y de β, para luego aplicar la función correspondiente sobre las simulaciones y obtener una muestra posterior de $\theta + \beta \cdot 50$.

```
# reajustamos para poder simular de las posterioris
fit <- inla(formula, data = davis,
  control.compute = list(config = TRUE))
# simulamos especificando los parámetros en los que tenemos interés
sims = inla.posterior.sample(100, fit, selection = list("(Intercept)"=1,repwt = 1))
```

Esto nos devuelve una lista de la dimensión del número de simulaciones (cada simulación en un elemento de la lista), y en cada uno de los elementos tenemos los valores simulados de los hiperparámetros (`hyper`), de los efectos latentes (`latent`)

```
length(sims)
#> [1] 100
names(sims[[1]])
#> [1] "hyperpar" "latent"   "logdens"
sims[[1]]$hyperpar
#> Precision for the Gaussian observations
#>                                0.215942
sims[[1]]$latent
#>                 sample:1
#> (Intercept):1 3.6452565
#> repwt:1       0.9483437
```

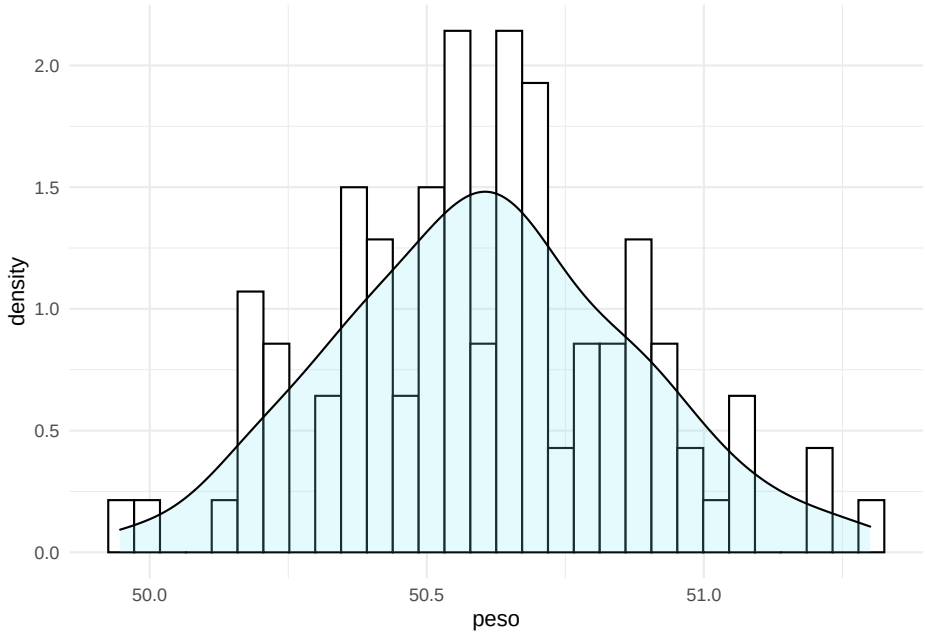
y la log-densidad de la posterior en esos valores (`logdens`)

```
 sims[[1]]$logdens
#> $hyperpar
#> [1] 2.561111
#>
#> $latent
#> [1] 1008.692
#>
#> $joint
#> [1] 1011.253
```

Ahora con la función `inla.posterior.sample.eval`, dado que nuestra función depende de efectos fijos (latentes), (θ, β), evaluamos la operación pretendida, y con descriptivos gráficos y numéricos de las simulaciones, podemos aproximar los descriptivos de la distribución posterior.

```
peso_real=inla.posterior.sample.eval(function(...) {(Intercept)+repwt*50},sims)
peso_real
#>        sample:1 sample:2 sample:3 sample:4 sample:5
#> fun[1] 51.06244 50.52529 50.70436 50.22757 50.34733
#>        sample:6 sample:7 sample:8 sample:9 sample:10
#> fun[1] 50.44512   51.092 50.70692  50.9964  50.99253
#>        sample:11 sample:12 sample:13 sample:14 sample:15
#> fun[1]  50.69731  50.47479  50.89714   51.0874  50.72708
#>        sample:16 sample:17 sample:18 sample:19 sample:20
#> fun[1]  50.78735  50.80462  50.43316  50.80696  50.65538
#>        sample:21 sample:22 sample:23 sample:24 sample:25
#> fun[1]  51.30029  50.51282  50.91033  50.50204  50.64499
#>        sample:26 sample:27 sample:28 sample:29 sample:30
#> fun[1]  50.36004  50.64481  50.61538   50.8281  50.33711
#>        sample:31 sample:32 sample:33 sample:34 sample:35
#> fun[1]  50.93313  50.19321  51.21467  50.87816  50.69253
#>        sample:36 sample:37 sample:38 sample:39 sample:40
#> fun[1]  50.93738  50.24804   50.4517   50.7124  50.78072
#>        sample:41 sample:42 sample:43 sample:44 sample:45
#> fun[1]  50.13386  50.89244  50.18202  50.84824   50.6755
#>        sample:46 sample:47 sample:48 sample:49 sample:50
#> fun[1]  50.86679  50.65348  50.61743  50.51478  50.36984
#>        sample:51 sample:52 sample:53 sample:54 sample:55
#> fun[1]  50.55761  50.71592    50.626  51.02911  50.18379
#>        sample:56 sample:57 sample:58 sample:59 sample:60
#> fun[1]  51.20378  50.83704  50.54454  50.40454  50.67299
#>        sample:61 sample:62 sample:63 sample:64 sample:65
#> fun[1]   50.3547   50.3291  50.73312  50.34969  50.40699
#>        sample:66 sample:67 sample:68 sample:69 sample:70
#> fun[1]  50.51819  50.24715  50.54578  50.35828  50.17157
#>        sample:71 sample:72 sample:73 sample:74 sample:75
#> fun[1]  50.51734  50.56676  50.64262  50.57489  50.55392
#>        sample:76 sample:77 sample:78 sample:79 sample:80
#> fun[1]   50.6588   50.6248  50.38832  50.94177  50.52546
#>        sample:81 sample:82 sample:83 sample:84 sample:85
#> fun[1]  50.33435  50.55521  50.83632   50.6393  49.94654
#>        sample:86 sample:87 sample:88 sample:89 sample:90
#> fun[1]  50.39626  50.88559  50.01248  50.65951  50.24457
#>        sample:91 sample:92 sample:93 sample:94 sample:95
#> fun[1]   50.5403  50.63917  50.59824  50.41855  50.41156
#>        sample:96 sample:97 sample:98 sample:99 sample:100
```

```
#> fun[1]   50.7114   50.5354  50.87874  50.56769   50.17803
pred=data.frame(peso=as.vector(peso_real))
summary(pred)
#>       peso
#>  Min.   :49.95
#>  1st Qu.:50.41
#>  Median :50.62
#>  Mean   :50.61
#>  3rd Qu.:50.79
#>  Max.   :51.30
ggplot(pred,aes(x=peso))+
  geom_histogram(aes(y=..density..), colour="black", fill="white")+
  geom_density(alpha=.2, fill="#80E7F5")
#> `stat_bin()` using `bins = 30`. Pick better value with
#> `binwidth`.
```



2.9 Regresión lineal múltiple con INLA

Vemos a continuación cómo se trabaja la regresión múltiple con INLA, simplemente añadiendo más predictores.

El modelo de regresión asume una distribución normal para los datos, datos los efectos latentes (todos los relacionados con el valor esperado o predictor

lineal) y el resto de parámetros o hiperparámetros del modelo (en nuestro caso la varianza de los datos).

Si tenemos n observaciones

$$(y_i|\mu_i, \sigma^2) \sim N(\mu_i, \sigma^2), \quad i = 1, ...n$$

donde μ_i representa la media y σ^2 la varianza de los datos.

$$E(y_i|\mu_i, \sigma^2) = \mu_i, \ Var(y_i|\mu_i, \sigma^2) = \sigma^2$$

Si tenemos varios regresores $x_1, x_2, ..., x_J$, la media μ_i coincide con el predictor lineal η_i que se construye a partir de una combinación lineal de los predictores:

$$\mu_i = \eta_i = \theta + \sum_{j=1}^{J} \beta_j x_{ij}$$

Nos queda a continuación especificar las distribuciones a priori sobre el vector de efectos latentes, en nuestro caso efectos fijos, $(\theta, \beta_1, ..., \beta_J)$, y sobre el parámetro de dispersión o hiperparámetro σ^2.

Cuando no tenemos información previa especificamos distribuciones difusas (vagas) sobre los parámetros. En INLA por defecto tendremos:

$$\begin{aligned} \theta &\sim N(0, \sigma_\theta^2) \\ \beta_j &\sim N(0, \sigma_\beta^2) \\ log(\tau = 1/\sigma^2) &\sim Log - Ga(1, 0.00005) \end{aligned}$$

con $\sigma_\theta^2 = \infty$ y $\sigma_\beta^2 = 1000$.

Ejemplificamos el análisis de regresión lineal múltiple sobre la base de datos `usair`, en la librería `brinla`. Esta BD contiene datos recopilados para investigar los factores determinantes de la polución, utilizando el nivel de SO2 como variable dependiente y las restantes como variables explicativas potenciales. Las relaciones entre las variables que incluye se muestra en la Figura 2.1 a continuación.

```
data(usair, package = "brinla")
library(GGally) # contiene la función de graficado 'ggpairs'
pairs.chart <- ggpairs(usair, lower = list(continuous = "cor"),
                       upper = list(continuous = "points", combo = "dot"))
ggplot2::theme(axis.text = element_text(size = 6))
#> List of 1
#>  $ axis.text:List of 11
#>   ..$ family       : NULL
#>   ..$ face         : NULL
```

```
#>   ..$ colour       : NULL
#>   ..$ size         : num 6
#>   ..$ hjust        : NULL
#>   ..$ vjust        : NULL
#>   ..$ angle        : NULL
#>   ..$ lineheight   : NULL
#>   ..$ margin       : NULL
#>   ..$ debug        : NULL
#>   ..$ inherit.blank: logi FALSE
#>   ..- attr(*, "class")= chr [1:2] "element_text" "element"
#>  - attr(*, "class")= chr [1:2] "theme" "gg"
#>  - attr(*, "complete")= logi FALSE
#>  - attr(*, "validate")= logi TRUE
pairs.chart
```

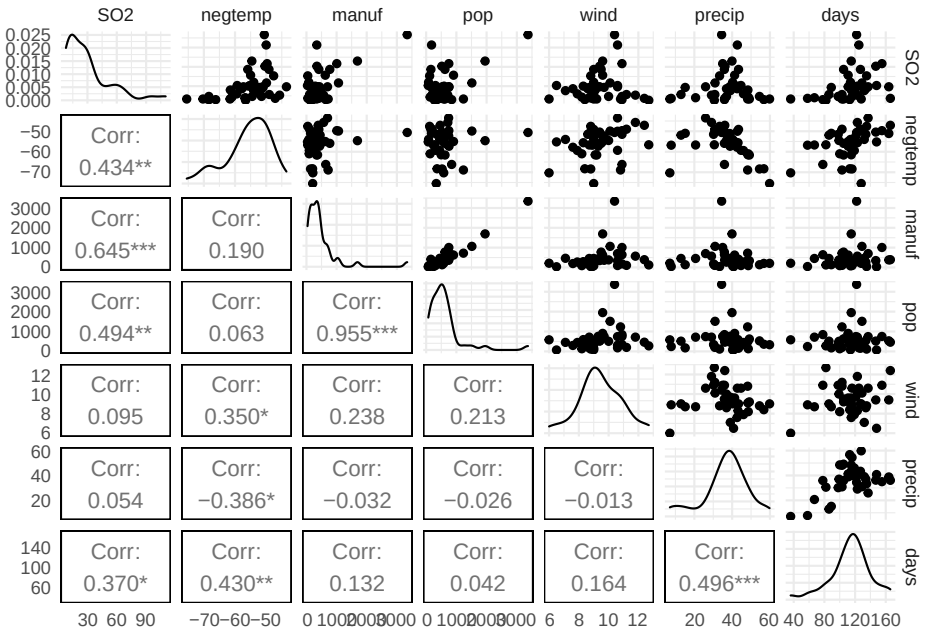


Figure 2.1: Relaciones entre variables en la base de datos usair(brinla).

Apreciamos ya en el gráfico una correlación positiva muy alta entre las variables `pop` y `manuf`, y relevante para `negtemp` y `days` y también para `precip` y `days`, lo que posteriormente condicionará la selección de variables.

2.9.1 Selección de variables

Cuando trabajamos con más de una variable predictora en regresión lineal (realmente en cualquier modelo) surge un problema adicional, que es la **selección de variables**, o selección del mejor modelo de predicción. Esto se resuelve en INLA utilizando diversos criterios de selección entre los que destacamos:

- la verosimilitud marginal (valor de la log-verosimilitud): `mlik`; al cambiarle el signo tendremos *-(log-likelihood)*
- el criterio de información de la deviance (DIC): `dic`
- el criterio de información bayesiana ampliado (WAIC): `waic`
- la transformada integral predictiva (PIT): `cpo`

El procedimiento a utilizar para la selección del modelo (de variables) es el siguiente:

1. ajustar todos los modelos resultantes de todas las combinaciones posibles de predictoras,
2. calcular los índices de selección para cada uno de ellos
3. proceder con la selección en base a dichos valores.

Siempre se prefieren modelos con los valores más bajos para estos criterios y se descartan los que proporcionan valores más altos. Cuando no es el mismo modelo el que proporciona el valor mínimo en estos criterios, habremos de optar por alguno de ellos.

Por defecto, al ajustar un modelo con `inla`, nos devuelve la log-verosimilitud marginal (accesible con `fit$mlik` si el ajuste se guardó en el objeto `fit`). Para obtener las otras medidas de selección, hemos de incluir como argumento de la función `inla`, la opción `control.compute = list(dic = TRUE, waic = TRUE))`. Los valores por defecto de `control.compute` los podemos consultar ejecutando el comando `inla.set.control.compute.default()`.

Vayamos pues con el ajuste del modelo de regresión con todas las variables. Recordemos que si no especificamos el argumento `family`, interpreta por defecto la opción `gaussian`, esto es, normalidad para los datos. Así ajustamos el modelo y obtenemos las siguientes inferencias sobre los efectos fijos.

```
formula <-  SO2 ~ negtemp + manuf + wind + precip + days
fit= inla(formula, data = usair,
          control.compute = list(dic = TRUE, waic = TRUE))
#summary(fit)
round(fit$summary.fixed,3)
#>                mean     sd 0.025quant 0.5quant 0.975quant
#> (Intercept) 135.489 49.862     37.153  135.496    233.784
```

```
#> negtemp       1.769  0.634      0.518    1.769      3.020
#> manuf         0.026  0.005      0.017    0.026      0.035
#> wind         -3.723  1.935     -7.537   -3.723      0.093
#> precip        0.625  0.387     -0.139    0.625      1.388
#> days         -0.057  0.174     -0.400   -0.057      0.287
#>              mode kld
#> (Intercept)    NA   0
#> negtemp        NA   0
#> manuf          NA   0
#> wind           NA   0
#> precip         NA   0
#> days           NA   0
```

La inferencia posterior sobre la desviación típica de los datos, σ, la resolvemos con sus descriptivos.

```
sigma.post= inla.tmarginal(function(tau) tau^(-1/2),
                           fit$marginals.hyperpar[[1]])
inla.zmarginal(sigma.post)
#> Mean            15.6719
#> Stdev           1.85609
#> Quantile  0.025 12.5298
#> Quantile  0.25  14.3479
#> Quantile  0.5   15.4936
#> Quantile  0.75  16.7992
#> Quantile  0.975 19.8195
```

Para seleccionar las variables relevantes seguimos el procedimiento descrito anteriormente. Añadimos también el ajuste del modelo de regresión frecuentista, para el que calculamos como criterio de bondad de ajuste el AIC.

```
# variables en la bd
vars=names(usair)[-1] # variables predictoras (excluye v.dpte)
nvars=length(vars) # nº variables predictoras

# Truco para concatenar todos los modelos posibles en una fórmula (de Faraway)
listcombo <- unlist(sapply(1:nvars,
                           function(x) combn(nvars, x, simplify=FALSE)),
                    recursive=FALSE)
predterms <- lapply(listcombo,
                    function(x) paste(vars[x],collapse="+"))
nmodels <- length(listcombo)
coefm <- matrix(NA,length(listcombo),4,
                dimnames=list(predterms,c("AIC","DIC","WAIC","MLIK")))
```

```
# Ajuste de todos los modelos posibles
for(i in 1:nmodels){
  formula <- as.formula(paste("SO2 ~ ",predterms[[i]]))
  # modelo frecuentista
  lmi <- lm(formula, data=usair)
  # modelo bayesiano
  result <- inla(formula, family="gaussian", data=usair, control.compute=list(dic=TRUE, waic=TRUE
  coefm[i,1] <- AIC(lmi)
  coefm[i,2] <- result$dic$dic
  coefm[i,3] <- result$waic$waic
  coefm[i,4] <- -result$mlik[1]
}
```

Ya solo resta comparar los resultados, respecto de cada uno de los criterios, para seleccionar con qué variables nos quedamos, y por lo tanto con qué modelo de predicción. Basta con encontrar el modelo que proporciona el mínimo valor en cada uno de los criterios.

```
gana.aic = predterms[which.min(coefm[,1])]
gana.dic = predterms[which.min(coefm[,2])]
gana.waic = predterms[which.min(coefm[,3])]
gana.mlik = predterms[which.min(coefm[,4])]
gana.aic;gana.dic
#> [[1]]
#> [1] "negtemp+manuf+pop+wind+precip"
#> [[1]]
#> [1] "negtemp+manuf+pop+wind+precip"
gana.waic;gana.mlik
#> [[1]]
#> [1] "negtemp+manuf+pop+wind+precip"
#> [[1]]
#> [1] "manuf"
```

Concluimos pues que, tanto el criterio AIC en el modelo de regresión frecuentista, como los criterios DIC y WAIC en el modelo de regresión bayesiano, proporcionan el mejor ajuste. Este mejor modelo incluye como predictores las variables 'negtemp+manuf+pop+wind+precip'. Reajustamos el modelo con estas variables para derivar las inferencias y predicciones.

Las inferencias sobre los efectos fijos y la precisión de los datos se muestran a continuación.

```
formula=SO2 ~ negtemp+manuf+pop+wind+precip
fit=inla(formula, family="gaussian", data=usair, control.compute=list(dic=TRUE, waic=TRUE))
fit$summary.fixed
```

```
#>                      mean          sd   0.025quant
#> (Intercept) 100.01507684 30.13792337 40.581810484
#> negtemp       1.12026648  0.41427010  0.303360918
#> manuf         0.06489618  0.01548972  0.034355160
#> pop          -0.03934757  0.01488331 -0.068694574
#> wind         -3.07254481  1.75652267 -6.535114234
#> precip        0.41922382  0.21546548 -0.005636355
#>                 0.5quant   0.975quant mode          kld
#> (Intercept) 100.01864744 159.42790364   NA 1.157176e-08
#> negtemp       1.12029323   1.93701886   NA 1.159988e-08
#> manuf         0.06489594   0.09543860   NA 1.161204e-08
#> pop          -0.03934723  -0.01000253   NA 1.161129e-08
#> wind         -3.07280006   0.39148534   NA 1.154611e-08
#> precip        0.41923027   0.84404705   NA 1.160859e-08
fit$summary.hyperpar
#>                                                    mean
#> Precision for the Gaussian observations 0.005066391
#>                                                      sd
#> Precision for the Gaussian observations 0.001179073
#>                                                 0.025quant
#> Precision for the Gaussian observations 0.00303505
#>                                                    0.5quant
#> Precision for the Gaussian observations 0.004975893
#>                                                  0.975quant mode
#> Precision for the Gaussian observations 0.007618676   NA
```

Y en la Figura 2.2 se muestran las distribuciones posteriores de todos los efectos latentes (efectos fijos) en el modelo: interceptación y coeficientes de los regresores.

```
nfixed=length(fit$names.fixed)
g=list()
for(i in 1:nfixed){
  g[[i]]=ggplot(as.data.frame(fit$marginals.fixed[[i]])) +
    geom_line(aes(x = x, y = y)) +
    labs(x=fit$names.fixed[i],y="")+
    geom_vline(xintercept=inla.hpdmarginal(0.95,fit$marginals.fixed[[i]]),
               linetype="dashed",color="red")+
    geom_vline(xintercept=inla.emarginal(function(x) x,fit$marginals.fixed[[i]]),
               linetype="dashed",color="blue")
}
grid.arrange(g[[1]],g[[2]],g[[3]],g[[4]],g[[5]],g[[6]],ncol=2)
```

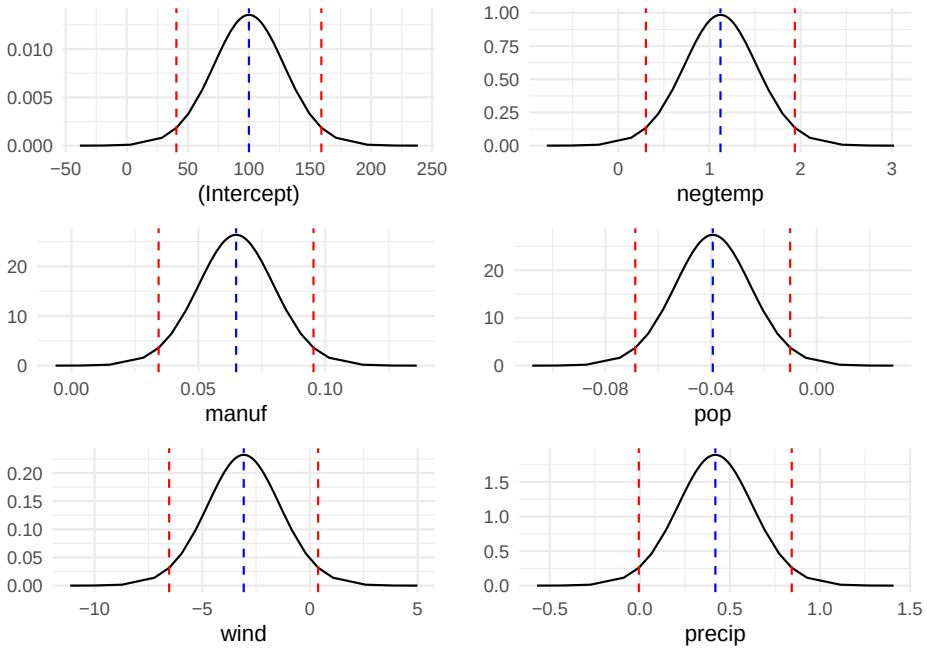


Figure 2.2: Distribuciones posteriores de los efectos latentes.

2.9.2 Predicción de medias

Por último, si queremos predecir la respuesta esperada para ciertos valores de las variables explicativas, no necesariamente existentes en la base de datos, utilizamos el argumento `control.predictor` en la función `inla`, especificando en una lista los valores a predecir. Veamos cómo hacerlo con `inla`, ajustando un modelo sobre un `data.frame` combinado, en el que añadimos tantas filas como predicciones queremos conseguir, con los valores deseados para los predictores (y valores faltantes en la respuesta), y especificamos los valores a predecir en un vector indicador, a través del argumento `control.predictor(list(link=vector.indicador))`. Las predicciones se obtienen después, con el resumen de los datos ajustados `fitted.values`. Recordemos que para poder mostrar las distribuciones predictivas, hemos de añadir en `inla` el argumento `control.compute=list(return.marginals.predictor=TRUE)`.

```
# Predicción con INLA
## valores de los predictores en los que predecir: 3 escenarios
formula=SO2 ~ negtemp+manuf+pop+wind+precip
new.data <- data.frame(negtemp = c(-50, -60, -40),
                       manuf = c(150, 100, 400),
                       pop = c(200, 100, 300),
                       wind = c(6, 7, 8),
```

```
                      precip = c(10, 30, 20),
                      days=c(NA, NA,NA))

## añadimos los tres escenarios de predicción a la bd original,
## dejando como faltantes los valores a predecir de la v.dpte
usair.combinado <- rbind(usair, data.frame(SO2=c(NA,NA,NA),new.data))
## creamos un vector con NA's para observaciones y 1's para predicciones
usair.indicador <- c(rep(NA, nrow(usair)), rep(1, nrow(new.data)))
## reajustamos el modelo añadiendo la opción de predicción de datos
fit.pred <- inla(formula, data = usair.combinado,
                 control.compute=list(return.marginals.predictor=TRUE),
                 control.predictor = list(link = usair.indicador))
## y describimos los valores ajustados para los tres escenarios añadidos
fit.pred$summary.fitted.values[(nrow(usair)+1):nrow(usair.combinado),]
#>                          mean       sd 0.025quant 0.5quant
#> fitted.Predictor.42 31.62374 8.210923   15.43362 31.62453
#> fitted.Predictor.43 26.42297 5.478438   15.62107 26.42334
#> fitted.Predictor.44 53.16282 7.097297   39.16875 53.16340
#>                     0.975quant mode
#> fitted.Predictor.42   47.80910   NA
#> fitted.Predictor.43   37.22256   NA
#> fitted.Predictor.44   67.15337   NA
```

Así graficamos en la Figura 2.3 la distribución predictiva del nivel de SO2 para una combinación dada de valores de las variables predictivas, específicamente la que aparece en el escenario 1 propuesto, o lo que es lo mismo, en el registro 42 de la base de datos completada con las nuevas predicciones: negtem=-50, manuf=150, pop=200, wind=6 y precip=10.

```
pred=fit.pred$marginals.fitted.values[[42]]
ggplot(as.data.frame(pred)) +
  geom_line(aes(x = x, y = y)) +
  labs(x=expression(eta),y="")+
  geom_vline(xintercept=inla.hpdmarginal(0.95,pred),
             linetype="dashed",color="red")+
  geom_vline(xintercept=inla.emarginal(function(x) x,pred),
             linetype="dashed",color="blue")
```

2.10 Conclusiones

En este tema hemos trabajado con el ajuste con INLA de modelos lineales de regresión, esto es, con efectos fijos. En posteriores temas trabajaremos modelos

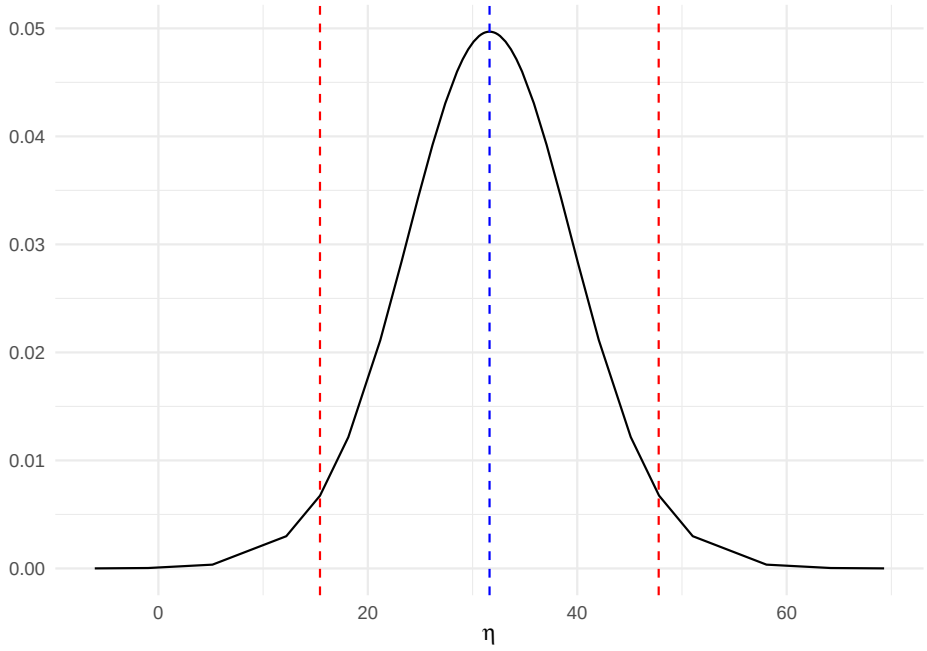


Figure 2.3: Distribución predictiva a posteriori de SO2 para una configuración dada de los predictores.

más sofisticados en los que incluiremos los efectos aleatorios, generalizaremos el modelo lineal y entenderemos el planteamiento de modelos a través de modelos jerárquicos bayesianos.

Chapter 3

Modelo de ANOVA

3.1 Introducción

El modelo de ANOVA se plantea para comparar poblaciones normales, especialmente cuando son más de dos las poblaciones a comparar. Las poblaciones a comparar se identifican a través de una variable clasificadora (de tipo categórico) que actúa como predictora para estimar respuestas medias supuestamente distintas a comparar.

3.2 El modelo de ANOVA

Consideremos una variable respuesta Y que se distribuye normal, y que viene afectada por una variable de clasificación A con a niveles de respuesta distintos (uno por cada una de las poblaciones a comparar). Supongamos que tenemos n_i observaciones de la respuesta para cada uno de los niveles de respuesta de la variable clasificadora, $i = 1, ..., a$. El modelo de ANOVA se plantea asumiendo que en cada nivel o subpoblación, esperamos un valor distinto para la respuesta,

$$(y_{ij}|\mu_i, \sigma^2) \sim N(\mu_i, \sigma^2)$$

de modo que

$$E(y_{ij}|\mu_i, \sigma^2) = \mu_i;\ Var(y_{ij}|\mu_i, \sigma^2) = \sigma^2, \quad i = 1, ..., a;\ j = 1, ..., n_i$$

La formulación habitual de este modelo se suele dar en términos de un efecto global y común a todas las observaciones, θ, y un efecto diferencial respecto del primer nivel del factor de clasificación A, α_i, con los que se construye la media (identificada generalmente por μ) o predictor lineal (identificada generalmente por η) y que en el modelo lineal coinciden:

$$\mu_{ij} = \eta_{ij} = \theta + \alpha_i$$

donde $\alpha_i = \mu_i - \mu_1$ y $\mu_1 = \theta$, para $i \geq 1$, esto es, $\alpha_1 = 0$.

Estamos pues asumiendo que todos los n_i sujetos en el subgrupo de población i identificado por la variable clasificadora A, comparten una media común μ_i y cierta variabilidad σ^2. Podríamos asumir varianzas distintas para cada subpoblación, pero por simplicidad consideramos que son iguales.

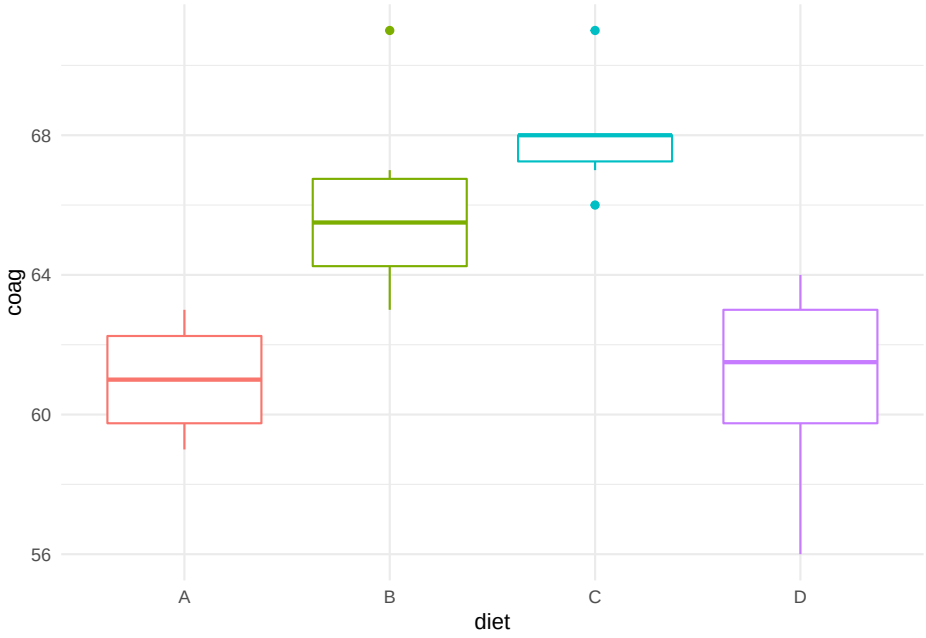
En la modelización bayesiana es preciso añadir distribuciones a priori para cada uno de los parámetros del modelo: los efectos fijos θ, α_i, y la varianza σ^2 de los datos. Ante ausencia de información, se asumirán las distribuciones difusas habituales en INLA:

$$\begin{aligned} (Y_{ij}|\mu_i, \sigma^2) &\sim N(\mu_i, \sigma^2) \\ & \quad \mu_i = \theta + \alpha_i, i \geq 1 \\ \theta &\sim N(0, \infty) \\ \alpha_i &\sim N(0, 1000), i \geq 1 \\ \tau = 1/\sigma^2 &\sim Ga(1, 0.00005) \end{aligned}$$

3.3 Anova de una vía

Vamos a ilustrar el análisis ANOVA en INLA a través de la base de datos `coagulation`, en la librería `faraway`, referidos a un estudio de tiempos de coagulación de la sangre en 24 animales a los que aleatoriamente se les asignó una de entre tres dietas distintas (variable clasificadora `diet`). Posteriormente, para estudiar el efecto de dichas dietas en la coagulación, se tomaron muestras de los tiempos de coagulación (en la variable `coag`, que es la respuesta).

```
data(coagulation, package="faraway")
str(coagulation)
#> 'data.frame':    24 obs. of  2 variables:
#>  $ coag: num  62 60 63 59 63 67 71 64 65 66 ...
#>  $ diet: Factor w/ 4 levels "A","B","C","D": 1 1 1 1 2 2 2 2 2 2 ...
ggplot(coagulation,aes(x=diet,y=coag))+
  geom_boxplot(aes(color=diet))+
  theme(legend.position="none")
```



Estamos planteando un modelo de Anova como el propuesto en la sección anterior, donde α_i identifica el efecto diferencial sobre la respuesta con la dieta A, para el resto de las dietas B y C. Los parámetros del modelo son, como en regresión, los efectos fijos (θ, α_i) y la varianza σ^2, para los que asumimos las priors difusas que por defecto propone INLA. Ajustamos el modelo y obtenemos las inferencias a posteriori

```
formula=coag ~ diet
fit=inla(formula,family="gaussian",data=coagulation,
         control.compute=list(config=TRUE,return.marginals.predictor=TRUE))
fijos=round(fit$summary.fixed,3);fijos
#>               mean    sd 0.025quant 0.5quant 0.975quant
#> (Intercept) 61.016 1.172     58.700   61.016     63.337
#> dietB        4.979 1.513      1.983    4.980      7.970
#> dietC        6.977 1.513      3.981    6.978      9.968
#> dietD       -0.016 1.435     -2.859   -0.016      2.822
#>               mode kld
#> (Intercept)   NA   0
#> dietB         NA   0
#> dietC         NA   0
#> dietD         NA   0
tau=round(fit$summary.hyperpar,3);tau
#>                                                mean    sd
#> Precision for the Gaussian observations 0.197 0.059
#>                                                0.025quant 0.5quant
```

```
#> Precision for the Gaussian observations      0.099    0.191
#>                                          0.975quant mode
#> Precision for the Gaussian observations      0.328   NA
medias=round(fit$summary.linear.predictor,4)
```

Atendiendo a los descriptivos de la distribución posterior para los efectos fijos, concluimos:

- El tiempo esperado de coagulación para los animales que han seguido la dieta A es de 61.016(58.7,63.337).
- Los animales que han seguido la dieta B tienen un tiempo de coagulación esperado superior en 4.979 unidades a los de la dieta A, y dicha diferencia es *significativamente distinta de cero* en el contexto bayesiano, dado que su RC no incluye al cero, (1.983,7.97).
- Una conclusión similar se deriva para la dieta C, que da un tiempo de coagulación esperado superior en 6.977 unidades a los de la dieta A, y una RC (3.981,9.968).
- Las diferencias en los tiempos de coagulación de seguir una dieta D frente a la dieta A no son relevantes. De hecho, la diferencia entre ellos es de -0.016 y el intervalo RC contiene al cero, (-2.859,2.822).

Pintamos a continuación en la Figura 3.1 la distribución posterior de los tiempos esperados de coagulación μ_i (o predictores lineales) para cada una de las dietas. En la Figura 3.2 se añaden las medias posteriores y las regiones creíbles.

```
dietas=levels(coagulation$diet)
pred=NULL
for(i in 1:length(dietas)){
index=which(coagulation$diet==dietas[i])[1]
# distrib. posterior
post=as.data.frame(fit$marginals.fitted.values[[index]])
# media
e=fit$summary.fitted.values[index,1]
rc.low=fit$summary.fitted.values[index,3]
rc.up=fit$summary.fitted.values[index,5]
pred=rbind(pred,data.frame(dieta=dietas[i],
                           post,e,rc.low,rc.up))
}

ggplot(pred, aes(x = x, y =y)) +
  geom_line(aes(color=dieta))+
  labs(x=expression(paste("Tiempo medio de coagulación:",mu)),
       y="D.Posterior")
```

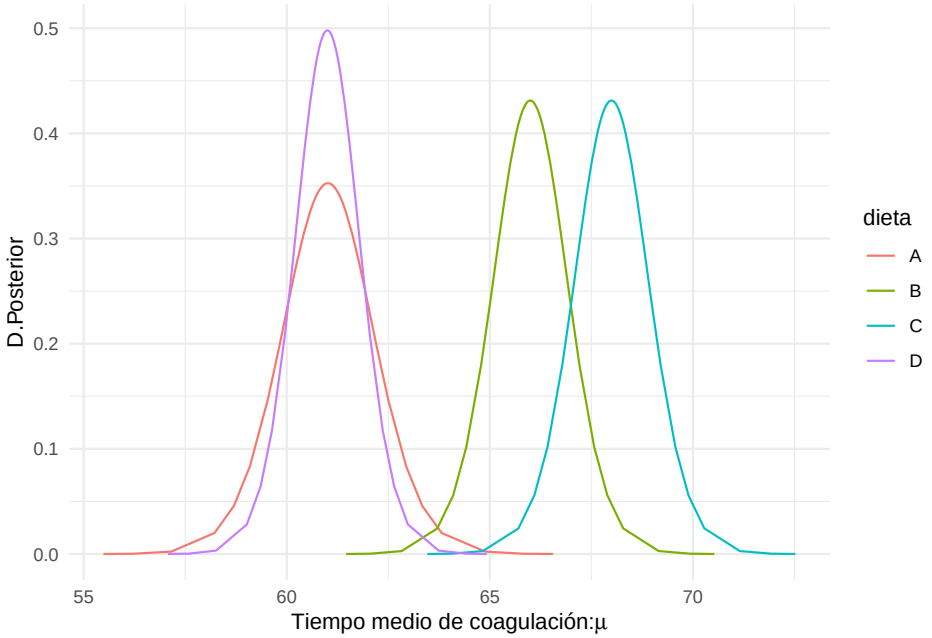


Figure 3.1: Distribución posterior del tiempo medio de coagulación para las 4 dietas.

Podríamos ajustar el modelo prescindiendo del efecto de interceptación y estimar directamente los efectos.

```
formula=coag ~ -1 + diet
fit=inla(formula,family="gaussian",data=coagulation,
         control.compute=list(config=TRUE,return.marginals.predictor=TRUE))
round(fit$summary.fixed,3)
#>          mean    sd 0.025quant 0.5quant 0.975quant mode kld
#> dietA 60.916 1.176     58.577   60.919     63.233   NA   0
#> dietB 65.939 0.961     64.030   65.942     67.832   NA   0
#> dietC 67.937 0.961     66.028   67.940     69.830   NA   0
#> dietD 60.958 0.832     59.306   60.960     62.599   NA   0
round(fit$summary.hyperpar,3)
#>                                              mean   sd
#> Precision for the Gaussian observations 0.197 0.06
#>                                         0.025quant 0.5quant
#> Precision for the Gaussian observations      0.098     0.19
#>                                         0.975quant mode
#> Precision for the Gaussian observations      0.324   NA
```

Con lo cual la representación gráfica se simplifica a través, directamente, de las

distrubuciones posteriores de los efectos fijos.

```
pred=NULL
for(i in 1:length(names(fit$marginals.fixed))){
  pred=rbind(pred,data.frame(as.data.frame(fit$marginals.fixed[[i]]),
                  dieta=names(fit$marginals.fixed)[i],
                  mean=fit$summary.fixed$mean[i],
                  rc.low=fit$summary.fixed$'0.025quant'[i],
                  rc.up=fit$summary.fixed$'0.975quant'[i]))}

ggplot(pred, aes(x = x, y =y)) +
  geom_line(aes(color=dieta))+
  geom_vline(aes(xintercept=mean,color=dieta),linetype="dashed")+
  geom_vline(aes(xintercept=rc.low,color=dieta),linetype="dotted")+
  geom_vline(aes(xintercept=rc.up,color=dieta),linetype="dotted")+
  facet_wrap(vars(dieta))+
  labs(x=expression(paste("Tiempo medio de coagulación:",mu)),
       y="D.Posterior",title="D.Posterior, medias y RC95%")+
  theme(legend.position="none")
```

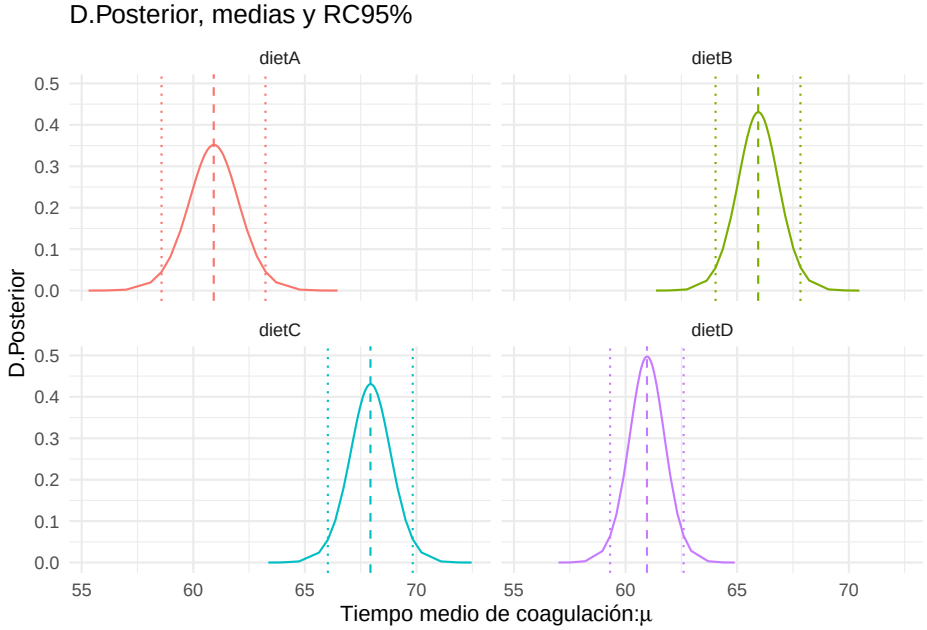


Figure 3.2: Distribuciones posteriores, medias y RC del tiempo esperado de coagulación.

Como ya hacíamos en regresión, podemos inferir sobre la desviación típica de

los datos, σ, transformando la distribución para la precisión τ. La distribución posterior junto con su media y RC95% se muestra en la Figura 3.3.

```
sigma.post=inla.tmarginal(function(tau) tau^(-1/2),
  fit$marginals.hyperpar[[1]])
# y la pintamos
ggplot(as.data.frame(sigma.post)) +
  geom_line(aes(x = x, y = y)) +
  labs(x=expression(sigma),y="D.Posterior")+
  geom_vline(xintercept=inla.hpdmarginal(0.95,sigma.post),
             linetype="dotted",color="blue")+
  geom_vline(xintercept=inla.emarginal(function(x) x,sigma.post),
             linetype="dashed",color="blue")
```

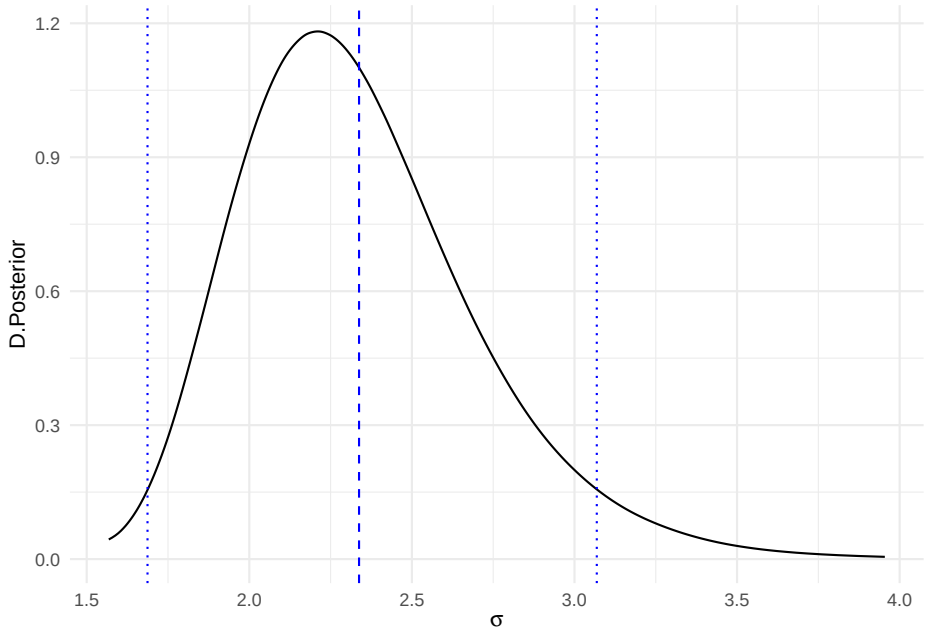
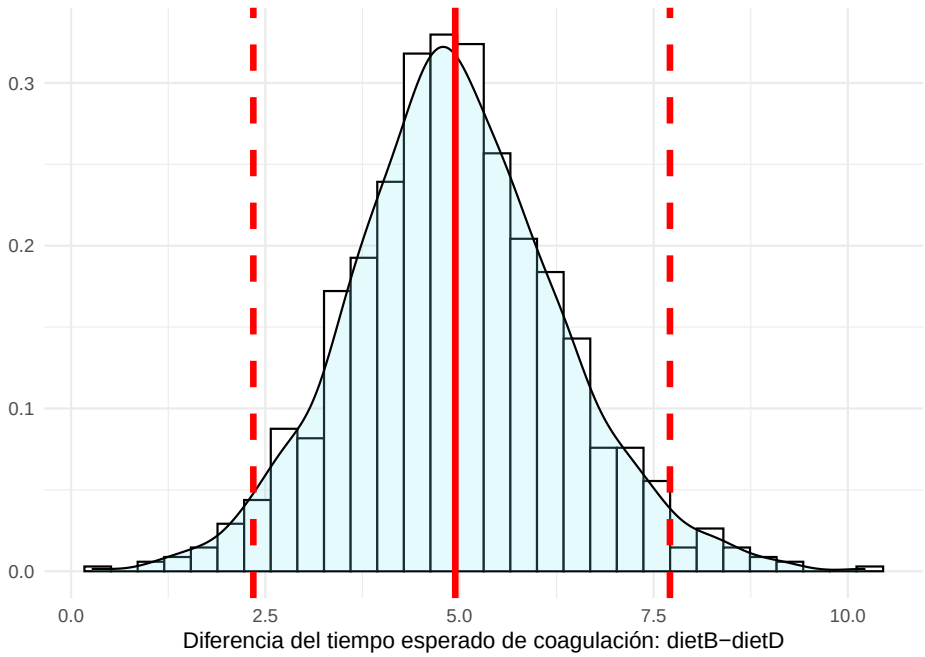


Figure 3.3: Distribución posterior, media y RC, de la desviación típica de los datos (sigma)

```
# Valor esperado
sigma.e=round(inla.emarginal(function(tau) tau^(-1/2),
  fit$marginals.hyperpar[[1]]),4)
# HPD95%
sigma.hpd=round(inla.hpdmarginal(0.95,sigma.post),3)
paste("E(sigma.post)=",sigma.e,"HPD95%=(",sigma.hpd[1],",",sigma.hpd[2],")")
#> [1] "E(sigma.post)= 2.3379 HPD95%=( 1.687 , 3.069 )"
```

Si queremos inferir sobre la diferencia entre cualesquiera de los efectos podemos recurrir a simular la distribución posterior de las diferencias. Por ejemplo, supongamos que queremos comparar la dieta B con la dieta D. Simulamos entonces de las distribuciones posteriores de μ_B y de μ_D, y obtenemos la diferencia $\mu_B - \mu_D$.

```
sims=inla.posterior.sample(1000,fit,selection=list(dietB=1,dietD=1))
dif_BD=as.vector(inla.posterior.sample.eval(function(...) dietB-dietD, sims))
pred=data.frame(dif=dif_BD)
ggplot(pred,aes(x=dif))+
  geom_histogram(aes(y=..density..), colour="black", fill="white")+
 geom_density(alpha=.2, fill="#80E7F5")+
  geom_vline(xintercept=mean(dif_BD),color="red",size=1.5)+
  geom_vline(xintercept=quantile(dif_BD,probs=c(0.025,0.975)),color="red",size=1.5,line
  labs(x="Diferencia del tiempo esperado de coagulación: dietB-dietD",y="")
#> `stat_bin()` using `bins = 30`. Pick better value with
#> `binwidth`.
```



3.4 Anova de varias vías

Generalmente, y en especial cuando trabajamos con experimentación, son varios los factores que controlamos para investigar el efecto que producen en una respuesta continua. Hablamos de modelos de Anova de varias vías.

Utilizamos la base de datos `butterfat` en la librería `faraway` para ilustrar el ajuste con INLA de un modelo de Anova de varias vías. Esta base de datos contiene 100 registros del contenido en grasa láctea, `Butterfat`, para muestras aleatorias de 20 vacas (10 de ellas de 2 años y 10 maduras, con más de 4 años -en la variable `Age`) de cada una de cinco razas (en la variable `Breed`).

El objetivo es investigar las diferencias en materia grasa entre razas y edad, con el fin último de identificar cuáles producen más materia grasa y cuáles menos. Veamos los datos en la Figura 3.4.

```
data(butterfat,package="faraway")
str(butterfat)
#> 'data.frame':    100 obs. of  3 variables:
#>  $ Butterfat: num  3.74 4.01 3.77 3.78 4.1 4.06 4.27 3.94 4.11 4.25 ...
#>  $ Breed    : Factor w/ 5 levels "Ayrshire","Canadian",..: 1 1 1 1 1 1 1 1 1 1 ...
#>  $ Age      : Factor w/ 2 levels "2year","Mature": 2 1 2 1 2 1 2 1 2 1 ...
ggplot(butterfat,aes(x=Breed,y=Butterfat))+
  geom_boxplot(aes(color=Age))+
  coord_flip()
```

A la vista del gráfico, apreciamos que por lo general, en la mayoría de las razas, las vacas más jóvenes tienen menor contenido en materia grasa que las más viejas. Sin embargo, tal afirmación no parece tan clara en las razas *Guernsey* y *Holstein-Fresian*, de modo que para modelizar nuestros datos vamos a considerar a priori, la posibilidad de interacciones entre los factores de clasificación `Breed` y `Age`.

Cuando nos enfrentamos a varios factores de clasificación, cabe la posibilidad de que interaccionen entre ellos, esto es, que en algunos niveles de un factor actúen de forma diferente a los otros cuando se combinan con los niveles de algún otro factor. El orden de una interacción viene dado por el número de factores de clasificación que involucra, de modo que hablamos de interacciones de orden 2 si consideramos la interacción entre dos factores, de orden 3 si consideramos la interacción entre tres factores, etc. Generalmente trabajamos con interacciones de orden bajo, dada la complejidad de las conclusiones en interacciones de orden alto. Por otro lado, siempre es importante tener en cuenta de cuántos datos disponemos para conocer a priori la posibilidad de estimar con fiabilidad los distintos efectos de interacción: una interacción de dos factores con n_1 y n_2 niveles de clasificación respectivamente, revierte en la estimación de $(n_1 - 1) \times (n_2 - 1)$ efectos de interacción adicionales.

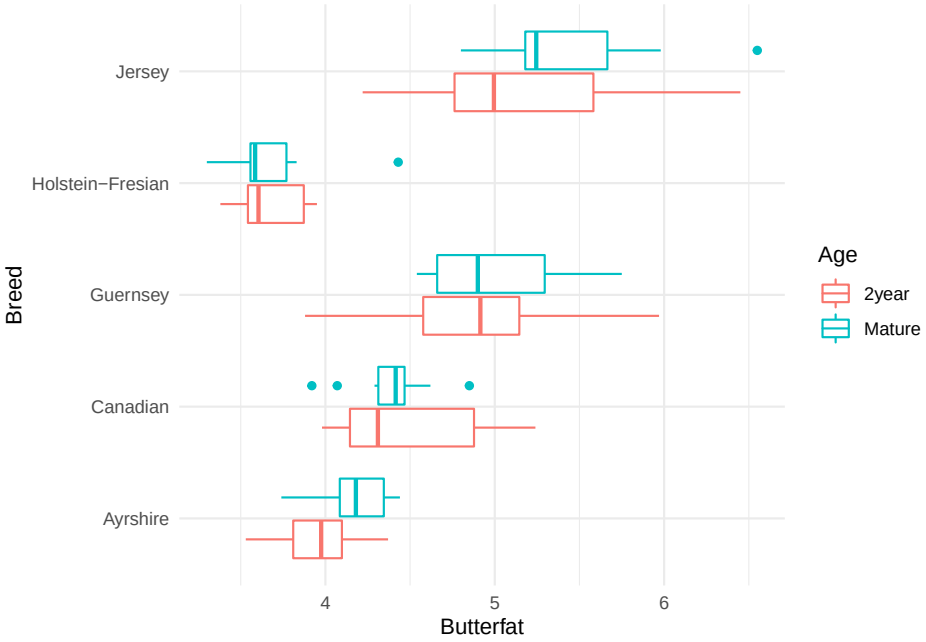


Figure 3.4: Base de datos butterfat, en la librería Faraway

Así, en nuestro problema si estamos planteando la posibilidad de que haya interacciones entre los dos factores de clasificación, estamos asumiendo un modelo de tipo siguiente, asumiendo normalidad en la respuesta:

$$(y_{ijk}|\mu_{ij}, \sigma^2) \sim N(\mu_{ij}, \sigma^2)$$

con

$$\mu_{ij} = \theta + \alpha_i + \beta_j + \alpha\beta_{ij}$$

donde en nuestro ejemplo, α_i es el efecto diferencial (respecto del primer nivel) que aporta el nivel i de la variable `Breed`, β_j el efecto asociado a la variable `Age`, y $\alpha\beta$ la correspondiente interacción entre ellas. En R una interacción de orden 2 entre dos variables f_1 y f_2 se especifica con $f_1 : f_2$; los efectos principales y la interacción se pueden especificar de varios modos alternativos:

$$f_1 + f_2 + f_1 : f_2 \equiv f_1 * f_2 \equiv (f_1 + f_2)2$$

Veamos cómo ajustar con INLA este modelo, recabando también los criterios de selección DIC y WAIC.

```
formula=Butterfat ~ Breed * Age
fit=inla(formula,data=butterfat,
         control.compute=list(dic = TRUE, waic = TRUE))
```

```
round(fit$summary.fixed,3)
#>                                   mean    sd 0.025quant
#> (Intercept)                      3.966 0.131      3.708
#> BreedCanadian                    0.522 0.186      0.157
#> BreedGuernsey                    0.933 0.186      0.568
#> BreedHolstein-Fresian           -0.303 0.186     -0.668
#> BreedJersey                      1.167 0.186      0.802
#> AgeMature                        0.188 0.186     -0.177
#> BreedCanadian:AgeMature         -0.287 0.263     -0.804
#> BreedGuernsey:AgeMature         -0.086 0.263     -0.603
#> BreedHolstein-Fresian:AgeMature -0.175 0.263     -0.692
#> BreedJersey:AgeMature            0.131 0.263     -0.386
#>                                 0.5quant 0.975quant mode
#> (Intercept)                        3.966      4.224   NA
#> BreedCanadian                      0.522      0.887   NA
#> BreedGuernsey                      0.933      1.298   NA
#> BreedHolstein-Fresian             -0.303      0.062   NA
#> BreedJersey                        1.167      1.532   NA
#> AgeMature                          0.188      0.553   NA
#> BreedCanadian:AgeMature           -0.287      0.230   NA
#> BreedGuernsey:AgeMature           -0.086      0.431   NA
#> BreedHolstein-Fresian:AgeMature   -0.175      0.342   NA
#> BreedJersey:AgeMature              0.131      0.648   NA
#>                                 kld
#> (Intercept)                       0
#> BreedCanadian                     0
#> BreedGuernsey                     0
#> BreedHolstein-Fresian             0
#> BreedJersey                       0
#> AgeMature                         0
#> BreedCanadian:AgeMature           0
#> BreedGuernsey:AgeMature           0
#> BreedHolstein-Fresian:AgeMature   0
#> BreedJersey:AgeMature             0
fit$dic$dic
#> [1] 120.4908
fit$waic$waic
#> [1] 121.7376
```

Observamos en la inferencia posterior para los efectos fijos, que todas las RC asociadas a los efectos de interacción contienen al cero, lo que descarta la relevancia de la interacción a la hora de predecir la respuesta. Reajustamos pues el modelo eliminando la interacción, y comprobamos que efectivamente al eliminarla conseguimos reducir los valores del DIC y WAIC que usamos habitualmente para la selección de variables.

```
formula=Butterfat ~ Breed + Age
fit=inla(formula,data=butterfat,
         control.predictor=list(compute=TRUE),
         control.compute=list(return.marginals.predictor=TRUE,
                              dic = TRUE, waic = TRUE))
fit$summary.fixed
#>                               mean         sd  0.025quant
#> (Intercept)             4.0077184 0.10124860  3.80873701
#> BreedCanadian           0.3784787 0.13071136  0.12159362
#> BreedGuernsey           0.8899744 0.13071136  0.63308913
#> BreedHolstein-Fresian -0.3905147 0.13071136 -0.64739962
#> BreedJersey             1.2324714 0.13071136  0.97558612
#> AgeMature               0.1045993 0.08267009 -0.05787069
#>                          0.5quant 0.975quant mode
#> (Intercept)             4.0077182  4.2067008   NA
#> BreedCanadian           0.3784790  0.6353626   NA
#> BreedGuernsey           0.8899746  1.1468581   NA
#> BreedHolstein-Fresian -0.3905145 -0.1336306   NA
#> BreedJersey             1.2324717  1.4893551   NA
#> AgeMature               0.1045993  0.2670692   NA
#>                                kld
#> (Intercept)           2.525012e-09
#> BreedCanadian         2.524990e-09
#> BreedGuernsey         2.524990e-09
#> BreedHolstein-Fresian 2.524988e-09
#> BreedJersey           2.524990e-09
#> AgeMature             2.525221e-09
fit$waic$waic
#> [1] 116.3439
fit$dic$dic
#> [1] 115.3807
```

Observamos ya a partir del modelo ajustado, que el efecto de la edad no es relevante (su RC incluye al cero), pero sin embargo sí que hay diferencias debido a las razas.

Reajustamos de nuevo el modelo, excluyendo la variable `Age`, y verificamos la reducción (ligera) del DIC/WAIC, lo cual justifica usar este modelo para la predicción.

```
formula=Butterfat ~ Breed
fit=inla(formula,data=butterfat,
         control.predictor=list(compute=TRUE),
         control.compute=list(return.marginals.predictor=TRUE,
                              dic = TRUE, waic = TRUE))
fit$summary.fixed
```

```
#>                                mean         sd 0.025quant
#> (Intercept)             4.0600181 0.09271695  3.8778078
#> BreedCanadian           0.3784786 0.13112185  0.1207924
#> BreedGuernsey           0.8899742 0.13112185  0.6322879
#> BreedHolstein-Fresian -0.3905148 0.13112185 -0.6482008
#> BreedJersey             1.2324713 0.13112185  0.9747849
#>                          0.5quant 0.975quant mode
#> (Intercept)             4.0600180  4.2422295   NA
#> BreedCanadian           0.3784788  0.6361636   NA
#> BreedGuernsey           0.8899745  1.1476590   NA
#> BreedHolstein-Fresian -0.3905146 -0.1328296   NA
#> BreedJersey             1.2324715  1.4901560   NA
#>                                 kld
#> (Intercept)           2.471627e-09
#> BreedCanadian         2.471710e-09
#> BreedGuernsey         2.471706e-09
#> BreedHolstein-Fresian 2.471709e-09
#> BreedJersey           2.471713e-09
fit$waic$waic
#> [1] 115.9279
fit$dic$dic
#> [1] 115.0092
```

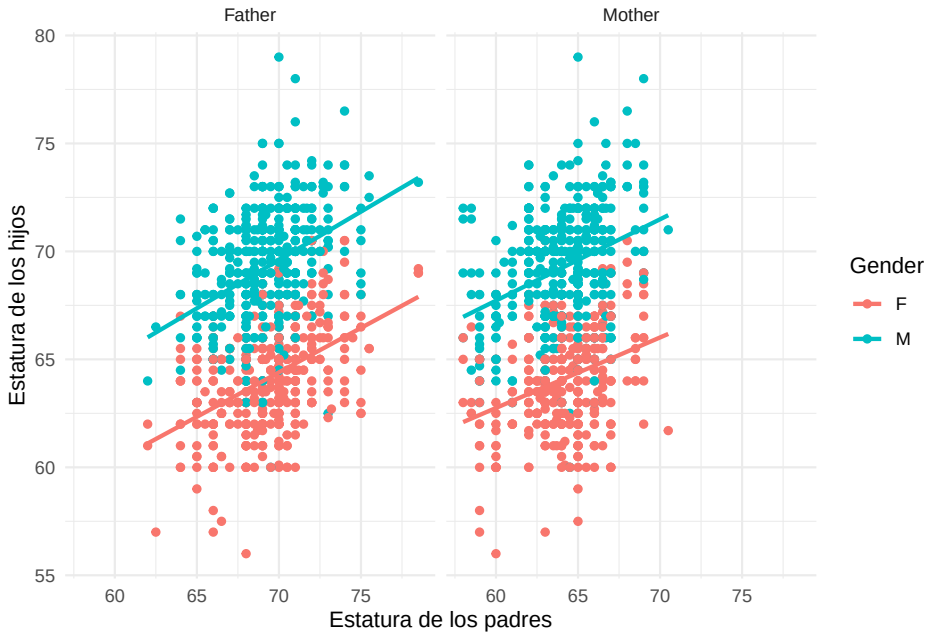
Procederíamos igual que en el modelo de Anova de una vía para la representación de las distribuciones posteriores sobre las medias o predictores lineales en cada una de las razas. Igualmente representaremos la distribución posterior del parámetro de dispersión de los datos σ.

3.5 Análisis de ANCOVA

En ocasiones tenemos una variable respuesta de tipo numérico, y como posibles predictores, tanto variables de tipo numérico como variables clasificadoras o factores. Surge entonces la posibilidad de que los predictores numéricos afecten a la respuesta de modo distinto en diferentes niveles de clasificación de los factores; hablamos entonces de **interacción entre covariables y factores**. Veamos un ejemplo para comprender cómo funcionan estos modelos y cómo se ajustan con INLA.

Consideramos los datos de Galton sobre la regresión de las alturas de los hijos sobre la de los padres (Fte: Galton's Height Data). Tenemos la estatura del padre, de la madre y del hijo/a, identificado/a por su sexo. Vamos a formular un modelo de regresión de la estatura de los hijos en función de la de sus padres y su género.

```
url="https://raw.githubusercontent.com/BayesModel/data/main/Galton.txt"
datos<-read.csv(file=url,header=TRUE,dec=".", sep="")
str(datos)
#> 'data.frame':    898 obs. of  6 variables:
#>  $ Family: chr  "1" "1" "1" "1" ...
#>  $ Father: num  78.5 78.5 78.5 78.5 75.5 75.5 75.5 75.5 75 75 ...
#>  $ Mother: num  67 67 67 67 66.5 66.5 66.5 66.5 64 64 ...
#>  $ Gender: chr  "M" "F" "F" "F" ...
#>  $ Height: num  73.2 69.2 69 69 73.5 72.5 65.5 65.5 71 68 ...
#>  $ Kids  : int  4 4 4 4 4 4 4 4 2 2 ...
datos %>%
  pivot_longer(cols=c("Father","Mother"),
               names_to = "Parents",values_to="Height_parents") %>%
  ggplot(aes(x=Height_parents,y=Height))+
  geom_point(aes(color=Gender))+
  geom_smooth(method="lm",aes(color=Gender),se=FALSE)+
  facet_wrap(vars(Parents))+
  labs(x="Estatura de los padres",y="Estatura de los hijos")
#> `geom_smooth()` using formula 'y ~ x'
```



Asumimos pues como respuesta la variable `y=Height`, como regresores las variables x_1 =`Father` y x_2 =`Mother` con las estaturas del padre y la madre respectivamente, y con factor de clasificación la variable G =`Gender`, con niveles M/F. En principio cabrían posibles interacciones entre los regresores (estaturas del padre y de la madre) y los factores de clasificación (sexo del sujeto). Esto

implicaría que las pendientes de relación 'estatura padres' versus 'estatura hijos' no serían paralelas para los sujetos hombres y mujeres. Planteamos pues el modelo:

$$(y_{ij}|\mu_{ij},\sigma^2) \sim N(\mu_{ij},\sigma^2)$$

con el predictor lineal

$$\eta_{ij} = \mu_{ij} = \beta_0 + (\beta_1 + \alpha_{1M})x_{1j} + (\beta_2 + \alpha_{2M})x_{2j} + \alpha_M; \quad j = 1, ..., n_i; i = M, F$$

donde α_M es el efecto diferencial global de los hombres frente a las mujeres al predecir la estatura, y α_{1M}, α_{2M} son los efectos diferenciales que afectan a los regresores, y por lo tanto que provocan pendientes distintas al predecir la estatura del sujeto con las de los padres, en función de si este es hombre o mujer.

Asumimos una distribución vaga sobre todos los efectos fijos y $\tau = 1/\sigma^2$, y ajustamos el modelo Gausiano en INLA:

```
formula = Height ~ 1+(Father+Mother)*Gender
fit = inla(formula,family = "gaussian",data=datos)
round(fit$summary.fixed,3)
#>                    mean    sd 0.025quant 0.5quant 0.975quant
#> (Intercept)     16.652 3.887      9.028   16.652     24.276
#> Father           0.400 0.039      0.324    0.400      0.477
#> Mother           0.307 0.045      0.218    0.307      0.396
#> GenderM          2.707 5.428     -7.940    2.707     13.354
#> Father:GenderM   0.012 0.058     -0.103    0.012      0.126
#> Mother:GenderM   0.027 0.062     -0.096    0.027      0.149
#>                  mode kld
#> (Intercept)        NA   0
#> Father             NA   0
#> Mother             NA   0
#> GenderM            NA   0
#> Father:GenderM     NA   0
#> Mother:GenderM     NA   0
```

Observamos que ninguna de las interacciones tienen un efecto a considerar (su RC posterior incluye al cero), de modo que las descartamos y reajustamos el modelo sin ellas.

$$\eta_{ij} = \mu_{ij} = \beta_0 + \beta_1 x_{1j} + \beta_2 x_{2j} + \alpha_M; \quad j = 1, ..., n_i; i = M, F.$$

```
formula = Height ~ Father+Mother+Gender
fit = inla(formula,family = "gaussian",data=datos,
        control.predictor=list(compute=TRUE),
```

```
        control.compute=list(return.marginals.predictor=TRUE,
                             dic = TRUE, waic = TRUE))
round(fit$summary.fixed,3)
#>               mean    sd 0.025quant 0.5quant 0.975quant
#> (Intercept) 15.345 2.746      9.958   15.345     20.732
#> Father       0.406 0.029      0.349    0.406      0.463
#> Mother       0.321 0.031      0.260    0.321      0.383
#> GenderM      5.226 0.144      4.943    5.226      5.508
#>             mode kld
#> (Intercept)   NA   0
#> Father        NA   0
#> Mother        NA   0
#> GenderM       NA   0
```

Ahora todos los efectos fijos son relevantes para predecir la estatura de los hijos. Utilizamos este modelo para derivar las inferencias.

Representamos a continuación en la Figura **??**fig:galton1) las distribuciones posteriores de los efectos fijos:

```
fixed=names(fit$marginals.fixed)
g=list()
for(i in 1:length(fixed)){
  g[[i]]=ggplot(as.data.frame(fit$marginals.fixed[[i]]),aes(x=x,y=y))+
    geom_line()+
    geom_vline(xintercept=fit$summary.fixed$mean[i],linetype="dashed")+
    geom_vline(xintercept=fit$summary.fixed[i,3],linetype="dotted")+
    geom_vline(xintercept=fit$summary.fixed[i,5],linetype="dotted")+
    labs(x=fixed[i],y="D.posterior")
}
grid.arrange(g[[1]],g[[2]],g[[3]],g[[4]],ncol=2)
```

En media observamos que la estatura de los hombres es 5.23 unidades superior a la de las mujeres.

Con estas distribuciones podemos calcular cualquier probabilidad, como por ejemplo,la probabilidad de que la estatura de un hombre supere en 5 unidades a la de una mujer, independientemente de cómo sean sus padres:

```
1-inla.pmarginal(5,fit$marginals.fixed$"GenderM")
#> [1] 0.9414591
```

Podemos acceder a las distribuciones posteriores de la estatura esperada de un sujeto concreto y posicionar las estaturas de sus padres, que se muestran en la Figura 3.6

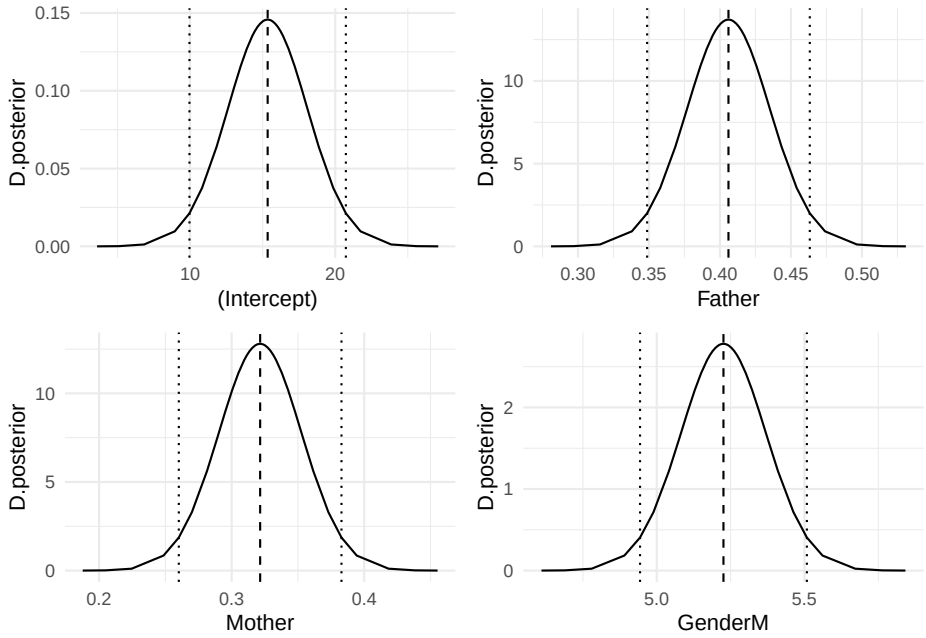


Figure 3.5: Distribución posterior de los efectos fijos

```
# la predicción del predictor lineal para cada sujeto es:
pred<-fit$marginals.linear.predictor
# que en este caso coincide con los valores ajustados
fitted<-fit$marginals.fitted.values
ggplot(as.data.frame(pred$Predictor.1),aes(x=x,y=y))+
  geom_line()+
  labs(x="Estatura media del sujeto 1",y="D.posterior")+
  geom_vline(xintercept=fit$summary.fitted.values$mean[1],linetype="dashed")+
  geom_vline(xintercept=datos$Father[1],linetype="dashed",color="blue")+
  geom_vline(xintercept=datos$Mother[1],linetype="dashed",color="pink")+
  annotate("text",x=datos$Mother[1]+1,y=1,label="Madre")+
  annotate("text",x=datos$Father[1]-1,y=1,label="Padre")
```

Podemos ir más allá, prediciendo la estatura (esperada) de un sujeto, sea hombre o mujer, cuando su padre mide 1.75m (68.9 pulgadas) y su madre 1.70m (66.9 pulgadas). Expresamos los resultados en centímetros.

```
new.data=data.frame(Father=c(68.9,68.9),
                    Mother=c(66.9,66.9),
                    Gender=c("M","F"),
                    Height=c(NA,NA))
datos.combinado <- rbind(datos, data.frame(Family=c(NA,NA),new.data,Kids=c(NA,NA)))
```

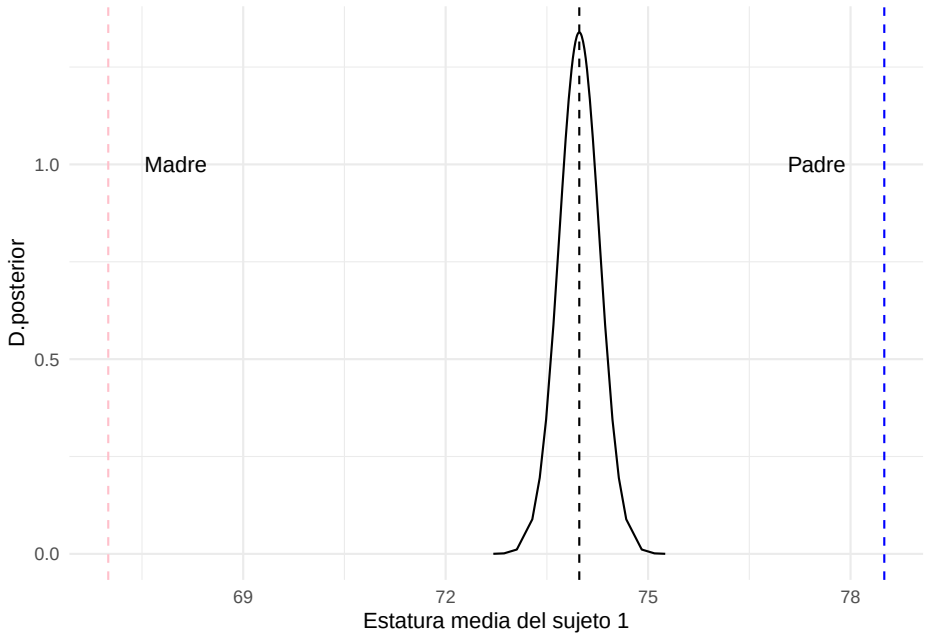


Figure 3.6: Predicción de la estatura del primer sujeto en la muestra

```
## creamos un vector con NA's para observaciones y 1's para predicciones
datos.indicador <- c(rep(NA, nrow(datos)), rep(1, nrow(new.data)))
## reajustamos el modelo añadiendo la opción de predicción de datos
fit.pred <- inla(formula, data = datos.combinado,
                 control.compute=list(return.marginals.predictor=TRUE),
                 control.predictor = list(link = datos.indicador))
## y describimos los valores ajustados para los escenarios añadidos
round(fit.pred$summary.fitted.values[(nrow(datos)+1):nrow(datos.combinado),]*2.54,1)
#>                       mean  sd 0.025quant 0.5quant
#> fitted.Predictor.899 177.9 0.3      177.3    177.9
#> fitted.Predictor.900 164.7 0.3      164.0    164.7
#>                      0.975quant mode
#> fitted.Predictor.899      178.6   NA
#> fitted.Predictor.900      165.3   NA
```

También graficar las distribuciones posteriores y calcular las probabilidades, por ejemplo, de que dicho sujeto supere el 1.65m si es mujer, o el 1.78m si es hombre (Figura 3.7).

```
# Distribuciones predictivas
pred.M=as.data.frame(fit.pred$marginals.fitted.values[[(nrow(datos)+1)]])*2.54
```

```
pred.F=as.data.frame(fit.pred$marginals.fitted.values[[(nrow(datos)+2)]])*2.54
d.pred=rbind(pred.M,pred.F)

# atributo Gender
d.pred$Gender=rep(c("M","F"),c(nrow(pred.M),nrow(pred.F)))
# objetivo de estatura
d.pred$obj=rep(c(178,165),c(nrow(pred.M),nrow(pred.F)))

# cálculo de probabilidades
p165F=round(1-inla.pmarginal(165,pred.F),2)
cat(paste("Pr(estatura>165|mujer,padre=175,madre=170)=",p165F))
#> Pr(estatura>165|mujer,padre=175,madre=170)= 0.16
cat("\n")
p178M=round(1-inla.pmarginal(178,pred.M),2)
cat(paste("Pr(estatura>178|hombre,padre=175,madre=170)=",p178M))
#> Pr(estatura>178|hombre,padre=175,madre=170)= 0.42

d.pred$prob=rep(c(p178M,p165F),c(nrow(pred.M),nrow(pred.F)))


ggplot(d.pred,aes(x=x,y=y))+
  geom_line(aes(color=Gender))+
  geom_vline(aes(xintercept=obj),linetype="dashed")+
  facet_wrap(vars(Gender),scales="free")+
  labs(x="Estatura",y="D.posterior")+
    theme(legend.position = "none")
```

Podríamos también, modificar las especificaciones a priori sobre los parámetros β_0 y β_1 mediante el comando `control.fixed`. Por ejemplo, queremos asumir a priori $\beta_0 \sim N(0, 10^4)$ y $\beta_1 \sim N(0, 100)$ y ver cómo afecta a las inferencias.

```
fit<-inla(formula,family="gaussian",data=datos,
                   control.fixed=list(mean=0,prec=0.01,
                   mean.intercept=0, prec.intercept=0.0001))
round(fit$summary.fixed,3)
#>               mean    sd 0.025quant 0.5quant 0.975quant
#> (Intercept) 15.335 2.746      9.950   15.335     20.721
#> Father       0.406 0.029      0.349    0.406      0.463
#> Mother       0.322 0.031      0.260    0.322      0.383
#> GenderM      5.225 0.144      4.942    5.225      5.507
#>              mode kld
#> (Intercept)    NA   0
#> Father         NA   0
#> Mother         NA   0
#> GenderM        NA   0
```

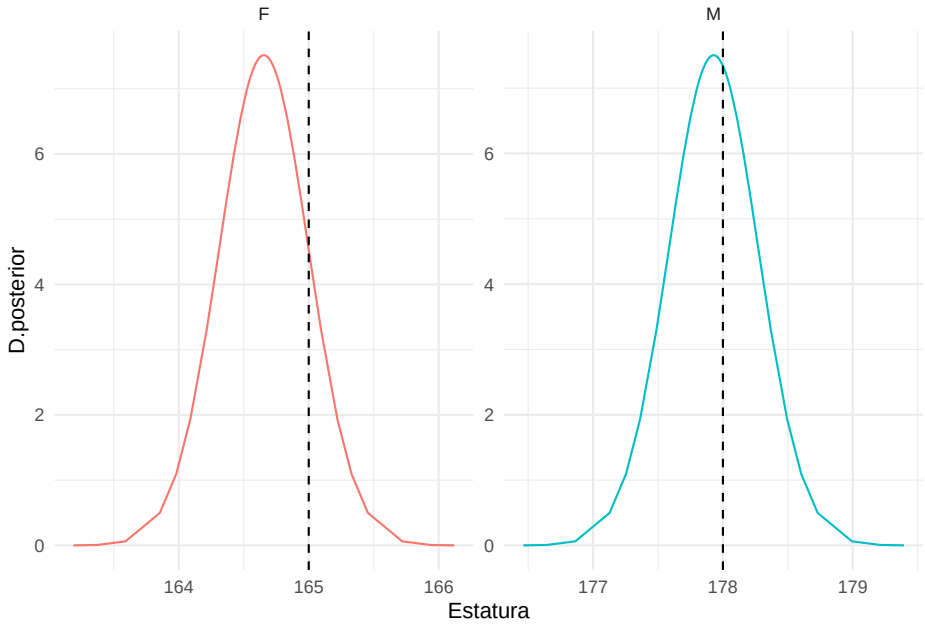


Figure 3.7: Distribución posterior de la estatura de un sujeto cuyo padre mide 1,75cm y madre 1,07cm.

Si queremos especificar medias a priori diferentes para los coeficientes de los distintos regresores, hemos de especificarlos con listas.

```
fit = inla(formula,family = "gaussian",data=datos,
                   control.fixed=list(mean=list(Father=0.2,Mother=0.1)))
round(fit$summary.fixed,3)
#>               mean    sd 0.025quant 0.5quant 0.975quant
#> (Intercept) 15.345 2.746      9.958   15.345     20.732
#> Father       0.406 0.029      0.349    0.406      0.463
#> Mother       0.321 0.031      0.260    0.321      0.383
#> GenderM      5.226 0.144      4.943    5.226      5.508
#>              mode kld
#> (Intercept)    NA   0
#> Father         NA   0
#> Mother         NA   0
#> GenderM        NA   0
```

Si queremos modificar la especificación de la prior en σ^2, o lo que es equivalente, en la precisión τ, con la distribución $log(\tau) \sim N(0,1)$ en lugar de $\tau \sim Ga(1, 10^{-5})$, vemos cómo afecta a la inferencia posterior sobre la precisión.

```
fit_n = inla(formula,family="gaussian", data=datos,
                    control.family=list(hyper=list(
                      prec=list(prior="gaussian",param=c(0,1)))))
# con el modelo log-gamma para precisión
round(fit$summary.hyperpar,3)
#>                                          mean   sd
#> Precision for the Gaussian observations 0.216 0.01
#>                                         0.025quant 0.5quant
#> Precision for the Gaussian observations      0.197    0.216
#>                                         0.975quant mode
#> Precision for the Gaussian observations      0.236   NA
# con el modelo normal para precisión
round(fit_n$summary.hyperpar,3)
#>                                          mean   sd
#> Precision for the Gaussian observations 0.216 0.01
#>                                         0.025quant 0.5quant
#> Precision for the Gaussian observations      0.197    0.216
#>                                         0.975quant mode
#> Precision for the Gaussian observations      0.237   NA
```

Cuando tenemos información previa disponible sobre la variación de los datos, será generalmente más intuitivo expresarla en términos de la desviación estándar σ. Bastará con conseguir la equivalencia en la escala de $log(\tau)$ para incluirla en el modelo. Por ejemplo, si sabemos que la desviación típica está entre 2 y 14, $\sigma \sim Unif(2, 14)$, podemos calcular una prior para la log-precisión del siguiente modo:

1. simular una muestra de $\sigma \sim Unif(2, 14)$
2. transformar a precisiones
3. calcular los parámetros de la Gamma para la precisión, a partir de su media y varianza

Hacemos los cálculos y graficamos la prior en la Figura 3.8.

```
# parámetros para sigma~Un(a1,b1)
a1<-2
b1<-14
# simulamos sigma de una distribución Unif(a1,b1)
sigma<-runif(n=10000,min=a1,max=b1)
# obtenemos la precisión
tau<-1/sigma^2
# Calculamos los parámetros alpha,beta de una distrib. Gamma para la precisión
# mean=alpha/beta; var=alpha/beta^2
beta= mean(tau)/var(tau)
```

```
alpha<-mean(tau)*beta
# dibujamos los valores de la precisión
tau.seq=sort(tau)
  # seq(min(tau),max(tau),length=1000)
prior=data.frame(tau=tau.seq,dprior=dgamma(tau.seq,alpha,beta))
ggplot(prior, aes(x=tau))+
  geom_histogram(aes(y=..density..),color="grey",fill="white")+
  geom_line(aes(y=dprior))
#> `stat_bin()` using `bins = 30`. Pick better value with
#> `binwidth`.
```

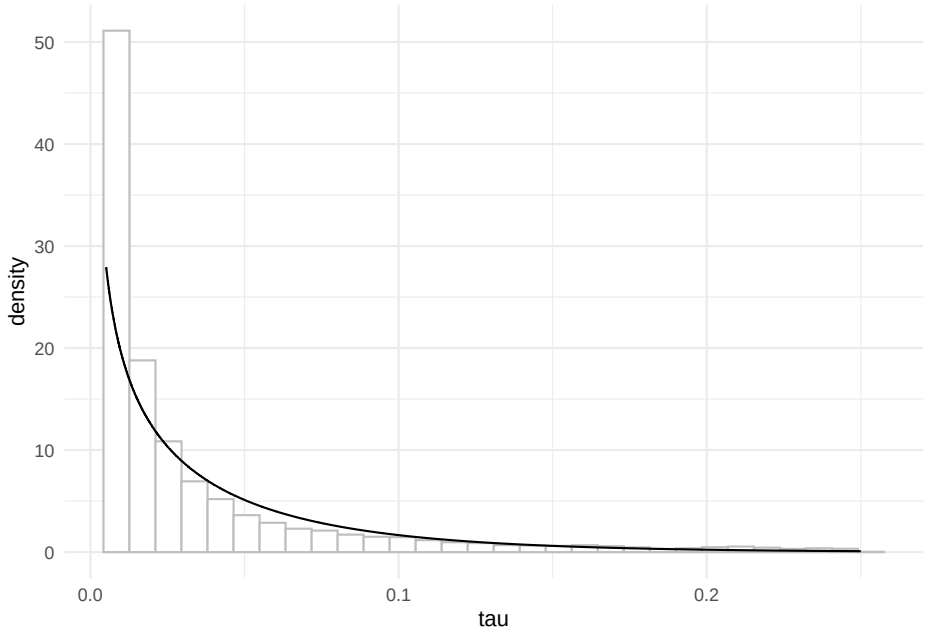


Figure 3.8: Distribución a prior para tau con sigma ~ Uniforme(2,14)

Utilicemos pues esos parámetros para especificar la prior sobre τ en INLA:

```
fit = inla(formula,family="gaussian",data=datos,
                   control.family=list(hyper=list(
                     prec=list(prior="loggamma",param=c(alpha,beta)))))
round(fit$summary.hyperpar,3)
#>                                          mean   sd
#> Precision for the Gaussian observations 0.214 0.01
#>                                         0.025quant 0.5quant
#> Precision for the Gaussian observations      0.195    0.214
#>                                         0.975quant mode
```

```
#> Precision for the Gaussian observations      0.234    NA
```

3.6 Efectos aleatorios

Desde una perspectiva frecuentista un modelo básico de Anova podría ser un modelo de efectos fijos, pero también de efectos aleatorios. Así por ejemplo el 'tratamiento' dado en un ensayo clínico es un efecto relevante para comparar y diferenciar cómo afecta a la respuesta; 'tratamiento' sería entonces un efecto fijo, por ser un efecto de interés primario. En otro ejemplo, se han aplicado varios fertilizantes a cultivos en fincas distintas; el interés primario será comparar los fertilizantes, pero la diversidad de fincas solo se ha incluido para introducir variabilidad e incrementar, por supuesto, el número de datos en el estudio; así pues, 'fertilizante' será un efecto fijo, pero no es un objetivo comparar las fincas, por lo que se considerará como un efecto aleatorio.

Una variable predictiva, numérica o categórica, entra en el modelo como **efecto fijo** cuando se piensa que afecta a todas las observaciones del mismo modo, y que su efecto es de interés primario en el estudio. En un contexto bayesiano un efecto fijo tendrá un coeficiente asociado al que se le asigna a menudo una distribución a priori vaga (mínimo informativa), como una gausiana con media cero y varianza (conocida) grande. En cualquier caso, la distribución a priori que se asume para los efectos fijos es siempre una distribución conocida.

Un **efecto aleatorio** identifica a variables de tipo categórico que no son de interés primario en la investigación, pero que se considera que añaden incertidumbre y por lo tanto variabilidad a la respuesta. La modelización habitual de los efectos aleatorios es una prior gausiana con media cero y una precisión desconocida, para la que será preciso asignar, así mismo, una distribución a priori. La distribución a priori de los efectos aleatorios tiene parámetros desconocidos, llamados **hiperparámetros**, a los que habrá que asignar también distribuciones a priori.

Puesto que no salimos del modelo lineal, seguiremos asumiendo una respuesta normal, *gaussian*, con media igual a un predictor lineal $\mu = \eta = \theta + Zu$, donde Z es la correspondiente matriz de diseño para los efectos aleatorios $z_1, z_2, ...$ Se asume además una varianza desconocida σ^2.

En INLA la fórmula de predicción de una respuesta y a partir de un conjunto de efectos aleatorios z1,z2,… se especifica como:

```
formula = y ~ 1  + f(z1, model="") + f(z2,model="")
```

donde la función $f()$ especifica la relación entre el predictor lineal de la respuesta y los efectos aleatorios z. La función $f()$ tiene muchos argumentos, que se pueden consultar con el comando `?f`. El tipo de relación asumida se

incluye en el argumento `model` o modelo latente, que tiene como posibilidades `names(inla.models()$latent)`. En el modelo lineal, la opción habitual es `model="iid"`, que asume efectos aleatorios independientes e idénticamente distribuidos.

```
names(inla.models()$latent)
#>  [1] "linear"       "iid"          "mec"
#>  [4] "meb"          "rgeneric"     "cgeneric"
#>  [7] "rw1"          "rw2"          "crw2"
#> [10] "seasonal"     "besag"        "besag2"
#> [13] "bym"          "bym2"         "besagproper"
#> [16] "besagproper2" "fgn"          "fgn2"
#> [19] "ar1"          "ar1c"         "ar"
#> [22] "ou"           "intslope"     "generic"
#> [25] "generic0"     "generic1"     "generic2"
#> [28] "generic3"     "spde"         "spde2"
#> [31] "spde3"        "iid1d"        "iid2d"
#> [34] "iid3d"        "iid4d"        "iid5d"
#> [37] "iidkd"        "2diid"        "z"
#> [40] "rw2d"         "rw2diid"      "slm"
#> [43] "matern2d"     "dmatern"      "copy"
#> [46] "clinear"      "sigm"         "revsigm"
#> [49] "log1exp"      "logdist"
```

Veamos cómo ajustar un modelo de efectos aleatorios a partir de un ejemplo sencillo. Comenzamos con la base de datos `broccoli` en la librería `faraway`. Varios cultivadores suministran brócoli a una planta de procesamiento de alimentos. La planta da instrucciones a los cultivadores para que empaquen el brócoli en cajas de tamaño estándar. Debe haber 18 racimos de brócoli por caja y cada racimo debe pesar entre 1,33 y 1,5 libras. Debido a que los productores utilizan diferentes variedades, métodos de cultivo, etc., hay cierta variación en el peso de los racimos. El responsable de la planta seleccionó 3 cultivadores al azar y luego 4 cajas al azar suministradas por estos cultivadores. Se seleccionaron 3 racimos de brocoli de cada caja (a modo de repeticiones).

La variable de interés es el peso del racimo de brócoli, en la variable `wt`. Sin embargo, dado cómo se ha seleccionado la muestra, el objetivo no es ni la comparación entre cultivadores (`grower`), ni entre cajas (`box`), asumiendo que tenemos varios racimos (`cluster`) en cada una de las combinaciones de los anteriores factores. Sin embargo, de manera lógica intuimos que habrá variabilidad entre cajas (efecto aleatorio `box`) y también entre cultivadores (efecto aleatorio `grower`), lo que nos conduce a un modelo en el que todos los predictores, `box` y `grower` intervienen como efectos aleatorios; la variable `cluster` la aprovechamos a modo de repeticiones de medidas en una misma caja de un mismo cultivador.

La base de datos cuenta con 36 registros (3 observaciones en cada combinación `grower-box`.

$$(y_{ijk}|\mu_{ij}, \sigma^2) \sim N(\mu_{ij}, \sigma^2)$$

con

$$\eta_{ij} = \mu_{ij} = \theta + \alpha_i^G + \beta_j^B; \quad i = 2, 3; j = 2, 3, 4$$

el peso medio que comparten todos los racimos en cada combinación de agricultor-caja: $k = 1, 2, 3$, y donde α^G representa el efecto aleatorio asociado al cultivador (`grower`) y β^B a la caja (`box`).

Así el vector de efectos latentes está compuesto por el efecto fijo de interceptación *theta* y los efectos aleatorios $u = (\alpha_2^G, \alpha_3^G, \beta_2^B, \beta_3^B, \beta_4^B)$.

El siguiente paso es especificar una distribución a priori sobre los parámetros. INLA por defecto asigna una prior difusa sobre la interceptación θ y también sobre la precisión de los datos $\tau = 1/\sigma^2$. Dado que los α_i^G representan el efecto diferencial asociado al cultivador, es razonable asumir independencia entre todos estos parámetros y una distribución idéntica, centrada en el cero (ante ausencia de información) y con una varianza desconocida. Con esto estamos diciendo que en principio no tenemos información sobre que efectivamente el efecto cultivador sea relevante (media cero), pero sí que añade cierta variabilidad σ_α^2 a la respuesta. Del mismo modo, se asume que los β_j^B son a priori independientes e idénticamente distribuidos (iid) con una normal centrada en el cero (ante ausencia de información) y con varianza desconocida σ_β^2.

$$\begin{aligned}
\theta &\sim N(0, \sigma_\theta^2),\ \sigma_\theta^2 = \infty \\
log(\tau) &\sim Log - Ga(1, 5 \cdot 10^{-5}) \\
\alpha_i^G &\sim_{iid} N(0, \sigma_\alpha^2), i = 2, 3 \\
\beta_j^B &\sim_{iid} N(0, \sigma_\beta^2), j = 2, 3, 4
\end{aligned}$$

Surgen pues, dos nuevos parámetros en las a priori, o hiperparámetros, σ_α^2 y σ_β^2, a los que también habrá que asignar una distribución a priori. Dado que se trata de varianzas, por defecto INLA asume gammas inversas difusas, o lo que es lo mismo, log-gammas difusas para las precisiones

$$\begin{aligned}
\tau_\alpha = 1/\sigma_\alpha^2 &\sim Ga(1, 5 \cdot 10^{-5}) \\
\tau_\beta = 1/\sigma_\beta^2 &\sim Ga(1, 5 \cdot 10^{-5})
\end{aligned}$$

Surgen pues, tres niveles de especificación del modelo: datos, parámetros e hiperparámetros, que generan un **modelo jerárquico de tres niveles**, y sobre el que hablaremos más adelante.

```
data(broccoli, package="faraway")
str(broccoli)
#> 'data.frame':    36 obs. of  4 variables:
#>  $ wt     : num  352 369 383 339 367 328 376 359 388 365 ...
#>  $ grower : Factor w/ 3 levels "1","2","3": 1 1 1 2 2 2 3 3 3 1 ...
#>  $ box    : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 2 ...
#>  $ cluster: Factor w/ 3 levels "1","2","3": 1 2 3 1 2 3 1 2 3 1 ...
formula = wt ~ f(grower,model="iid")+ f(box,model="iid")
fit = inla(formula, family="gaussian",data=broccoli,
           control.compute = list(dic=TRUE,waic=TRUE))
```

Cuando queremos mostrar los resultados a posteriori sobre los efectos aleatorios a partir de un ajuste `fit` con `inla`, tenemos las siguientes opciones:

- `fit$summary.random` resume la inferencia posterior sobre los efectos aleatorios
- `names(fit$marginals.random)` lista los nombre de todos los efectos aleatorios
- `fit$marginals.random` da las distribuciones posteriores marginales de los efectos aleatorios

```
fit$summary.random
#> $grower
#>   ID          mean          sd   0.025quant      0.5quant
#> 1  1  2.575283e-07 0.004709469 -0.009322078  2.479812e-07
#> 2  2 -1.802713e-06 0.004709469 -0.009324333 -1.735883e-06
#> 3  3  1.545184e-06 0.004709469 -0.009320669  1.487901e-06
#>    0.975quant mode          kld
#> 1 0.009322642   NA 2.733238e-08
#> 2 0.009320387   NA 2.733239e-08
#> 3 0.009324051   NA 2.733238e-08
#>
#> $box
#>   ID          mean         sd  0.025quant      0.5quant
#> 1  1  2.745375e-05 0.01372465 -0.02741857  2.289973e-05
#> 2  2 -1.574019e-05 0.01372464 -0.02748910 -1.312919e-05
#> 3  3 -6.955020e-06 0.01372463 -0.02747473 -5.801330e-06
#> 4  4 -4.758694e-06 0.01372463 -0.02747114 -3.969305e-06
#>   0.975quant mode          kld
#> 1 0.02750826   NA 9.882400e-05
#> 2 0.02743767   NA 9.882047e-05
#> 3 0.02745201   NA 9.881908e-05
#> 4 0.02745559   NA 9.881890e-05
```

Sin embargo, lo relevante en un modelo de efectos aleatorios es la inferencia sobre las varianzas asociadas a los datos, pero también la variabilidad extra que añaden los efectos aleatorios:

```
fit$summary.hyperpar
#>                                                mean
#> Precision for the Gaussian observations 3.839158e-03
#> Precision for grower                    2.403323e+04
#> Precision for box                       4.224069e+03
#>                                                  sd
#> Precision for the Gaussian observations 9.327656e-04
#> Precision for grower                    2.335509e+04
#> Precision for box                       2.275383e+03
#>                                          0.025quant
#> Precision for the Gaussian observations 2.247165e-03
#> Precision for grower                    7.003574e+02
#> Precision for box                       1.174507e+03
#>                                             0.5quant
#> Precision for the Gaussian observations 3.762866e-03
#> Precision for grower                    1.650764e+04
#> Precision for box                       3.785638e+03
#>                                          0.975quant mode
#> Precision for the Gaussian observations 5.890322e-03   NA
#> Precision for grower                    8.387697e+04   NA
#> Precision for box                       9.847258e+03   NA
```

Vemos que tanto la precisión asociada al efecto aleatorio caja (`box`), como al efecto cultivador, `grower`, son muy grandes, lo que implica varianzas muy pequeñas que posiblemente nos permitiría prescindir de dichos efectos aleatorios para ajustar un mejor modelo.Cuando transformamos a escala de desviaciones estándar, tenemos la distribución posterior para los tres tipos de error, representados en la Figura 3.9.

```
nombres=c("sigma","grower","box")
sigma.post=as.data.frame(inla.tmarginal(function(tau) tau^(-1/2),
  fit$marginals.hyperpar[[1]]))
sigma.grower.post =as.data.frame(inla.tmarginal(function(tau) tau^(-1/2),
  fit$marginals.hyperpar[[2]]))
sigma.box.post = as.data.frame(inla.tmarginal(function(tau) tau^(-1/2),
  fit$marginals.hyperpar[[3]]))

sigma=rbind(sigma.post,sigma.grower.post,sigma.box.post)
sigma$efecto=rep(c("sigma","grower","box"),
                 c(nrow(sigma.post),nrow(sigma.grower.post),nrow(sigma.box.post)))
```

```
ggplot(sigma,aes(x=x,y=y)) +
  geom_line(aes(color=efecto)) +
  labs(x=expression(sigma),y="D.Posterior")+
  facet_wrap(vars(efecto),scales = "free")+
  theme(axis.text.x = element_text(angle = 45))
```

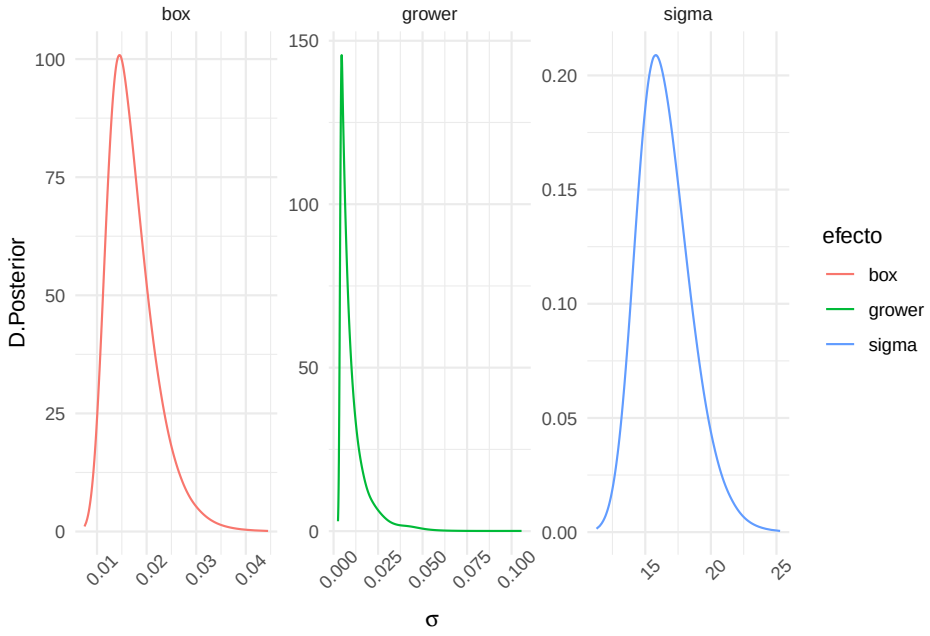


Figure 3.9: Distribución posterior de la desviación típica para las tres fuentes de error: datos, caja y cultivador

No obstante, antes de tomar una decisión sobre la exclusión de los efectos aleatorios, vamos a hacer una aproximación del porcentaje de varianza explicada por cada una de estas fuentes de variación. Utilizando simulaciones de las distribuciones posteriores de $\sigma^2, \sigma_\alpha^2$ y σ_β^2 vamos a calcular la contribución a la varianza del efecto cultivador, $ $\{\}^{2/(}2 + \{\}^{2+}\{\}\hat{}2)$$ *y la contribución a la varianza del efecto caja,* $ $\{\}2/(\hat{}2 + \{\}\hat{}2+\{\}\hat{}2)$$, y calcular con ellas un porcentaje promedio.

```
n=1000
tau=as.data.frame(inla.hyperpar.sample(n,fit,improve.marginals=TRUE))
sigma2=apply(tau,2,function(x) 1/x)
colnames(sigma2)=c("sigma2d","sigma2G","sigma2B")
# contribución a la varianza de grower
cG= sigma2[,2]/apply(sigma2,1,sum)
# contribución a la varianza de grower
```

```
cB= sigma2[,3]/apply(sigma2,1,sum)
cat(paste("Contribución media de grower a la varianza:",round(mean(cG)*100,6),"por 100","\n"))
#> Contribución media de grower a la varianza: 0.00011 por 100
cat(paste("Contribución media de box a la varianza:",round(mean(cB)*100,6),"por 100"))
#> Contribución media de box a la varianza: 0.000132 por 100
```

Ante estos resultados, y dados los valores del DIC (307.1382561) y del WAIC (307.0682172), se justifica la opción de prescindir de los efectos `grower` y `box` como efectos aleatorios y ajustar el modelo con un único efecto fijo global.

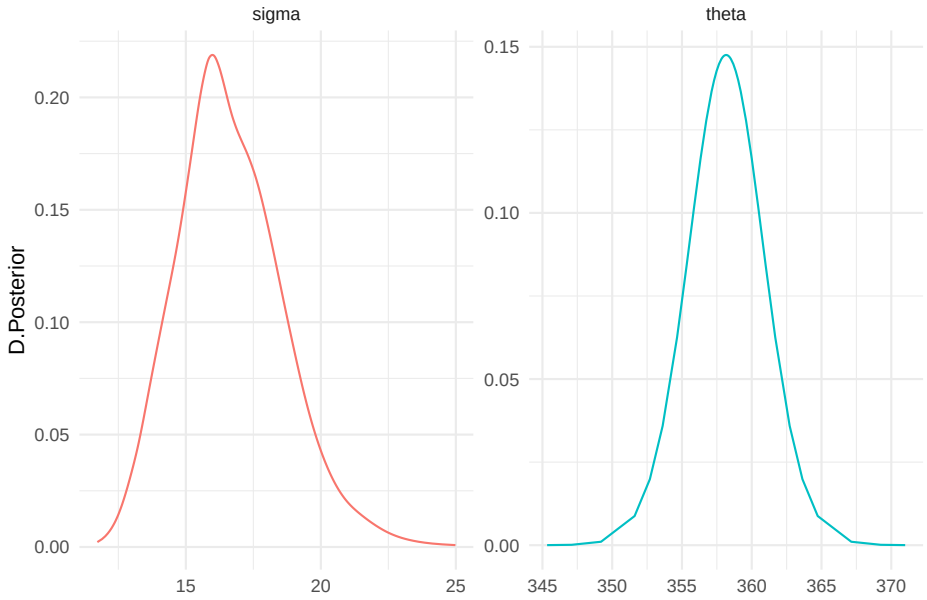
```
formula = wt ~ 1
fit = inla(formula, family="gaussian",data=broccoli,
           control.compute = list(dic=TRUE,waic=TRUE))
fit$summary.fixed
#>                  mean       sd 0.025quant 0.5quant
#> (Intercept) 358.1667 2.771112   352.6996 358.1667
#>             0.975quant mode          kld
#> (Intercept)   363.6337   NA 1.211947e-08
fit$summary.hyperpar
#>                                                    mean
#> Precision for the Gaussian observations 0.003758278
#>                                                      sd
#> Precision for the Gaussian observations 0.0008870513
#>                                                 0.025quant
#> Precision for the Gaussian observations 0.002261799
#>                                                   0.5quant
#> Precision for the Gaussian observations 0.003680228
#>                                                 0.975quant mode
#> Precision for the Gaussian observations 0.005742648   NA
```

Vemos que la variación en los indicadores DIC (308.0755903) y WAIC (308.0833022) es despreciable para este nuevo modelo.

Inferimos a continuación con las distribuciones posteriores para la media global y la varianza de los datos.

```
theta.post = as.data.frame(fit$marginals.fixed[[1]])
sigma.post=as.data.frame(inla.tmarginal(function(tau) tau^(-1/2),
  fit$marginals.hyperpar[[1]]))

posterior=rbind(theta.post,sigma.post)
posterior$efecto=rep(c("theta","sigma"),
                     c(nrow(theta.post),nrow(sigma.post)))
ggplot(posterior,aes(x=x,y=y)) +
  geom_line(aes(color=efecto)) +
```

```
  labs(x="",y="D.Posterior")+
  facet_wrap(vars(efecto),scales = "free")+
  theme(legend.position = "none")
```



Hemos concluido con este análisis, que todos los cultivadores han respetado los protocolos de calidad establecidos para el empaquetado en cajas.

3.7 Modelos mixtos

En ocasiones cuando ajustamos un modelo lineal tendremos algunos factores de clasificación que operan como efectos fijos y otros que operan como efectos aleatorios. Estaremos ante **modelos lineales mixtos**. Siendo estrictos, realmente el modelo con solo efectos aleatorios ya es un modelo mixto, puesto que incluye como efecto fijo una interceptación global.

En un modelo lineal mixto seguimos asumiendo una respuesta normal, *gaussian*, con media igual a un predictor lineal $\eta = X\beta + Zu$, donde X es una matriz de diseño con los efectos fijos $x_1, x_2, ...$, y Z la correspondiente para los efectos aleatorios $z_1, z_2, ...$ Se asume además una varianza desconocida que puede ser distinta para distintos niveles de los predictores, e incluso contener correlaciones entre niveles distintos, y que en general se suele expresar a través de una matriz de covarianzas Σ, $(y|\eta, \Sigma) \sim N(\eta, \Sigma)$.

En INLA la fórmula de predicción de una respuesta y a partir de un conjunto de efectos fijos x1,x2,..., y un conjunto de efectos aleatorios z1,z2,... se especifica como:

```
formula = y ~ 1 + x1 + x2  + f(z1, model="") + f(z2,model="")
```

De nuevo mencionar que la opción más habitual para los efectos aleatorios en un modelo lineal es `model="iid"`.

3.7.1 Datos longitudinales con pendientes iguales

Veamos cómo resolver las inferencias a través de un ejemplo disponible en R-bloggers, proporcionado por Patrick Curran y descargable desde Github. Se refieren estos datos, a un estudio con 405 niños en los dos primeros años de la escuela infantil, medidos a lo largo de cuatro instantes equidistantes (no medidos todos en todos los sujetos) para registrar su progreso en lectura y en comportamiento antisocial. Nos centramos aquí exclusivamente en intentar predecir los progresos en lectura (variable `read`) para cada sujeto (identificado como `id`) a lo largo de los 4 instantes de medición (`occasion`).

Cargamos los datos, prescindimos de los que tienen valores faltantes, y los inspeccionamos en la Figura 3.10.

```
url="https://raw.githubusercontent.com/BayesModel/data/main/curran_dat.csv"
curran_dat=read.csv(url) %>%
  select(id, occasion, read) %>%
  filter(complete.cases(.))
# el identificador de cada sujeto lo convertimos a factor
curran_dat$id=as.factor(curran_dat$id)
curran_dat$occasion=as.double(curran_dat$occasion)
# Relaciones
g1=ggplot(curran_dat, aes(x=as.factor(occasion),y=read))+
  geom_boxplot()
g2=ggplot(curran_dat, aes(x=occasion,y=read))+
  geom_line(aes(group=id),color="grey",size=0.4)
grid.arrange(g1,g2,ncol=2)
```

Como base vamos a asumir normalidad en la respuesta de un sujeto i en un instante j, y plantear un modelo lineal para obtener nuestras conclusiones.

$$(y_{ij}|\mu_{ij},\sigma^2) \sim N(\mu_{ij},\sigma^2);\ i=1,...,450; j=1,2,3,4$$

A la vista de la Figura 3.10 podríamos considerar el tiempo afecta de modo positivo y lineal sobre las habilidades lectoras (a más tiempo, mejores habilidades), lo que convierte a la variable `occasion` en una covariable numérica (efecto fijo): nos interesará cuantificar cómo afecta el tiempo a la capacidad lectora.

Sin embargo, también en el gráfico apreciamos que cada sujeto arranca de un inicio diferente, o lo que es lo mismo, su recta de predicción tiene una interceptación

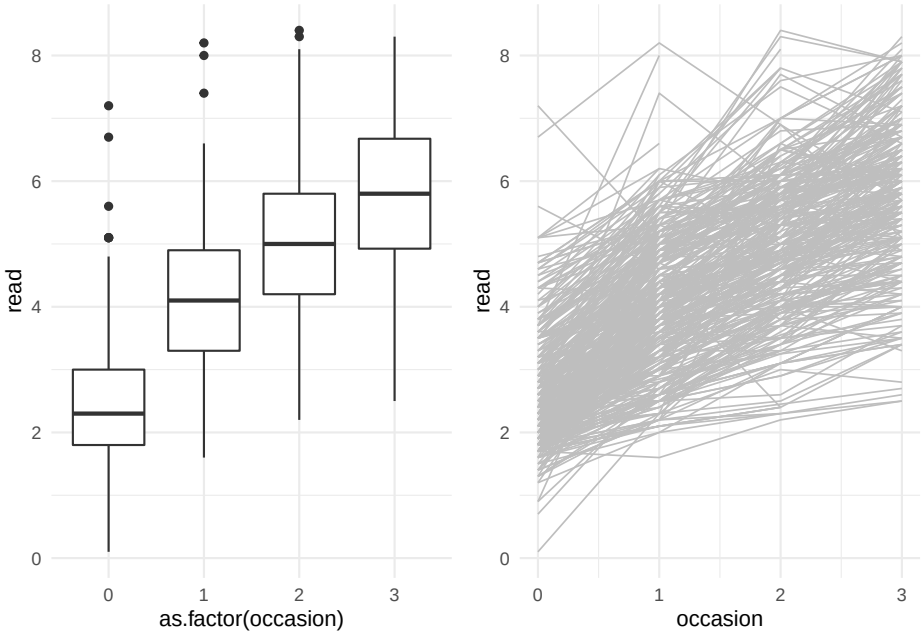


Figure 3.10: Descripción de la BD CurranLong sobre desarrollo de las habilidades lectoras en niños.

distinta. Puesto que no nos interesa comparar los individuos, planteamos incorporar un efecto aleatorio del sujeto (variable `id`). Estamos pues, hablando de predecir la habilidad lectora de un sujeto i en un instante $t_{ij} = j$ con:

$$\mu_{ij} = \theta + \alpha_i + \beta \cdot t_{ij}$$

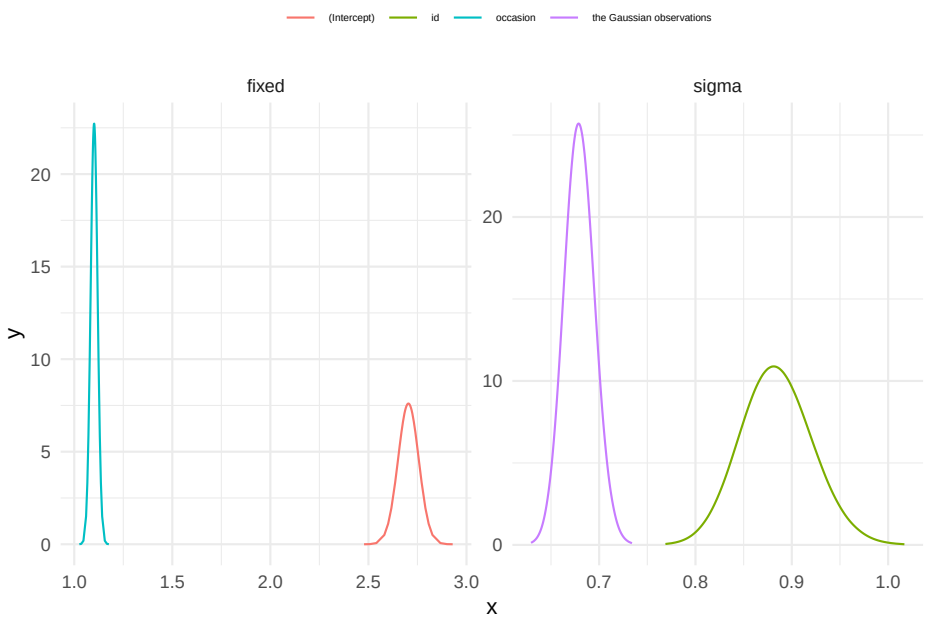
con

$$\begin{aligned} \text{Nivel II} & \\ \theta &\sim N(0, 100) \\ \beta &\sim N(0, 100) \\ \alpha_i &\sim N(0, \sigma^2_\alpha) \\ \tau = 1/\sigma^2 &\sim Ga(0.001, 0.001) \\ \text{Nivel III} & \\ \tau_\alpha = 1/\sigma^2_\alpha &\sim Ga(0.001, 0.001) \end{aligned}$$

La varianza σ^2_α representa la variabilidad existente entre las distintas interceptaciones o niveles cognitivos de los sujetos en el inicio del estudio.

En la Figura **??** se muestran las distribuciones posteriores obtenidas sobre efectos fijos y varianzas.

```
prec.prior=list(prec=list(param=c(0.001,0.001)))
formula= read ~ occasion + f(id,model="iid",hyper = prec.prior)
fit=inla(formula,family="gaussian",data=curran_dat,
          control.family=list(hyper=prec.prior))
fit$summary.fixed
#>                  mean         sd 0.025quant 0.5quant
#> (Intercept) 2.703751 0.05266776   2.600408 2.703757
#> occasion    1.101333 0.01760845   1.066782 1.101336
#>             0.975quant mode          kld
#> (Intercept)   2.807060   NA 1.184714e-11
#> occasion      1.135862   NA 5.550080e-12
fit$summary.hyperpar
#>                                                mean        sd
#> Precision for the Gaussian observations 2.169523 0.1013917
#> Precision for id                        1.286183 0.1093176
#>                                         0.025quant 0.5quant
#> Precision for the Gaussian observations   1.975790 2.167497
#> Precision for id                          1.083454 1.281901
#>                                         0.975quant mode
#> Precision for the Gaussian observations   2.375188   NA
#> Precision for id                          1.513859   NA

# Agrupamos todas las distribuciones posteriores
nfixed=length(names(fit$marginals.fixed))
nhyp=length(names(fit$marginals.hyperpar))
res=NULL
for(i in 1:nfixed){
res=rbind(res,data.frame(fit$marginals.fixed[[i]],
                         id=names(fit$marginals.fixed)[i],
                          tipo="fixed"))
}
for(j in 1:nhyp){
    res=rbind(res,data.frame(
    inla.tmarginal(function(tau) tau^(-1/2),fit$marginals.hyperpar[[j]]),
                         id=str_sub(names(fit$marginals.hyperpar)[j], start =15),
                          tipo="sigma"))
}
ggplot(res,aes(x=x,y=y))+
  geom_line(aes(color=id))+
  facet_wrap(vars(tipo),scales="free")+
  theme(legend.position="top",
        legend.title=element_blank(),
        legend.text = element_text(size=5))
```



Datos longitudinales con pendientes distintas

Belenky et al. (2003) describen un estudio de los tiempos de reacción en pacientes a los que se ha privado de sueño durante 10 días; cada día se ha ido registrando la respuesta para cada uno de los 18 sujetos en el estudio. Los datos están disponibles como `sleepstudy` en la librería `lme4` y tienen como variables el tiempo medio de reacción en microsegundos (`Reaction`), el número de días con privación de sueño (`Days`) y un id para cada sujeto (`Subject`). Los tiempos de reacción se transforman a segundos para tener mayor estabilidad. Aun así, en la Figura 3.11 se aprecia que el número de días de falta de sueño afecta de modo distinto a cada sujeto.

```
data(sleepstudy,package="lme4")
sleepstudy$Reaction <- sleepstudy$Reaction / 1000
ggplot(sleepstudy,aes(x=Days,y=Reaction))+
  geom_point(size=0.5)+
  geom_smooth(method="lm",color="blue",size=0.5)+
  facet_wrap(vars(Subject),ncol=6)+
  theme(axis.text.x = element_text(size=5),
        axis.text.y = element_text(size=5))
#> `geom_smooth()` using formula 'y ~ x'
```

Un modelo razonable para estos datos es un modelo lineal que relacione los tiempos de reacción con los días, pero que tenga interceptaciones y pendientes diferentes para cada sujeto. El efecto sujeto entraría en el modelo como un efecto aleatorio para relacionar todos los datos del mismo sujeto sin perder la asunción de independencia entre las observaciones de sujetos distintos. Si llamamos $y =$

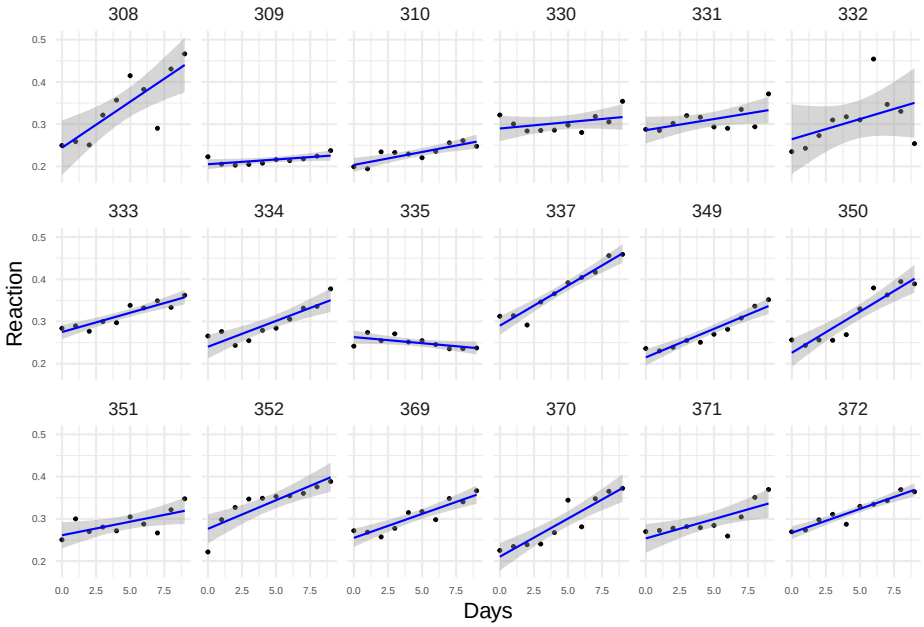


Figure 3.11: Tiempos de reacción en función del número de días con falta de sueño (sleepstudy) para los 18 sujetos en el estudio

Reaction a la respuesta, estaríamos planteando el siguiente modelo:

$$y_{ij}|\mu_{ij}, \sigma^2 \sim N(\mu, \sigma^2), i = 1, ..., 18; j = 1, ..., 10$$

con

$$\mu_{ij} = \theta + \alpha_i + \beta \cdot x_{ij} + \gamma_{ij}$$

donde el predictor x es la variable `Days`, (θ, β) se tratarían como efectos fijos con a prioris difusas ante falta de información, y (α_i, γ_{ij}) como efectos aleatorios, con normales centradas en cero y una varianza desconocida a la que habría que asignar así mismo, una distribución a priori. El modelo jerárquico que surge es pues:

$$
\begin{aligned}
&\text{Nivel I} \\
y_{ij}|\mu_{ij},\sigma^2 &\sim N(\mu,\sigma^2), i=1,...,18; j=1,...,10 \\
&\text{Nivel II} \\
\theta &\sim N(0,1000) \\
\beta &\sim N(0,1000) \\
\alpha_i &\sim N(0,\sigma^2_\alpha) \\
\gamma_{ij} &\sim N(0,\sigma_\gamma)^2 \\
\tau=1/\sigma^2 &\sim Ga(0.001,0.001) \\
&\text{Nivel III} \\
\tau_\alpha=1/\sigma^2_\alpha &\sim Ga(0.001,0.001) \\
\tau_\gamma=1/\sigma^2_\alpha &\sim Ga(0.001,0.001)
\end{aligned}
$$

En INLA modelizamos esta propuesta utilizando como predictores;

- la covariable para generar una interceptación 'media' (efecto fijo),
- el efecto aleatorio de cada sujeto para generar interceptaciones distintas,
- la interacción entre la covariable y el efecto aleatorio, a través de la matriz de diseño que hemos de construir específicamente, y definir en paralelo un índice de la misma dimensión de los datos, para aplicarla. En la interacción la primera variable define el número de grupos y la segunda el valor de la covariable.

```
prec.prior=list(prec=list(param=c(0.001,0.001)))
# matriz de diseño para la interacción
Z <- as(model.matrix( ~ 0 + Subject:Days, data = sleepstudy), "Matrix")
# índice para aplicar la matriz de diseño
DayR=1:nrow(sleepstudy)
formula= Reaction ~ Days + f(Subject,model="iid",hyper=prec.prior)+
  f(DayR,model="z",Z=Z,hyper = prec.prior)
fit=inla(formula,family="gaussian",data=sleepstudy,
         control.compute=list(config=TRUE))
#> Warning in inla.model.properties.generic(inla.trim.family(model), mm[names(mm) == :
#>   Use this model with extra care!!! Further warnings are disabled.
#> as(<dgCMatrix>, "dgTMatrix") is deprecated since Matrix 1.5-0; do as(., "TsparseMat
```

Obtenemos en consecuencia, efectos fijos e hiperparámetros, cuyas inferencias posteriores se resumen con:

```
round(fit$summary.fixed,3)
#>              mean    sd 0.025quant 0.5quant 0.975quant mode
#> (Intercept) 0.251 0.008      0.236    0.251      0.267   NA
#> Days        0.010 0.003      0.004    0.010      0.017   NA
#>             kld
#> (Intercept)   0
#> Days          0
round(fit$summary.hyperpar,3)
#>                                                    mean       sd
#> Precision for the Gaussian observations 1566.713  182.872
#> Precision for Subject                   1314.756  571.894
#> Precision for DayR                      6067.459 2128.986
#>                                         0.025quant 0.5quant
#> Precision for the Gaussian observations   1231.363 1558.539
#> Precision for Subject                      520.889 1209.117
#> Precision for DayR                        2819.871 5762.946
#>                                         0.975quant mode
#> Precision for the Gaussian observations   1950.856   NA
#> Precision for Subject                     2728.351   NA
#> Precision for DayR                       11105.474   NA
```

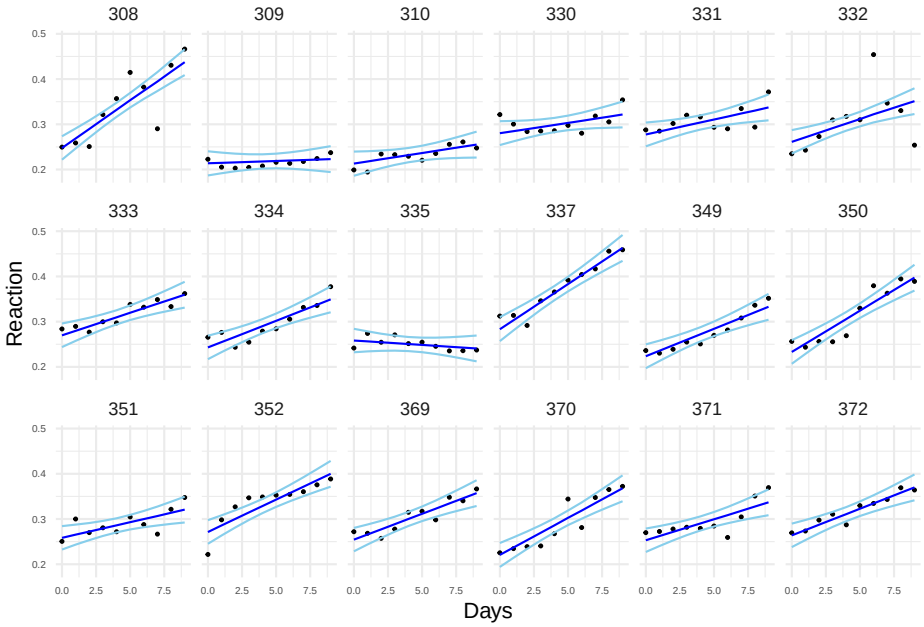
Y los efectos aleatorios:

```
names(fit$marginals.random)
#> [1] "Subject" "DayR"
head(fit$summary.random$Subject)
#>    ID         mean         sd    0.025quant     0.5quant
#> 1 308 -0.003768873 0.01446456 -0.0322860523 -0.003748488
#> 2 309 -0.037615026 0.01485947 -0.0674457611 -0.037392853
#> 3 310 -0.038204929 0.01487073 -0.0680612695 -0.037981229
#> 4 330  0.028706038 0.01469943  0.0002843602  0.028533780
#> 5 331  0.025993687 0.01465102 -0.0023753055  0.025835697
#> 6 332  0.009899260 0.01447477 -0.0184010587  0.009836417
#>     0.975quant mode          kld
#> 1  0.024632758   NA 5.824804e-09
#> 2 -0.009029249   NA 2.027815e-08
#> 3 -0.009601978   NA 2.046358e-08
#> 4  0.058095611   NA 1.427011e-08
#> 5  0.055251271   NA 1.290458e-08
#> 6  0.038554384   NA 6.836930e-09
head(fit$summary.random$DayR)
#>   ID         mean         sd   0.025quant     0.5quant
#> 1  1 6.924238e-06 0.001501481 -0.002937713 6.924441e-06
#> 2  2 1.056645e-02 0.004274277  0.002188708 1.055373e-02
#> 3  3 2.106572e-02 0.008121904  0.005144073 2.103765e-02
```

```
#> 4  4 3.184263e-02 0.012056271  0.008205544 3.180030e-02
#> 5  5 4.249502e-02 0.016013237  0.011099076 4.243831e-02
#> 6  6 5.322669e-02 0.019979621  0.014052978 5.315583e-02
#>   0.975quant mode          kld
#> 1 0.00295156   NA 5.527465e-11
#> 2 0.01901744   NA 7.952215e-09
#> 3 0.03714924   NA 1.020996e-08
#> 4 0.05572396   NA 1.066195e-08
#> 5 0.07421819   NA 1.083096e-08
#> 6 0.09280922   NA 1.090800e-08
```

En la Figura **??** mostramos los datos y también los valores ajustados para las rectas, en términos de las interceptaciones y pendientes medias de las correspondientes distribuciones posteriores, además de la banda de estimación que construimos con los correspondientes percentiles de las posterioris.

```
sleepstudy.pred = sleepstudy %>%
  mutate(fitted=fit$summary.fitted.values$mean,
         rc.low=fit$summary.fitted.values$"0.025quant",
         rc.up=fit$summary.fitted.values$"0.975quant")

ggplot(sleepstudy.pred,aes(x=Days,y=Reaction))+
  geom_point(size=0.5)+
  geom_line(aes(y=fitted),color="blue")+
  geom_line(aes(y= rc.low),color="skyblue")+
  geom_line(aes(y=rc.up),color="skyblue")+
  facet_wrap(vars(Subject),ncol=6)+
  theme(axis.text.x = element_text(size=5),
        axis.text.y = element_text(size=5))
```



En la Figura 3.12 mostramos la distribución posterior de los errores de datos y aleatorios.

```
nhyp=length(names(fit$marginals.hyperpar))
res=NULL
for(j in 1:nhyp){
    res=rbind(res,data.frame(
    inla.tmarginal(function(tau) tau^(-1/2),fit$marginals.hyperpar[[j]]),
                           id=str_sub(names(fit$marginals.hyperpar)[j], start =15)))
}
ggplot(res,aes(x=x,y=y))+
  geom_line(aes(color=id))+
  labs(x=expression(sigma),y="")+
  theme(legend.position="top",
        legend.title=element_blank(),
        legend.text = element_text(size=5))
```

3.7.2 Efectos anidados

Hablamos de efectos anidados cuando cada miembro de un grupo está contenido completamente dentro de una única unidad de otro grupo. Que un factor A esté anidado en otro B, implica que cada nivel de B contiene niveles distintos de A, esto es, cada nivel de A está vinculado solo a algún nivel de B.

La base de datos `eggs` en la librería `faraway` nos resulta útil para describir este

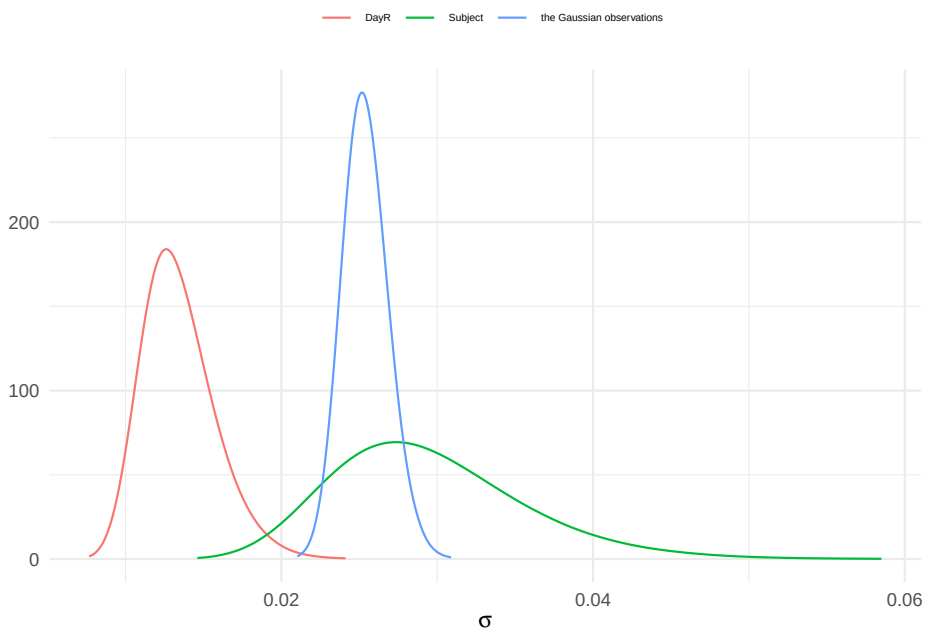


Figure 3.12: Distribución posterior del error de los datos y el error aleatorio

tipo de modelos con efectos anidados. Estos datos son los resultantes de un experimento para testar la consistencia en los tests de laboratorio que realizaban laboratorios distintos, técnicos distintos. Para ello se dividió en varias muestras un tarro de polvo de huevo seco homogeneizado (con idéntico contenido graso). Se enviaron 4 muestras a cada uno de los 6 laboratorios. De esas 4 muestras, 2 se etiquetaron como G y 2 como H (aun siendo idénticas). Se dieron instrucciones a los laboratorios de dar dos muestras a dos técnicos distintos. Los técnicos recibieron instrucciones de dividir sus muestras en dos partes y medir el contenido graso de cada una. Así, cada laboratorio reportó 8 mediciones del contenido graso (`Fat`), cada técnico 4 mediciones, con 2 réplicas en cada una de las dos muestras.

Realmente, el laboratorio, el técnico y la muestra solo deberían generar variabilidad en la respuesta, pero en ningún caso generar mediciones distintas. Estamos pues interesados en investigar la magnitud del error debido al laboratorio, al técnico y a la identificación de muestras. Es por ello que tiene sentido considerarlos efectos aleatorios.

Tenemos así en este ejemplo, a los técnicos (`Technician`) anidados en los laboratorios (`Lab`). En la Figura 3.13 se muestra claramente la variación entre laboratorios, entre técnicos, y debida al efecto irreal de tener dos muestras distintas G y H (`Sample`).

```
data(eggs,package="faraway")
ggplot(eggs,aes(x=Lab,y=Fat))+
  geom_boxplot(aes(color=Technician))+
  facet_wrap(vars(Sample))
```

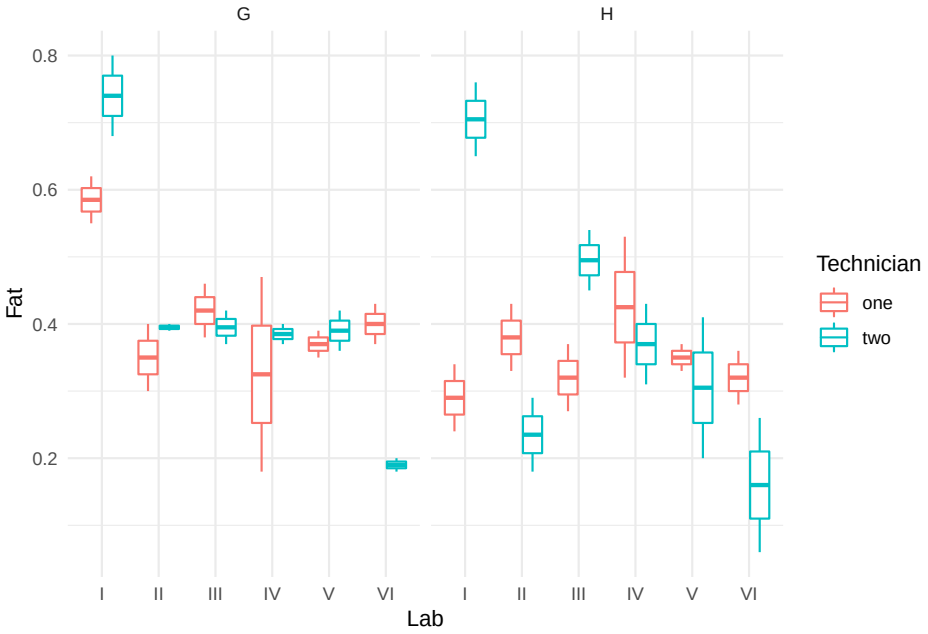


Figure 3.13: Descripción de eggs: variación entre laboratorios y técnicos.

El modelo que planteamos para estimar la respuesta y_{ijk}, contenido graso de la muestra k $(k = 1, 2)$ del laboratorio i $(i = 1, ..., 6)$, por el técnico j $(j = 1, 2)$ está basado como siempre, en el modelo normal, $(y_{ijk}|\mu_{ijk}, \sigma^2) \sim N(\mu_{ijk}, \sigma^2)$, con una media o predictor lineal representado por:

$$\mu_{ijk} = \theta + \alpha_i^{lab} + \beta_{j:i}^{tec} + \gamma_{k:(j:i)}^{sam}$$

y asumiendo en un segundo nivel del modelo las distribuciones a priori:

$$\begin{aligned}
\theta &\sim N(0, 1000) \\
\tau = 1/\sigma^2 &\sim Ga(0.001, 0.001) \\
\alpha_i^{lab} &\sim N(0, \sigma_{lab}^2);\ i = 1, ..., 4 \\
\beta_{j:i}^{tec} &\sim N(0, \sigma_{tec}^2);\ j : i = 1, ..., 12 \\
\gamma_{k:(j:i)}^{sam} &\sim N(0, \sigma_{sam}^2);\ k : (j : i) = 1, ..., 24
\end{aligned}$$

El tercer nivel recibiría las distribuciones a priori para los hiperparámetros

$\sigma^2_{lab}, \sigma^2_{tec}, \sigma^2_{sam}$, sobre los que interesa inferir. A priori, con mínima información asumiremos $GaI(0.001, 0.001)$.

Para especificar en INLA los efectos anidados hemos de recurrir a la matriz del modelo, `model.matrix()`, para crear las matrices de los efectos aleatorios anidados.

A continuación hemos de crear los correspondientes índices, de longitud similar a la del número de registros en la base de datos, para aplicarles las correspondientes matrices de efectos anidados, y ya proceder con el ajuste como habitualmente hacemos.

```
# matrices de efectos aleatorios anidados
Zlt <- as(model.matrix( ~ 0 + Lab:Technician, data = eggs), "Matrix")
Zlts <- as(model.matrix( ~ 0 + Lab:Technician:Sample, data = eggs), "Matrix")

# índices para aplicar los efectos aleatorios
eggs$IDt = eggs$IDts = 1:nrow(eggs)

# Ajuste
prec.prior=list(prec=list(param=c(0.001,0.001)))
formula = Fat ~ 1 + f(Lab,model="iid",hyper=prec.prior) +
  f(IDt,model="z",Z=Zlt,hyper=prec.prior)+
  f(IDts,model="z",Z=Zlts,hyper=prec.prior)

fit <- inla(formula,data = eggs,
            control.predictor = list(compute = TRUE),
            control.family=list(hyper=prec.prior),
            control.fixed=list(prec.intercept=0.001))
# inferencias de interés
round(fit$summary.hyperpar,4)
#>                                             mean       sd
#> Precision for the Gaussian observations 142.0221  39.6763
#> Precision for Lab                       349.8905 651.0955
#> Precision for IDt                       206.0418 210.4896
#> Precision for IDts                      366.9427 335.2837
#>                                         0.025quant 0.5quant
#> Precision for the Gaussian observations    77.8353 137.4884
#> Precision for Lab                          22.4901 170.1977
#> Precision for IDt                          26.6549 144.1431
#> Precision for IDts                         59.8945 270.7277
#>                                         0.975quant mode
#> Precision for the Gaussian observations   232.9086   NA
#> Precision for Lab                        1808.5093   NA
#> Precision for IDt                         762.1928   NA
#> Precision for IDts                       1256.5620   NA
```

```
nhyp=length(names(fit$marginals.hyperpar))
res=NULL
for(j in 1:nhyp){
    res=rbind(res,data.frame(
    inla.tmarginal(function(tau) tau^(-1/2),fit$marginals.hyperpar[[j]]),
                          id=str_sub(names(fit$marginals.hyperpar)[j], start =15)))
}
ggplot(res,aes(x=x,y=y))+
  geom_line(aes(color=id))+
  labs(x=expression(sigma),y="")+
  theme(legend.position="top",
        legend.title=element_blank(),
        legend.text = element_text(size=5))
```

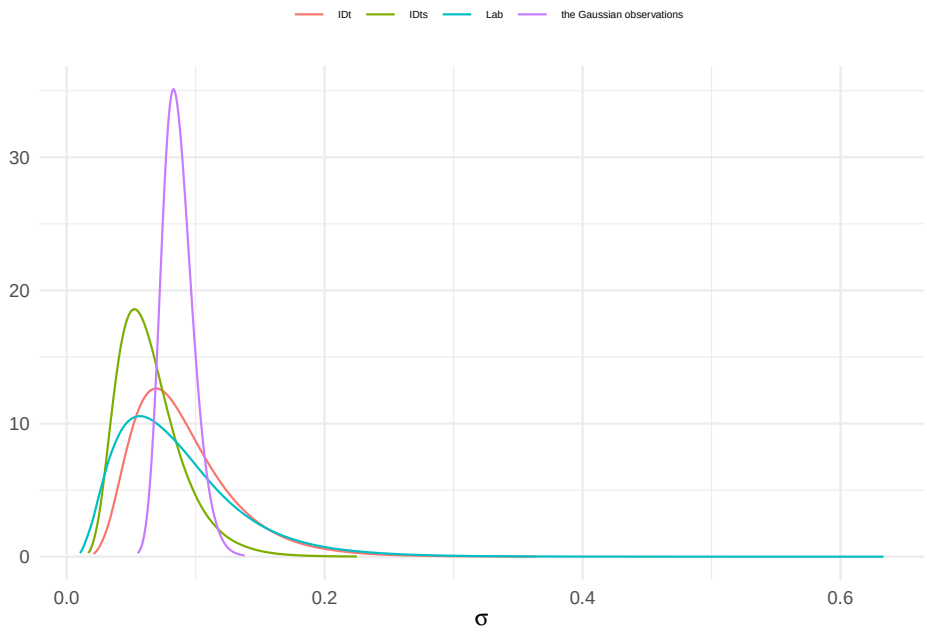


Figure 3.14: Distribución posterior del error de los datos y el error aleatorio

Alternativamente podríamos crear una variable índice a partir de las matrices de efectos aleatorios, para utilizarlas con `model="iid"` para describir los efectos aleatorios:

```
eggs$labtech <- as.factor(apply(Zlt, 1, function(x){names(x)[x == 1]}))
eggs$labtechsamp <- as.factor(apply(Zlts, 1, function(x){names(x)[x == 1]}))

formula=Fat ~ 1 + f(Lab, model = "iid", hyper = prec.prior) +
```

```
    f(labtech, model = "iid", hyper = prec.prior) +
  f(labtechsamp, model = "iid", hyper = prec.prior)
fit=inla(formula, data = eggs,
            control.predictor = list(compute = TRUE),
            control.family=list(hyper=prec.prior),
         control.fixed=list(prec.intercept=0.001))
round(fit$summary.fixed,4)
#>               mean     sd 0.025quant 0.5quant 0.975quant
#> (Intercept) 0.3875 0.0554     0.2756   0.3875     0.4994
#>             mode kld
#> (Intercept)   NA   0
round(fit$summary.hyperpar,4)
#>                                                mean       sd
#> Precision for the Gaussian observations 141.9183  39.6197
#> Precision for Lab                       349.8687 651.1198
#> Precision for labtech                   206.0426 210.4997
#> Precision for labtechsamp               366.9591 335.2916
#>                                         0.025quant 0.5quant
#> Precision for the Gaussian observations    77.8083 137.3958
#> Precision for Lab                          22.4884 170.1766
#> Precision for labtech                      26.6547 144.1403
#> Precision for labtechsamp                  59.8978 270.7425
#>                                         0.975quant mode
#> Precision for the Gaussian observations   232.6622   NA
#> Precision for Lab                        1808.4661   NA
#> Precision for labtech                     762.2166   NA
#> Precision for labtechsamp                1256.5985   NA
```

En la Figura 3.15 se muestra la distribución posterior del error de los datos y de los errores aleatorios.

```
nhyp=length(names(fit$marginals.hyperpar))
res=NULL
for(j in 1:nhyp){
    res=rbind(res,data.frame(
    inla.tmarginal(function(tau) tau^(-1/2),fit$marginals.hyperpar[[j]]),
                          id=str_sub(names(fit$marginals.hyperpar)[j], start =15)))
}
ggplot(res,aes(x=x,y=y))+
  geom_line(aes(color=id))+
  labs(x=expression(sigma),y="")+
  theme(legend.position="top",
        legend.title=element_blank(),
        legend.text = element_text(size=5))
```

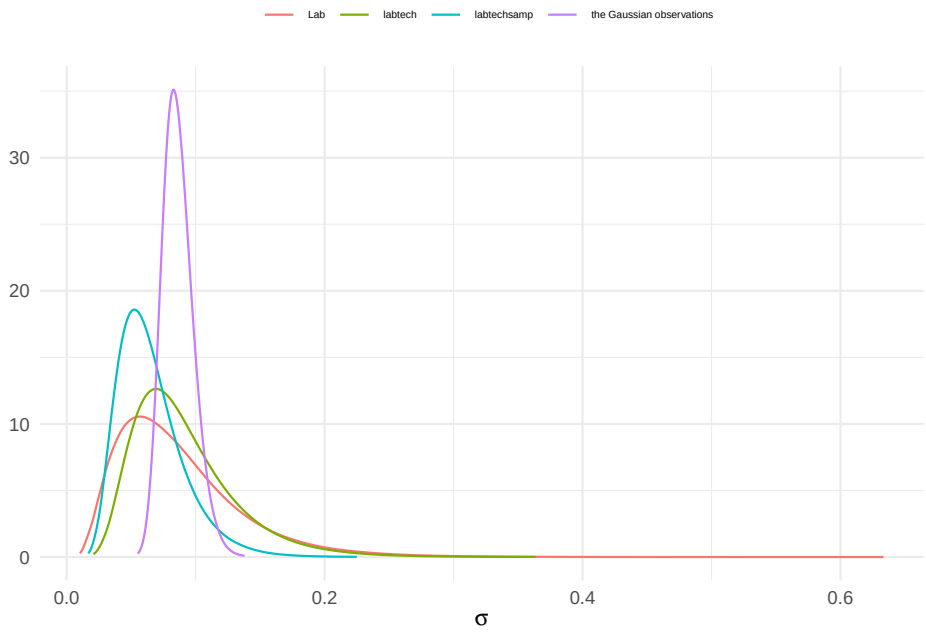


Figure 3.15: Distribución posterior del error de los datos y el error aleatorio

3.8 Conclusiones

Hasta aquí desarrollamos los modelos lineales basados en Anova, esto es, en la integración de factores de clasificación como variables que van a explicar diferencias en la respuesta, como los efectos fijos, o variabilidad extra en los datos, como los efectos aleatorios, a veces incluso con otros predictores de tipo numérico, e incluso interaccionando con ellos.

Chapter 4

Modelos lineales generalizados

Los modelos lineales generalizados (Generalized Linear Models or GLM), son una clase de modelos introducidos por Nelder y Wedderburn (1972) y McCullagh y Nelder (1989), con el objetivo de extender la regresión lineal al caso en el que la variable dependiente no se distribuya necesariamente según una normal, pero su distribución todavía pertenezca a la familia exponencial (Binomial, Poisson, Gamma, Gausiana inversa básicamente). Trabajamos a continuación con dos de los GLM más comunes en epidemiología y ciencias sociales: la regresión logística y la de Poisson, mostrando cómo usar R-INLA.

Un modelo lineal generalizado está basado en asumir, además de una distribución de los datos dentro de la familia exponencial, una relación lineal entre cierta transformación del valor esperado de la respuesta $E(y_i)$ y los predictores disponibles, sean covariables, efectos fijos, efectos aleatorios, o incluso alguna función de estos.

Si y representa una respuesta observada, $x_1, x_2, ..$ una serie de covariables o efectos fijos, y $z_1, z_2, ...$ efectos aleatorios, el valor esperado de la respuesta lo denotamos como μ, que

$$E(y_i|x, z, \theta) = \mu_i$$

Pues bien, la relación entre esta media μ y un predictor lineal η que construimos a partir de los predictores disponibles, viene dado por una función link g tal que:

$$g(\mu_i) = \eta_i = \mu + \beta_1 x_{1i} + \beta_2 x_{2i} + ... + z_{1i} + z_{2i} + ...$$

Todos los parámetros involucrados en el predictor lineal η son los efectos latentes del modelo (fijos o aleatorios). Estos, junto con el resto de parámetros definidos

en este primer nivel de la modelización (nivel de datos), han de modelizarse a continuación, en un segundo nivel del modelo, con sus correspondientes distribuciones a priori.

El predictor lineal η está relacionado linealmente con los predictores según:

$$\eta_i = \mu + \beta_1 x_{1i} + \beta_2 x_{2i} + ... + z_{1i} + z_{2i} + ...$$

relación que se suele representar en forma matricial como

$$\eta = X\beta + Zu$$

Los modelos lineales que hemos visto antes (regresión, anova, ancova, modelos mixtos) se engloban dentro del modelo lineal generalizado.

El argumento `control.predictor=list(compute=TRUE)` en la función `inla` permite obtener las distribuciones predictivas para el predictor lineal, que en estos casos es distinto a los valores ajustados, `fit$summary.fitted`. Además para obtener la distribución marginal de los valores ajustados y predichos necesitamos incorporar a la función `inla` el argumento `control.compute=list(return.marginals.predictor=TRUE)`, y ya con todo ello podemos:

- `fit$summary.linear.predictor` resumir la inferencia posterior sobre los predictores lineales (distintos a los fitted cuando hay una función *link*)
- `fit$marginals.linear.predictor` graficar y describir las distribuciones posteriores marginales para los predictores lineales

4.1 Modelos jerárquicos bayesianos

A lo largo del curso ya hemos ido comentando en ocasiones, algo sobre la especificación de un modelo en varios niveles. Presentamos ya de lleno estos modelos lineales generalizados como modelos multi-nivel o modelos jerárquicos, denominados así porque se va especificando por niveles (o jerarquías) la información disponible sobre todo aquello que es desconocido, distribución de los datos y parámetros.

Un modelo bayesiano se modeliza a través de un modelo jerárquico o multinivel en el que en el nivel I se define la distribución asumida sobre la variable respuesta y que determina la verosimilitud. Esta variable depende de unos parámetros que definen los efectos fijos y aleatorios, y para los que hay que proporcionar la información previa disponible a través de una distribución a priori en el segundo nivel del modelo jerárquico. La distribución a priori para los efectos fijos generalmente será común a todos ellos, mientras que la distribución a priori para los efectos aleatorios estará vinculada a otros hiperparámetros para los que

también será preciso especificar una distribución a priori en un tercer nivel de la modelización, y así sucesivamente.

$$
\begin{array}{rcl}
Nivel I & & \\
(y|X,Z,\theta) & \sim & f(y|x,z,\theta)\text{f en fam.exponencial} \\
 & & E(y|x,z,\theta)=\mu; Var(y|x,z,\theta)=\Sigma \\
 & & g(\mu)=\eta=X\beta+Zu \\
Nivel II & & \\
\beta & \sim & N(0,\sigma_\beta),\ \text{con un valor dado para } \sigma_\beta \\
u|\sigma_u^2 & \sim_{iid} & N(0,\sigma_u^2) \\
\Sigma|s & \sim & F_{\Sigma|s} \\
Nivel III & & \\
\sigma_u^2 & \sim & F_\sigma \\
s & \sim & F_s
\end{array}
$$

4.2 Regresión logística

La regresión logística es el modelo estándar para respuestas binarias (éxitos/fracasos). Tiene dos variaciones, en función de si la respuesta representa observaciones individuales (0/1) o conteos (de éxitos) en grupos de sujetos.

Si las observaciones son individualizadas, entonces

$$y_i|\pi_i \sim Ber(\pi_i),\ i=1,...,n$$

En el caso de que sean conteos en grupos,

$$y_i|\pi_i \sim Bin(n_i,\pi_i),\ i=1,...,n$$

siendo n_i el tamaño de cada uno de los n grupos disponibles, y π_i la probabilidad de éxito (output de interés).

La relación entre el predictor lineal η construido con los predictores disponibles $x=(x_1,...x_M)$ y la probabilidad π se especifica a través de la función *logit*:

$$logit(\pi)=log\left(\frac{\pi}{1-\pi}\right)=\eta=X\beta=\beta_0+\sum_{j=1}^{M}\beta_j x_j$$

de forma que

$$\pi=logit^{-1}(X\beta)=\frac{exp(X\beta)}{1-exp(X\beta)}$$

Una vez especificado el modelo, si no hay información previa disponible sobre los efectos (fijos) $\beta_o, \beta_1, ...\beta_M$, se asumen distribuciones a priori independientes y normales con media cero y varianza muy grande.

Interpretación de los coeficientes en la regresión logit

Puesto que $X\beta = \beta_0 + \sum_{j=1}^{M} \beta_j x_j$, la interceptación del predictor lineal β_0 se interpreta como los predictores toman el valor cero si son numéricos, o están en el nivel de referencia (para la estimación) si son categóricos, $\eta(X = 0) = \beta_0 = logit(\pi)$. En consecuencia, el logit inverso de β_0 se interpreta como la probabilidad de éxito π_i cuando los predictores están en su nivel de referencia o son cero.

$$logit^{-1}(\beta_0) = Pr(y = 1|X = 0).$$

En cuanto a la interpretación de cualquier otro coeficiente de regresión en el predictor lineal, como β_1, echamos mano del concepto de *odds* y *odds ratio*.

Los odds ratio, OR, comparan, a través de un cociente, las posibilidades a favor de un evento E bajo condiciones A y de las posibilidades del mismo evento bajo condiciones B. Nos sirve para evaluar cuánto afecta a dicho evento el hecho de variar las condiciones de B a A.

$$OR(A, B) = \frac{Pr(E|A)/(1 - Pr(E|A))}{Pr(E|B)/(1 - Pr(E|B))}.$$

En el modelo logístico, nos interesa saber el efecto que tiene sobre la respuesta (realmente sobre las probabilidad de éxito) el incremento de una unidad en la variable x, y para ello consideramos los odds bajo x y los odds bajo $x + 1$, y en particular el logaritmo de los odds, log-odds:

$$log.odds(x + 1) = log\left(\frac{P(y = 1|x + 1)}{P(y = 0|x + 1)}\right) = log\left(\frac{\pi}{1 - \pi}|x + 1\right) = \beta_0 + \beta_1(x + 1)$$

$$log.odds(x) = log\left(\frac{P(y = 1|x)}{P(y = 0|x)}\right) = log\left(\frac{\pi}{1 - \pi}|x\right) = \beta_0 + \beta_1 x$$

de modo que

$$log(OR(x + 1, x)) = log\left(\frac{ods(x + 1)}{ods(x)}\right) = log.odds(x + 1) - log.ods(x) = \beta_1$$

y tenemos entonces que la exponencial del coeficiente β_1 nos da el odds ratio asociado a dicha covariable.

$$exp(\beta_1) = \frac{odds(x + 1)}{odds(x)} = OR(x + 1, x).$$

Es decir, el coeficiente β_1 representa el cambio en los odds a favor de un éxito cuando se incrementa en una unidad el predictor x al que acompaña en el predictor linea. Esta interpretación es muy común en Epidemiología.

4.2.1 Intención de voto feb2022

Tenemos acceso a los datos completos obtenidos en la encuesta encargada por El País y la Cadena Ser a la empresa "40dB", en febrero de 2022, sobre la intención de voto nacional (fuente).

Queremos predecir la probabilidad de votar al partido que gobierna mayoritariamente en la actualidad, PSOE, registrado en la variable `psoe`. Vamos a utilizar como predictores dos factores que nos dicen si el sujeto tiene simpatía por ese partido, `psoe_sim`, y si votó PSOE en las últimas elecciones `psoe_past`; también utilizaremos la comunidad autónoma `ccaa` como un efecto aleatorio, para contabilizar posible variación extra.

```
url="https://raw.githubusercontent.com/BayesModel/data/main/barometro_feb22.csv"
barometro_feb22=read.csv(url)
names(barometro_feb22)
#>  [1] "X"                    "id"
#>  [3] "sexo"                 "edad"
#>  [5] "edad_r"               "hab"
#>  [7] "prov"                 "ccaa"
#>  [9] "edu"                  "cs"
#> [11] "p1"                   "p2"
#> [13] "p3"                   "p4_1"
#> [15] "p4_2"                 "p4_3"
#> [17] "p4_4"                 "p5"
#> [19] "p6"                   "p7"
#> [21] "hab_r"                "clase_social_r"
#> [23] "situacion_laboral_r" "educacion_r"
#> [25] "ponde"
datos=barometro_feb22 %>%
  select(id,p2,p3,p5,ccaa) %>%
  mutate(psoe=1*(p2=="PSOE (Partido Socialista Obrero Español)"),
         psoe_simp=1*(p3=="PSOE (Partido Socialista Obrero Español)"),
         psoe_past=1*(p5=="PSOE (Partido Socialista Obrero Español)"))
#summary(datos)
```

Para ajustar el modelo utilizamos el argumento `family=binomial` y la opción `control.predictor = list(link = 1)` para establecer la función link apropiada para tener los valores ajustados en la escala correcta.

```
prec.prior=list(prec=list(param=c(0.001,0.001)))
formula = psoe ~ psoe_simp + psoe_past+ f(ccaa,model="iid",hyper=prec.prior)
fit=inla(formula,family="binomial",data=datos,control.predictor = list(link = 1))
fit$summary.fixed
#>                    mean        sd 0.025quant  0.5quant
#> (Intercept) -3.590549 0.1897354  -3.998915 -3.578265
```

```
#> psoe_simp    3.085795 0.1865848   2.723898  3.084361
#> psoe_past    2.352348 0.1856950   1.989698  2.351803
#>             0.975quant mode          kld
#> (Intercept)  -3.251318   NA 5.820554e-07
#> psoe_simp     3.455832   NA 3.431097e-07
#> psoe_past     2.718088   NA 9.428328e-07
fit$summary.hyperpar
#>                         mean       sd 0.025quant 0.5quant
#> Precision for ccaa 70.82681 251.2122   1.979164 10.70692
#>                    0.975quant mode
#> Precision for ccaa   588.1799   NA
```

Las distribuciones posteriores de los exponenciales de los efectos aleatorios se muestran en la Figura 4.1, identificadas en verde las de efectos positivos en la media posterior del predictor lineal (log-odds > 1), y en rojo las de efectos negativos, e interpretables como más y menos favorables a votar por el PSOE.

```
random = as.data.frame(fit$summary.random)
random$pro=1*exp(random$ccaa.mean)>1
ggplot(random,aes(x=exp(ccaa.mean),y=ccaa.ID)) +
  geom_point(aes(color=pro))+
  geom_errorbarh(aes(xmin=exp(ccaa.0.025quant),xmax=exp(ccaa.0.975quant),color=pro))+
  geom_vline(xintercept=1,linetype="dotted")+
  labs(x="Medias y RC posteriores para el log-odds",y="Comunidad autónoma")+
  theme(legend.position="none")
```

La distribución posterior de la probabilidad de voto para el PSOE para la comunidad con más variabilidad en el efecto aleatorio (Castilla-La Mancha), viene representada en la Figura 4.2 para las cuatro combinaciones posibles de valores para los predictores de simpatía y voto en el pasado.

```
datos_pred = datos %>%
  mutate(f.post=round(fit$summary.fitted.values$mean,3),
         f.rc.low=round(fit$summary.fitted.values$"0.025quant",3),
          f.rc.up=round(fit$summary.fitted.values$"0.975quant",3)) %>%
  distinct(f.post,.keep_all = TRUE) %>%
  filter(ccaa=="Castilla - La Mancha") %>%
  mutate(simpast=str_c(psoe_simp,psoe_past))
ggplot(datos_pred,aes(x=f.post,y=simpast))+
  geom_point(aes(color=simpast))+
  geom_errorbarh(aes(xmin=f.rc.low,xmax=f.rc.up,color=simpast),height=0.2)+
  labs(x="Medias y RC posteriores para la probabilidad de voto PSOE",title="Castilla -
  scale_y_discrete(name="",
                   labels=c("No simpatía/No votó PSOE","No simpatía/Votó PSOE",
```

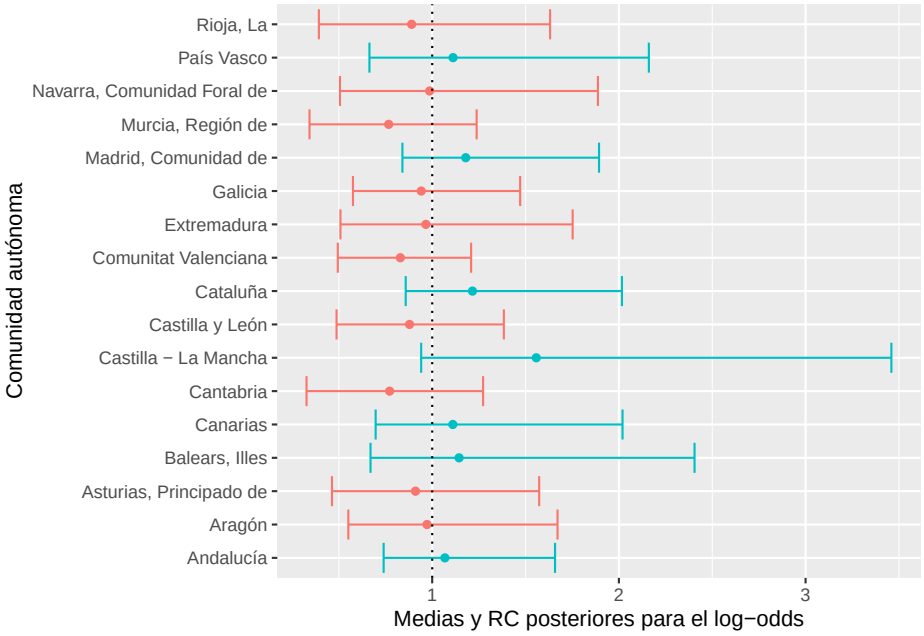


Figure 4.1: Medias y RC posterioris para el log-odds de los efectos aleatorios

```
                              "Simpatía/No votó PSOE","Simpatía/Votó PSOE"))+
  theme(legend.position="none")
```

4.2.2 Mortalidad por infarto en Sheffield

Utilizamos los datos *stroke*, disponibles en datasets in SSTM-RINLA. Queremos evaluar la presencia de cierta asociación entre los niveles de NOx y el infarto en Sheffield, UK. Se dispone de la concentración anual de NOx medida en $\mu g/m^3$ y categorizada en quintiles, promediada durante el periodo 1994-1999, en la variable *NOx.class* y su análoga NOx, y el número de muertes por infarto y en cada distrito identificado por el índice de desventajas y privación *Townsend* (categorizado en quintiles). Se dispone igualmente del tamaño de la población para cada registro, en la variable *pop*. La respuesta y se puede considerar entonces como conteos (de muertes) sobre la población de cada distrito,

$$y_i|\pi_i \sim Bin(n_i, \pi_i)$$

y el predictor lineal es función del nivel de NOx y del distrito:

$$\eta_i = logit(\pi_i) = \beta_0 + \sum_{k=2}^{5} \beta_{1k} I(NOx_i = k) + \sum_{h=2}^{5} \beta_{2h} I(Townsend_i = h) + logit(\tilde{p_i})$$

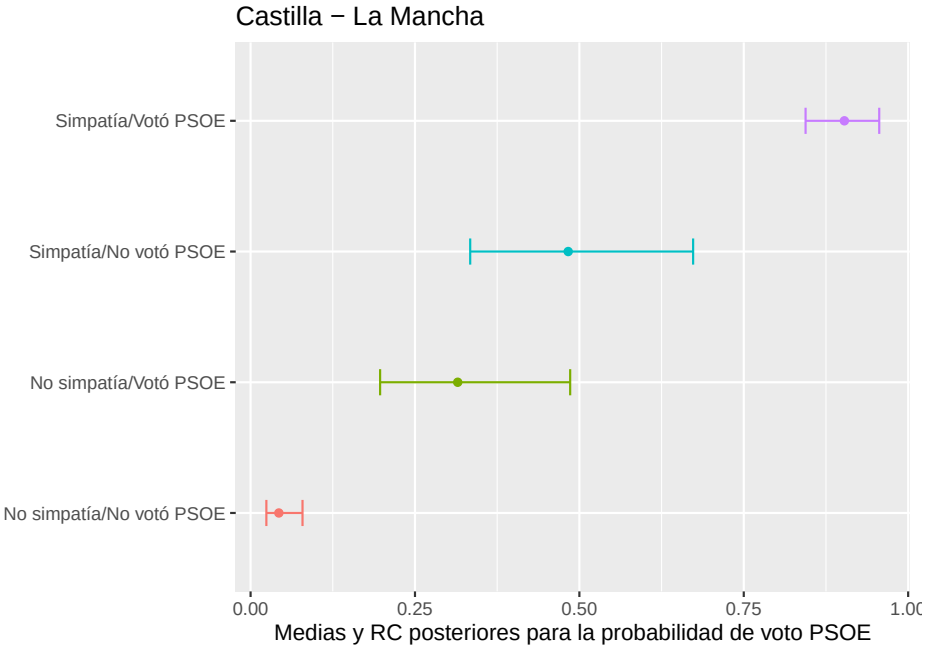


Figure 4.2: Medias y RC posteriores para la probabilidad de voto PSOE

siendo n_i la población (número total de habitantes) del distrito en el que se ubica el registro i (disponible en la variable *pop*). El término $\tilde{p}_i$ representa el resgo ajustado por sexo y edad de la mortalidad por infarto, calculada utilizando estandarización indirecta con ratios de referencia internos basados en 18 estratos (9 para edad y 2 para género), y que se usa como un riesgo base en el modelo (Maheswaran et al.2006). En el ejemplo se calcula como el ratio de la mortalidad dividido por la población de cada registro.

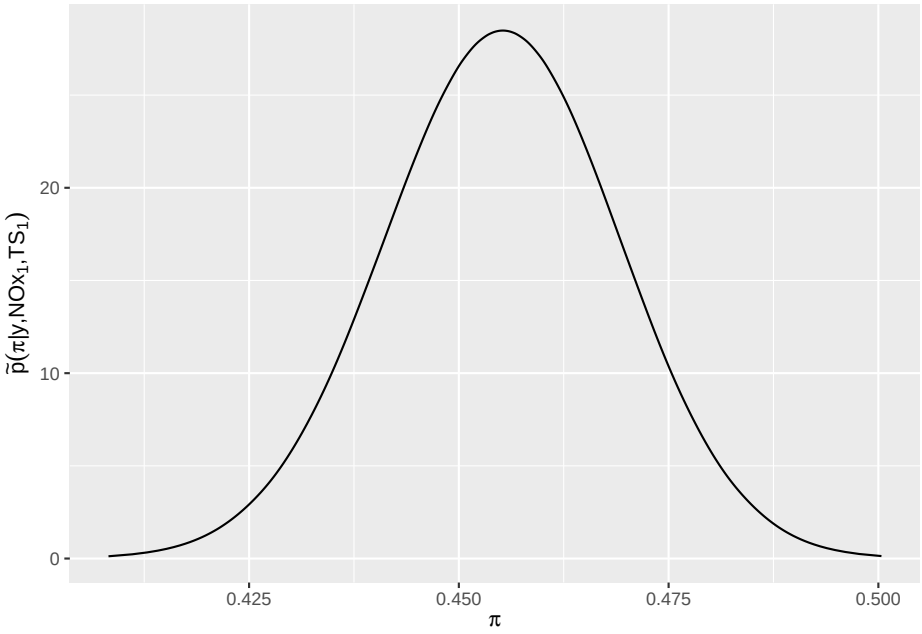
```
url="https://raw.githubusercontent.com/BayesModel/data/main/Stroke.csv"
Stroke <- read.csv(url,sep=",",dec=".",header=TRUE)
#riesgo base: ajuste por tamaño de la población
Stroke$Adjusted.prob <- Stroke$stroke_exp/Stroke$pop
# logit del riesgo base
Stroke$logit.adjusted.prob <- log(Stroke$Adjusted.prob/(1-Stroke$Adjusted.prob))
```

Ajustamos ya el modelo, en el que las variables NOx y Townsend actúan como factores (efectos fijos) y el riesgo base se introduce como offset, para estandarizar los riesgos en función del tamaño de la población y poder equiparar así todos los distritos:

```
formula.inla <- y ~ 1 + factor(NOx) + factor(Townsend) + offset(logit.adjusted.prob)
model.logistic <- inla(formula.inla, family="binomial", Ntrials=pop, data=Stroke)
round(model.logistic$summary.fixed[,1:5],3)
#>                      mean    sd 0.025quant 0.5quant
#> (Intercept)        -0.181 0.057     -0.293   -0.180
#> factor(NOx)2        0.132 0.059      0.016    0.132
#> factor(NOx)3        0.105 0.061     -0.014    0.105
#> factor(NOx)4        0.261 0.059      0.144    0.260
#> factor(NOx)5        0.425 0.062      0.302    0.425
#> factor(Townsend)2   0.077 0.061     -0.043    0.077
#> factor(Townsend)3   0.137 0.060      0.020    0.137
#> factor(Townsend)4  -0.132 0.063     -0.255   -0.132
#> factor(Townsend)5  -0.118 0.067     -0.250   -0.118
#>                    0.975quant
#> (Intercept)            -0.071
#> factor(NOx)2            0.248
#> factor(NOx)3            0.225
#> factor(NOx)4            0.377
#> factor(NOx)5            0.547
#> factor(Townsend)2       0.198
#> factor(Townsend)3       0.255
#> factor(Townsend)4      -0.009
#> factor(Townsend)5       0.014
```

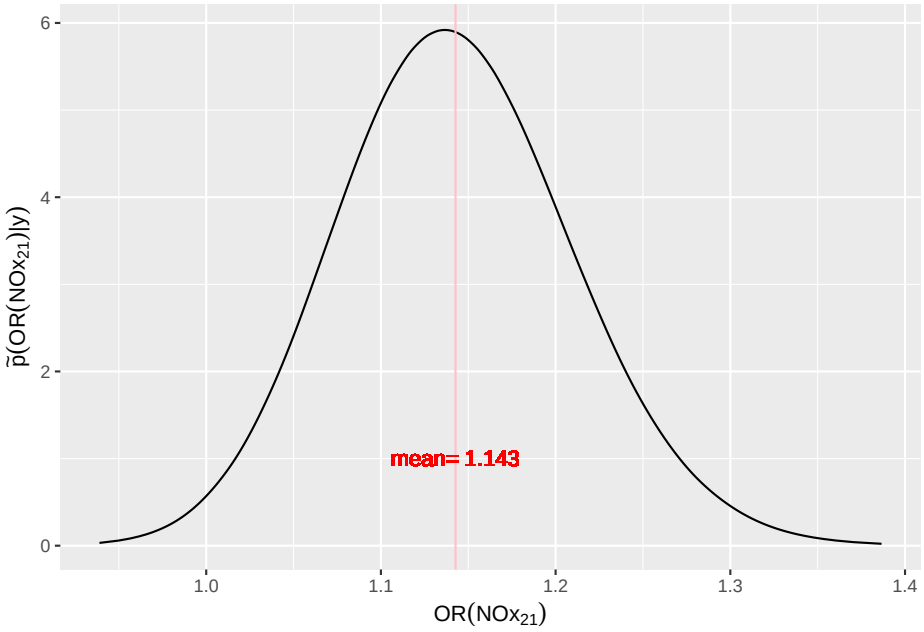
Para obtener la probabilidad promedio de muerte por infarto en el distrito Townsend=1 y para el nivel NOx=1, que son los niveles base, nos apoyamos en la interceptación β_0. Sobre sus simulaciones será preciso deshacer el logit (con la función logit-inversa):

```
prob.stroke <- inla.tmarginal(function(x) exp(x)/(1+exp(x)), model.logistic$marginals.fixed[[1]])
inla.zmarginal(prob.stroke)
#> Mean            0.455034
#> Stdev           0.0139168
#> Quantile  0.025 0.427486
#> Quantile  0.25  0.445615
#> Quantile  0.5   0.455082
#> Quantile  0.75  0.464485
#> Quantile  0.975 0.482137
ggplot(data.frame(prob.stroke),aes(x=x,y=y))+geom_line()+
        labs(x=expression(pi),y= expression(tilde(p)(paste(pi,"|",y,",",NOx[1],",",TS[1]))))
```



El efecto β_{12} representa el efecto en los log.odds de la mortalidad por infarto de estar en el nivel $NOx = 2$ frente al de estar en el nivel $NOx = 1$. Si queremos evaluar el odds-ratio, simplemente calculamos la distribución posterior de $exp(\beta_{12})$ con `inla.tmarginal`. Si sólo estamos interesados en su media, bastaría utilizar 'inla.emarginal':

```
odds.nox21 <- inla.tmarginal(function(x) exp(x), model.logistic$marginals.fixed$"factor
e<-inla.emarginal(exp, model.logistic$marginals.fixed$"factor(NOx)2")
ggplot(data.frame(odds.nox21),aes(x=x,y=y))+geom_line()+
        labs(x=expression(OR(NOx[21])),y= expression(tilde(p)(paste(OR(NOx[21]),"|",y)
        geom_vline(xintercept=e,color="pink")+ geom_text(x=e,y=1,label=paste("mean=",r
```



La probabilidad de muerte por infarto se incrementa en un 14.3% cuando la exposición de NOx cambia del primer al segundo nivel. Los odds-ratio para el resto de los niveles resultan realmente significativos: 11,3% el odds-ratio NOx3/NOx1, 30% NOx4/NOx1 y 53.2% NOx5/NOx1.

```
inla.emarginal(exp, model.logistic$marginals.fixed$"factor(NOx)3")
#> [1] 1.113128
inla.emarginal(exp, model.logistic$marginals.fixed$"factor(NOx)4")
#> [1] 1.299909
inla.emarginal(exp, model.logistic$marginals.fixed$"factor(NOx)5")
#> [1] 1.532265
```

4.3 Regresión de Poisson

La regresión de Poisson es útil cuando la variable respuesta representa conteos y estos toman valores discretos entre 0 y $+\infty$, sin una cota superior de referencia. El parámetro de interés es el número promedio de eventos $\lambda = E(y)$ y el link natural es el logaritmo, de modo que el predictor lineal está ligado con las covariables y factores según:

$$\eta = log(\lambda) = x\beta, \quad \text{y} \quad \lambda_i = exp(X\beta)$$

Un modelo de Poisson puede especificarse según

$$y_i \sim Po(\lambda_i), i = 1, ..., n$$

$$\eta_i = log(\lambda_i) = \beta_0 + \sum_{m=1}^{M} \beta_m x_{im}.$$

Para completar el modelo se especifican distribuciones a priori para β, típicamente como normales con media cero y una varianza grande cuando no hay información disponible de estudios previos u opinión de expertos.

Los coeficientes se interpretan a través de la función exponencial:

- $exp(\beta_0) = \lambda_i$ cuando todas las $x = 0$ si son continuas, o para el primer nivel de las categorías posibles si son categóricas.
- $exp(\beta_m)$ es el cambio que se produce en la respuesta promedio y cuando x_m se incrementa en una unidad.

La mayoría de las veces que se utiliza la regresión de Poisson, el interés recáe en las ratios o riesgos relativos, más que en el número promedio de casos λ_i. Para cambiar la escala en términos de riesgo, ha de utilizarse un offset como factor de corrección en la especificación del modelo. Este offset representa el denominador del riesgo y entra en la regresión en una escala logarítmica, asumiendo que tiene un coeficiente de regresión fijado a 1:

$$\eta_i = log(\lambda_i) = \beta_0 + \sum_{m=1}^{M} \beta_m x_{im} + log(Offset_i)$$

donde el riesgo relativo de que se produzca un evento se obtiene según

$$log\left(\frac{\lambda_i}{Offset_i}\right) = \beta_0 + \sum_{m=1}^{M} \beta_m x_{im}$$

y los coeficientes entonces se interpretan en una escala de riesgo. En este caso al exponenciar la interceptación obtenemos el riesgo base, mientras que $exp(\beta_m)$ representa el cambio en el riesgo relativo debido a un cambio de unidad en el predictor correspondiente.

4.3.1 Incidentes en barcos

Utilizamos los datos *ships.csv* en datasets for SSTM-RINLA para estimar el riesgo mensual de incidentes en barcos. Los factores potenciales del riesgo son el periodo de construcción (*built*), el periodo de operación (*oper*) y el tipo de barco (*type*). El modelo se escribe en INLA a continuación, utilizando como offset el *log(months)*, que son los meses que ha navegado y ponderan en consecuencia el riesgo de incidentes. El modelo con el offset será entonces

$$y_i \sim Poisson(E_i \rho_i),$$

donde $\eta_i = log(\rho_i)$ es el predictor lineal y el promedio del número de incidentes $\lambda_i = E_i \rho_i$. El offset no se incluye en esta formulación en el predictor lineal.

```
url="https://raw.githubusercontent.com/BayesModel/data/main/Ships.csv"
ShipsIncidents <- read.csv(url,sep=",")

formula.inla <- y ~ 1 + built + oper + type
model.poisson <- inla(formula.inla,family="poisson", data=ShipsIncidents, offset=log(months))

round(model.poisson$summary.fixed[,1:5],3)
#>               mean    sd 0.025quant 0.5quant 0.975quant
#> (Intercept) -6.416 0.217     -6.852   -6.413     -5.998
#> built65-69   0.696 0.150      0.406    0.695      0.994
#> built70-74   0.819 0.170      0.487    0.818      1.153
#> built75-79   0.453 0.233     -0.012    0.455      0.903
#> oper75-79    0.384 0.118      0.153    0.384      0.617
#> typeB       -0.543 0.178     -0.882   -0.546     -0.185
#> typeC       -0.688 0.329     -1.366   -0.678     -0.075
#> typeD       -0.075 0.291     -0.664   -0.069      0.476
#> typeE        0.326 0.236     -0.141    0.327      0.785
```

```
names(model.poisson$marginals.fixed)
#> [1] "(Intercept)" "built65-69"  "built70-74"  "built75-79"
#> [5] "oper75-79"   "typeB"       "typeC"       "typeD"
#> [9] "typeE"
# ratio medio de incidentes por mes en las categorías base
inla.emarginal(exp,model.poisson$marginals.fixed[[1]])
#> [1] 0.00167416
# riesgo relativo de barcos tipo E
inla.emarginal(exp,model.poisson$marginals.fixed$typeE)
#> [1] 1.424414
```

Así, la media de $exp(\beta_0)$, 0.0018, representa el ratio medio de incidentes por mes entre los barcos que fueron construidos entre el 60 y el 64, han operado entre el 60 y el 74 y son de tipo A (las categorías de referencia). El ratio en 1000 meses sería del 1.8.Para los barcos de tipo E el incremento en el ratio mensual de incidentes, comparado con los de tipo A es del 42,4%.

Otros datos modelizables con una regresión de Poisson son los que provienen del libro de Andrews and Herzberg y están descritos en (randomservices.org/random) consistentes en el número de soldados muertos por coces de caballo en diversos cuerpos de caballería del ejército prusiano, entre 1875 y 1894.

Este modelo se implementa en INLA, a partir de datos simulados, con el siguiente código

```
url="https://raw.githubusercontent.com/BayesModel/data/main/HorseKicks.txt"
horse<-read.csv(url,sep="", dec=".",header=TRUE)
horse$sum<-apply(horse[,2:ncol(horse)],1,sum)

fit=inla(sum~1,data=horse,family="poisson",control.predictor=list(compute=TRUE))
summary(fit)
#>
#> Call:
#>    c("inla.core(formula = formula, family = family,
#>    contrasts = contrasts, ", " data = data, quantiles =
#>    quantiles, E = E, offset = offset, ", " scale =
#>    scale, weights = weights, Ntrials = Ntrials, strata =
#>    strata, ", " lp.scale = lp.scale, link.covariates =
#>    link.covariates, verbose = verbose, ", " lincomb =
#>    lincomb, selection = selection, control.compute =
#>    control.compute, ", " control.predictor =
#>    control.predictor, control.family = control.family,
#>    ", " control.inla = control.inla, control.fixed =
#>    control.fixed, ", " control.mode = control.mode,
#>    control.expert = control.expert, ", " control.hazard
#>    = control.hazard, control.lincomb = control.lincomb,
#>    ", " control.update = control.update,
#>    control.lp.scale = control.lp.scale, ", "
#>    control.pardiso = control.pardiso, only.hyperparam =
#>    only.hyperparam, ", " inla.call = inla.call, inla.arg
#>    = inla.arg, num.threads = num.threads, ", "
#>    blas.num.threads = blas.num.threads, keep = keep,
#>    working.directory = working.directory, ", " silent =
#>    silent, inla.mode = inla.mode, safe = FALSE, debug =
#>    debug, ", " .parent.frame = .parent.frame)")
#> Time used:
#>     Pre = 2.26, Running = 0.165, Post = 0.00703, Total = 2.43
#> Fixed effects:
#>              mean    sd 0.025quant 0.5quant 0.975quant mode
#> (Intercept) 2.282 0.071      2.139    2.283       2.42   NA
#>             kld
#> (Intercept)   0
#>
#> Marginal log-Likelihood:  -61.30
#>  is computed
#> Posterior summaries for the linear predictor and the fitted values are computed
#> (Posterior marginals needs also 'control.compute=list(return.marginals.predictor=TR
```