COLEGIO DE POSTGRADUADOS

INSTITUCIÓN DE ENSEÑANZA E INVESTIGACIÓN
EN CIENCIAS AGRÍCOLAS

CAMPUS MONTECILLO

POSTGRADO EN SOCIOECONOMÍA, ESTADÍSTICA E
INFORMÁTICA
ESTADÍSTICA

MODELOS DE PREDICCIÓN ROBUSTOS CON DATOS MASIVOS: INFERENCIA Y ALTERNATIVAS AL MCMC CLÁSICO

DIANA YANIRA CAAMAL PAT

T E S I S

PRESENTADA COMO REQUISITO PARCIAL PARA
OBTENER EL GRADO DE:

DOCTORA EN CIENCIAS

MONTECILLO, TEXCOCO, ESTADO DE MÉXICO,
MÉXICO
2022

COLEGIO DE POSTGRADUADOS

INSTITUCIÓN DE ENSEÑANZA E INVESTIGACIÓN EN CIENCIAS AGRÍCOLAS

La presente tesis titulada: **"Modelos de predicción robustos con datos masivos: inferencia y alternativas al MCMC clásico"** realizada por la estudiante: **"Diana Yanira Caamal Pat"** bajo la dirección del Consejo Particular indicado, ha sido aprobada por el mismo y aceptada como requisito parcial para obtener el grado de:

DOCTORA EN CIENCIAS
SOCIOECONOMÍA, ESTADÍSTICA E INFORMÁTICA
ESTADÍSTICA

CONSEJO PARTICULAR

CONSEJERO — Pérez Rdz.
Dr. Paulino Pérez Rodríguez

CO-DIRECTOR
Dr. José F. Crossa Hiriart

ASESOR
Dr. Ciro Velasco Cruz

ASESOR
Dr. Sergio Pérez Elizalde

ASESOR
Dr. Mario Alberto Vázquez Peña

Montecillo, Texcoco, Estado de México, México, julio de 2022

MODELOS DE PREDICCIÓN ROBUSTOS CON DATOS MASIVOS: INFERENCIA Y ALTERNATIVAS AL MCMC CLÁSICO

Diana Yanira Caamal Pat, Dra.

Colegio de Postgraduados, 2022

RESUMEN

Los datos de alta dimensión han surgido gracias al desarrollo sostenido de las tecnologías que facilitan la generación y recolección de datos. Estos datos se caracterizan por tener una serie de covariables, p, mayores al tamaño de muestra, n, ($p \gg n$). El problema principal se presenta durante el ajuste de un modelo, en particular en los cálculos numéricos, asociados a la estimación de un gran número de componentes aleatorios. El modelo lineal mixto (MLM) es una alternativa para analizar datos agrupados, que incorpora efectos fijos y aleatorios. El ajuste del MLM se realiza mediante el enfoque clásico o bayesiano, en ambos casos generalmente se requiere de la implementación de algoritmos. En presencia de alta dimensionalidad, los algoritmos de ajuste se vuelve, en términos computacionales, muy intensos. Este trabajo presenta dos métodos para el ajuste del MLM con datos de alta dimensión. El primer método consiste en ajustar el MLM utilizando diferentes matrices de varianzas-covarianzas en los efectos fijos y aleatorios. Este método da como resultado el desarrollo del paquete lme4GS de R que proporciona estimaciones REML de los parámetros de interés. El segundo método consiste en aplicar la técnica de aumentación ortogonal de datos al MLM para su ajuste mediante el algoritmo esperanza-maximización. Para cada método se evaluó el poder predictivo del modelo y el tiempo de cómputo utilizando datos reales. Los resultados muestran que los métodos aquí propuestos son rápidos ya que los tiempos de cómputo son menores en comparación con los algoritmos basados en Cadenas de Markov Monte Carlo (MCMC), al menos en un 50 %. Las correlaciones entre las estimaciones de los parámetros de varianza mediante los métodos propuestos y los métodos MCMC fueron altas.

Palabras clave: modelo lineal mixto, dimensionalidad, algoritmos, lme4GS.

ROBUST PREDICTION MODELS WITH BIG DATA: INFERENCE AND ALTERNATIVES TO MCMC METHODS

Diana Yanira Caamal Pat, Dra.

Colegio de Postgraduados, 2022

ABSTRACT

High-dimensional data has emerged thanks to the sustained development of technologies that facilitate data generation and collection. Those data are characterized by a number of covariates p, larger than the sample size, n, ($p \gg n$). The main problem arises during the fitting of the model, particularly in the numerical calculations, associated with the estimation of a large number of random components. The linear mixed model (LMM) is an alternative to analyze clustered data that incorporate fixed and random effects. The LMM fit is performed by the classic and Bayesian approach, in both cases the implementation of algorithms is generally required. In the presence of high dimensionality, the fitting algorithms become very computationally intensive. This work presents two methods for fitting the LMM with high-dimensional data. The first method consists of fitting the LMM using different variance-covariance matrices in the fixed and random effects. This method results in the development of the `lme4GS` R package that provides REML estimates of the parameters of interest. The second method consists of applying the orthogonal data augmentation technique to the linear mixed model for its fit using the expectation-maximization algorithm. For each method, the prediction accuracy of the model and the computation time were evaluated using real data. The results show that the proposed methods are fast since the computation times are lower than Markov Chain Monte Carlo algorithms at least by 50 %. The correlations between the variance parameter estimates using the proposed methods and the MCMC methods were high.

Key words: linear mixed model, dimensionality, algorithms, lme4GS.

AGRADECIMIENTOS

Al Consejo Nacional de Ciencia y Tecnología (CONACyT), por el apoyo económico brindado para la realización de mis estudios de doctorado.

Al Colegio de Postgraduados, al campus Montecillo, al Postgrado de Socioeconomía, Estadística e Informática, y en especial al programa de Estadística por haberme brindado la oportunidad de seguir mi formación académica y profesional en sus aulas.

A los integrantes de mi consejo:

Dr. Paulino Pérez Rodríguez, por el apoyo incondicional durante mis estudios, por su excelente dirección para la realización del presente trabajo, por la amistad y consejos brindados.

Dr. José Luis Francisco Crossa Hiriart, por el apoyo durante mis estudios y acertadas sugerencias a esta investigación.

Dr. Ciro Velasco Cruz y Dr. Sergio Pérez Elizalde por las enseñanzas brindadas para mi formación académica y asesorías para la realización de la investigación.

Dr. Mario Alberto Vázquez Peña, por el apoyo y asesoría para la realización de la investigación.

A todos los profesores que contribuyeron en mi formación académica y al personal administrativo que me brindó su apoyo durante mis estudios en el Colegio de Postgraduados.

A todos mis compañeros y amigos que me han acompañado en esta etapa de mi vida.

DEDICATORIA

Con amor dedico este trabajo,

A mi esposo, por todo su apoyo y amor que me ha brindado para la culminación de mis estudios de doctorado.

A mi hija, por llegar a mi vida.

A mis padres, por su apoyo, amor y por alentarme a seguir siempre adelante.

A mis hermanos, por su apoyo y amor.

A mis suegros, por su apoyo y cariño que nos han brindado.

◇

CONTENIDO

LISTA DE CUADROS

LISTA DE FIGURAS

CAPÍTULO 1. INTRODUCCIÓN

El desarrollo sostenido de nuevas tecnologías, recursos de almacenamiento de datos y recursos informáticos da lugar a la producción, almacenamiento y procesamiento de un volumen de datos con crecimiento exponencial (Giraud, 2015). Los datos tienen un impacto importante en todas las actividades humanas, desde la ciencia animal, la medicina, la agricultura, los negocios, las finanzas, hasta la administración. Una característica importante de los datos modernos es que a menudo se registran simultáneamente miles y millones de características en cada objeto o individuo por lo que se dice que estos datos son de alta dimensión (Giraud, 2015). Los datos a gran escala o de alta dimensión se caracterizan por tener una serie de covariables, p, mayores al tamaño de muestra n, es decir $p \gg n$ (Gueuning, 2017), y se pueden encontrar en diversos campos como la genética, la agricultura, las finanzas, la ecología, la salud y el procesamiento de imagenes (Gueuning, 2017). El análisis de datos de alta dimensión permite evaluar y comprender mejor los fenómenos de estudio, lo que resulta imprescindible para crear nuevas tecnologías, monitorear cambios, optimizar o administrar, según sea el objetivo del análisis, con base en los resultados. (Giraud, 2015).

Un problema de los datos de alta dimensión son los cálculos numéricos, los cuales pueden volverse computacionalmente intensos y exceder en gran medida los recursos de cómputo disponibles (Giraud, 2015). Por ejemplo, en operaciones básicas con matrices $p \times p$ se requiere de al menos p^{α} operaciones con $\alpha > 2$. Cuando p incrementa en miles, las iteraciones de las operaciones pueden volverse intensivas.

En muchos casos, estamos interesados en determinar la influencia de un conjunto de covariables (por ejemplo, edad, sexo, datos médicos, expresiones genéticas) en una cantidad particular de interés llamada variable de respuesta (Gueuning, 2017). Con el objetivo de establecer matemáticamente la relación de interés, se pueden utilizar modelos de regresión lineal, modelos de índice único y modelos mixtos (Fan y Li, 2006). Sin embargo, la mayoría de los procedimientos estadísticos clásicos se han desarrollado en un marco de baja dimensión, como el método de Mínimos Cuadrados Ordinarios (MCO) que se utiliza para conjuntos de datos que contienen un pequeño número de covariables.

El desarrollo de la inferencia estadística en datos de alta dimensión se ha ido logrando

para modelos lineales y lineales generalizados, principalmente basados en técnicas comunes como la reducción directa de la dimensionalidad, que facilita la visualización de los datos. Métodos como el Análisis de Componentes Principales (ACP), introducido por Pearson (1901) (Franke *et al.*, 2016), es un claro ejemplo de reducción de dimensionalidad; técnicas de selección de variables como el criterio de información de Akaike (AIC, Akaike Information Criterion en inglés) propuesto por Akaike (1973), el criterio de información bayesiano (BIC, Bayesian Information Criterion en inglés) desarrollado por Schwarz (1978), son formas de resolver el problema, pero estos implican un inconveniente de optimización con un tiempo de cálculo que aumenta exponencialmente con la dimensionalidad (Fan y Li, 2006).

Uno de los métodos de análisis de regresión que realiza selección de variable y regularización para mejorar la exactitud e interpretabilidad del modelo estadístico es la regresión LASSO (Least Absolute Shrinkage and Selection Operator, por sus siglas en inglés), introducida por Tibshirani (1996), la cual se ha vuelto muy popular para problemas de estimación de alta dimensión por su viabilidad computacional (Bühlmann y Van De Geer, 2011) y es utilizado principalmente en aprendizaje automático.

El modelo lineal mixto provee una herramienta flexible para analizar datos agrupados y relacionados, el cual incluyen datos de medidas repetidas, datos longitudinales y datos multinivel (Li *et al.*, 2021). Este modelo incorpora efectos fijos y efecto aleatorios, donde los efectos aleatorios inducen correlaciones entre las observaciones dentro de cada grupo y se adaptan a la estructura del mismo. En muchos estudios genómicos y económicos, la dimensión del espacio puede ser grande y posiblemente mucho mayor que el tamaño de la muestra. Se han propuesto y estudiado una variedad de modelos y enfoques estadísticos para analizar datos de alta dimensión. Sin embargo, la mayoría de ellos se limitan a tratar con observaciones independientes, como modelos lineales y modelos lineales generalizados (Li *et al.*, 2021). Es por ello que el desarrollo y la comprensión de procedimientos estadísticos adecuados para datos de alta dimensión es un desafío importante y reciente para los investigadores (Gueuning, 2017).

1.1 Objetivos

- Presentar y desarrollar métodos estadísticos como una solución del problema de la alta dimensionalidad en el modelo lineal mixto.
- Evaluar la eficiencia de los recursos computacionales de los métodos propuestos en datos reales de alta dimensión.

1.2 Planteamiento del problema

En diversos problemas de análisis estadístico están presente los datos de alta dimensión, principalmente en cuanto a los modelos lineales. Lo que conlleva a un problema conocido como "la maldición de la dimensionalidad" (Bellman, 1961), es decir cuando el número de predictores p es más grande al número de observaciones n, ($p \gg n$) lo que plantea varios desafíos estadísticos y computacionales.

1.3 Organización del trabajo

En el capítulo 2 se realiza una revisión de literatura sobre el modelo de regresión lineal y el modelo lineal mixto desde el punto de vista clásico como bayesiano. También se presentan revisan los principales métodos para el ajuste de los modelos mediante los enfoques clásico y bayesiano.

En el capítulo 3 se presenta el primer método propuesto para el ajuste del modelo lineal mixto mediante el uso de diferentes matrices de varianzas-convarianzas; el desarrollo del paquete `lme4GS`, que está enfocado en ajustar el modelo lineal mixto con estructuras de covarianza definidas por el usuario, y se presentan ejemplos de modelos de selección genómica con datos de alta dimensión.

En el capítulo 4 se presenta el segundo método propuesto que está basado en la aumentación ortogonal de datos para el ajuste del modelo lineal mixto utilizando el algoritmo Esperanza-Maximización.

En el capítulo 5 se presentan las conclusiones generales obtenidas del desarrollo y análisis de los métodos propuestos. Finalmente se presentan como anexos los programas en R (R Core Team, 2021) y C (Kernighan y Ritchie, 1988), empleados para realizar los ajustes de los modelos.

CAPÍTULO 2. REVISIÓN DE LITERATURA

2.1 Modelo de Regresión Lineal (MRL)

El análisis de regresión lineal es una técnica estadística para estimar la relación entre variables que tiene una relación causa-efecto (Uyanik y Güler, 2013). El objetivo de la regresión lineal es analizar la relación entre una variable dependiente y una o más variables independientes formulando una ecuación lineal entre la variable dependiente y las variables independientes.

Dado un conjunto de n pares de observaciones independientes $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_n, y_n)$ donde y_i es la respuesta del i-ésimo sujeto o caso de estudio que dependen de ciertas características $\mathbf{x}_i$, el modelo de regresión lineal se define por la ecuación (2.1):

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + e_i. \tag{2.1}$$

La ecuación anterior se puede escribir como:

$$y_i = \beta_0 + \sum_{i=1}^{p} x_{ij}\beta_j + e_i, \tag{2.2}$$

donde y_i es la variable respuesta o dependiente, x_{ij} son las covariables o variables independientes, β_0 es el intercepto, β_j es el coeficiente de regresión asociado a las covariables, $j = 1, \ldots, p$ y e_i es el término de error, $i = 1, \ldots, n,\ j = 1, \ldots, p$. El error sigue una distribución de probabilidad $\mathcal{P}$ con media cero, $E\left(e_i|\mathbf{x}_i\right) = 0$. Típicamente e_i tiene una distribución independiente e idénticamente distribuida normal (NIID) con media cero y varianza constante, es decir,

$$e_i \sim NIID\left(0, \sigma_e^2\right).$$

En forma matricial, el modelo (2.2) se expresa de la siguiente manera:

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{e},$$

donde $\boldsymbol{y}$ es el vector de respuesta de dimensión n, $\boldsymbol{X}$ es la matriz de covariables de dimensión $n \times (p+1)$, $\boldsymbol{\beta}$ es el vector de parámetros de dimensión $(p+1)$, $\boldsymbol{e}$ es el vector de error de dimensión n el cual sigue una distribución normal multivariada (NM) con vector de medias $\mathbf{0}$ y matriz de varianzas-covarianzas $\sigma_e^2\boldsymbol{I}$, es decir, $\boldsymbol{e} \sim NM\left(\mathbf{0}, \sigma_e^2\mathbf{I}\right)$.

La estimación del vector de parámetros $\boldsymbol{\beta}$ se puede realizar mediante el método de Mínimos Cuadrados Ordinarios (MCO) o por Máxima Verosimilitud (MV). El método más común para la estimación es el MCO el cual minimiza la suma de cuadrados del error (SCE). Entonces,

$$\begin{aligned} \text{SCE}\left(\boldsymbol{\beta}\right) &= \boldsymbol{e}^t\boldsymbol{e} \\ &= \left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\right)^t\left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\right) \\ &= \boldsymbol{y}^t\boldsymbol{y} - 2\boldsymbol{\beta}^t\boldsymbol{X}^t\boldsymbol{y} + \boldsymbol{\beta}^t\boldsymbol{X}^t\boldsymbol{X}\boldsymbol{\beta}. \end{aligned} \tag{2.3}$$

Calculando la derivada de la ecuación (2.3) con respecto a $\boldsymbol{\beta}$, e igualando a cero se obtienen las ecuaciones normales:

$$\boldsymbol{X}^t\boldsymbol{X}\boldsymbol{\beta} = \boldsymbol{X}^t\boldsymbol{y},$$

si el número de observaciones n es mayor al número de covariables p, es decir $n > p$ y si $\boldsymbol{X}$ de rango completo, el mejor estimador lineal insesgado (BLUE, Best Linear Unbiased Estimator en inglés) está dado por (2.4):

$$\widehat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^t\boldsymbol{X}\right)^{-1}\boldsymbol{X}^t\boldsymbol{y}. \tag{2.4}$$

Cuando hay presencia de multicolinealidad, aunque los estimadores de mínimos cuadrados son insesgados, sus varianzas son grandes ya que $(\boldsymbol{X}^t\boldsymbol{X})^{-1}$ es casi singular. Por otro lado, con datos de alta dimensión con $p \gg n$, la estimación de $\boldsymbol{\beta}$ por el método de mínimos cuadrados es problemática ya que se trata de un problema mal condicionado y esto a su vez implica que $\widehat{\boldsymbol{\beta}}$ no es único (Pérez-Elizalde *et al.*, 2022).

El Estimador de Máxima Verosimilitud (EMV) se define como aquel valor del parámetro que maximiza la función de verosimilitud del modelo de regresión lineal. La variable respuesta y_i es modelada mediante una distribución normal,

$$y_i \sim NIID\left(x_i^t\boldsymbol{\beta}, \sigma_e^2\right).$$

Por lo que la función de verosimilitud es:

$$\begin{aligned} L\left(\boldsymbol{\beta}, \sigma_e^2 \mid \boldsymbol{y}\right) &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma_e^2}} \exp\left\{-\frac{\left(y_i - \boldsymbol{x}_i^t\boldsymbol{\beta}\right)^2}{2\sigma_e^2}\right\} \\ &= \left(\frac{1}{\sqrt{2\pi\sigma_e^2}}\right)^n \exp\left\{-\sum_{i=1}^{n} \frac{\left(y_i - \boldsymbol{x}_i^t\boldsymbol{\beta}\right)^2}{2\sigma_e^2}\right\} \\ &= \left(\frac{1}{\sqrt{2\pi\sigma_e^2}}\right)^n \exp\left\{-\frac{\left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\right)^t\left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\right)}{2\sigma_e^2}\right\}. \end{aligned} \tag{2.5}$$

Tomando el logaritmo de la función de verosimilitud (2.5) tenemos,

$$\log\left(L\left(\boldsymbol{\beta}, \sigma_e^2 \mid \boldsymbol{y}\right)\right) = -\frac{n}{2}\log\left(2\pi\right) - \frac{n}{2}\log\left(\sigma_e^2\right) - \frac{\left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\right)^t\left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\right)}{2\sigma_e^2}. \tag{2.6}$$

Obteniendo la derivada de 2.6,

$$\begin{aligned}\frac{\partial \log\left(L\left(\boldsymbol{\beta}, \sigma_e^2 \mid \boldsymbol{y}\right)\right)}{\partial \boldsymbol{\beta}} &= \frac{1}{2\sigma_e^2}\frac{\partial\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}\right)^t\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}\right)}{\partial \boldsymbol{\beta}} \\ &= \frac{1}{2\sigma_e^2}\frac{\partial\left(\boldsymbol{y}^2-2\boldsymbol{X}^t\boldsymbol{\beta}\boldsymbol{y}+\boldsymbol{X}\boldsymbol{X}^t\boldsymbol{\beta}^2\right)}{\partial \boldsymbol{\beta}} \\ &= \frac{1}{2\sigma_e^2}\left(-2\boldsymbol{X}^t\boldsymbol{y}+2\boldsymbol{X}^t\boldsymbol{X}\boldsymbol{\beta}\right).\end{aligned} \tag{2.7}$$

Igualando a cero la derivada (2.7),

$$\frac{1}{2\sigma_e^2}\left(-2\boldsymbol{X}^t\boldsymbol{y}+2\boldsymbol{X}^t\boldsymbol{X}\boldsymbol{\beta}\right)=\boldsymbol{0}. \tag{2.8}$$

multiplicando ambos lados de la ecuación por $2\sigma_e^2$ y re-acomodando términos se obtiene:

$$\boldsymbol{X}^t\boldsymbol{X}\boldsymbol{\beta}=\boldsymbol{X}^t\boldsymbol{y},$$

el cual corresponde a las ecuaciones normales cuya solución fue discutida previamente. Si $n > p$ y $\boldsymbol{X}$ es de rango completo por columnas, el estimador de máxima verosimilitud para $\boldsymbol{\beta}$ está dado por:

$$\widehat{\boldsymbol{\beta}}=\left(\boldsymbol{X}^t\boldsymbol{X}\right)^{-1}\boldsymbol{X}^t\boldsymbol{y},$$

el cual coincide con el estimador de MCO (2.4). Una vez que obtenido $\widehat{\boldsymbol{\beta}}$ es posible obtener los valores predichos usando la expresión $\widehat{\boldsymbol{y}}=\boldsymbol{X}\widehat{\boldsymbol{\beta}}$.

2.2 Modelo Lineal Mixto (MLM)

El modelo lineal mixto provee una herramienta poderosa y flexible para el análisis de una gran variedad de datos incluyendo datos longitudinales, medidas repetidas, multinivel, espacial y geoestadísticos, y bionformáticos (Gumedze y Dunne, 2011). El MLM es una extensión de los modelos lineales ya que permiten incluir efectos fijos y aleatorios en el

modelo. Se utilizan particularmente cuando las observaciones están correlacionadas. El modelo lineal mixto está definido por la ecuación (2.9):

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{u} + \boldsymbol{e}, \tag{2.9}$$

donde $\boldsymbol{y}$ es el vector de respuesta de dimension n, $\boldsymbol{\beta}$ es el vector de efectos fijos de dimension p con matriz diseño $\boldsymbol{X}$ de dimensión $n \times p$, $\boldsymbol{u}$ es el vector de efectos aleatorios de dimension q con matriz diseño $\boldsymbol{Z}$ de dimension $n \times q$, y $\boldsymbol{e}$ es el vector de errores de dimensión n.

Se supone que la esperanza y la matriz de varianzas-covarianzas de los componentes aleatorios $\boldsymbol{e}$ y $\boldsymbol{u}$ son:

$$E\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{e}\end{bmatrix} = \begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{0}\end{bmatrix},$$

y

$$\text{Var}\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{e}\end{bmatrix} = \sigma^2\begin{bmatrix}\boldsymbol{G} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{R}\end{bmatrix},$$

, $\boldsymbol{u}$ y $\boldsymbol{e}$ son independientes.

donde $\boldsymbol{G}$ y $\boldsymbol{R}$ son matrices no singulares. Además, se asume que $\boldsymbol{u}$ y $\boldsymbol{e}$ siguen una distribución normal multivariada, esto es:

$$\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{e}\end{bmatrix} \sim NM\left(\begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{0}\end{bmatrix},\ \sigma^2\begin{bmatrix}\boldsymbol{G} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{R}\end{bmatrix}\right).$$

De acuerdo a Patterson y Thompson (1971), la matriz de varianzas-covarianzas de $\boldsymbol{y}$ es,

$$\text{var}\left(\boldsymbol{y}\right) = \sigma^2\left(\boldsymbol{Z}\boldsymbol{G}\boldsymbol{Z}^t + \boldsymbol{R}\right) = \sigma^2\boldsymbol{V},$$

donde

$$\boldsymbol{V} = \boldsymbol{Z}\boldsymbol{G}\boldsymbol{Z}^t + \boldsymbol{R}. \tag{2.10}$$

Se prefiere que $\boldsymbol{G}$ sea de dimensión pequeña para que sea posible obtener su inversa de manera rápida. La forma más simple de $\boldsymbol{R}$ es $\sigma^2\boldsymbol{I}$, y si $\mathbf{u}$ es el vector nulo, el modelo mixto se reduce al modelo lineal estándar (Wolfinger *et al.*, 1991). Se pueden especificar las estructuras de $\boldsymbol{G}$ y $\boldsymbol{R}$ asignando un tipo para cada efecto aleatorio. Los posibles tipos incluyen:

- Simple,
- No estructurado,
- Series de tiempo,
- Distancia,

o una combinación diagonal de bloques de cualquiera de estos (Wolfinger *et al.*, 1991).

Una vez que el modelo ha sido formulado, se necesitan métodos de estimación para los parámetros del modelo. Existen diversos métodos para obtener de manera simultánea la estimación de los efectos fijos ($\boldsymbol{\beta}$) y la predicción de los efectos aleatorios ($\boldsymbol{u}$). Estos métodos incluyen las ecuaciones de los modelos mixtos de Henderson (1950), técnicas basadas en regresión en dos etapas, y estimación bayesiana (Gumedze y Dunne, 2011).

2.2.1 Estimación mediante las ecuaciones del modelo mixto

Henderson (1950) asume que $\boldsymbol{u}$ y $\boldsymbol{y}$ se distribuyen conjuntamente como una variable aleatoria con distribución normal multivariada,

$$\begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{y} \end{bmatrix} \sim NM\left(\begin{bmatrix} \mathbf{0} \\ \boldsymbol{X}\boldsymbol{\beta} \end{bmatrix}, \sigma^2 \begin{bmatrix} \boldsymbol{G} & \boldsymbol{G}\boldsymbol{Z}^t \\ \boldsymbol{Z}\boldsymbol{G} & \boldsymbol{V} \end{bmatrix} \right).$$

Por lo tanto, la función de densidad marginal de $\boldsymbol{y}$ es:

$$\boldsymbol{y} \sim NM\left(\boldsymbol{X\beta}, \sigma^2\boldsymbol{V}\right), \tag{2.11}$$

donde $\boldsymbol{V}$ está definido en (2.10), con $\boldsymbol{G}$ y $\boldsymbol{R}$ conocidos. Henderson (1950) maximiza el logaritmo de la distribución conjunta de $(\boldsymbol{y}, \boldsymbol{u})$ para obtener los estimadores de $\boldsymbol{\beta}$ y $\boldsymbol{u}$. Entonces la distribución marginal de $\boldsymbol{u}$ es:

$$\boldsymbol{u} \sim NM\left(\boldsymbol{0}, \sigma^2\boldsymbol{G}\right),$$

y la distribución condicional de $\boldsymbol{y}$ dado $\boldsymbol{u}$ es:

$$\boldsymbol{y} \mid \boldsymbol{u} \sim NM\left(\boldsymbol{X\beta} + \boldsymbol{Zu}, \sigma^2\boldsymbol{R}\right).$$

El logaritmo de la distribución conjunta de $(\boldsymbol{y}, \boldsymbol{u})$ está dado por (Gumedze y Dunne, 2011),

$$\begin{aligned}
\log f(\boldsymbol{y}, \boldsymbol{u}) &= \log f(\boldsymbol{y} \mid \boldsymbol{u}) + \log f(\boldsymbol{u}) \\
&= -\frac{1}{2}\left\{n\log\sigma^2 + \log\boldsymbol{R} + \frac{(\boldsymbol{y}-\boldsymbol{X\beta}-\boldsymbol{Zu})^t\boldsymbol{R}^{-1}(\boldsymbol{y}-\boldsymbol{X\beta}-\boldsymbol{Zu})}{\sigma^2}\right\} \\
&- \frac{1}{2}\left\{q\log\sigma^2 + \log\boldsymbol{G} + \frac{\boldsymbol{u}^t\boldsymbol{G}^{-1}\boldsymbol{u}}{\sigma^2}\right\} \\
&= -\frac{1}{2}\left\{(n+q)\log\sigma^2 + \log\boldsymbol{R} + \log\boldsymbol{G} + \frac{(\boldsymbol{y}-\boldsymbol{X\beta})^t\boldsymbol{R}^{-1}(\boldsymbol{y}-\boldsymbol{X\beta})}{\sigma^2}\right\} \\
&- \frac{1}{2\sigma^2}\left\{\boldsymbol{u}^t\left(\boldsymbol{ZR}^{-1}\boldsymbol{Z}^t + \boldsymbol{G}^{-1}\right)\boldsymbol{u} - 2(\boldsymbol{y}-\boldsymbol{X\beta})^t\boldsymbol{R}^{-1}\boldsymbol{Zu}\right\}.
\end{aligned}$$

Las estimaciones de $\boldsymbol{\beta}$ y $\boldsymbol{u}$ se obtienen resolviendo el siguiente par de ecuaciones:

$$\boldsymbol{X}^t\boldsymbol{R}^{-1}\left(\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}\right) - \boldsymbol{X}^t\boldsymbol{R}^{-1}\boldsymbol{Z}\tilde{\boldsymbol{u}} = \boldsymbol{0}, \tag{2.12}$$

$$\boldsymbol{Z}^t\boldsymbol{R}^{-1}\left(\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}\right) - \left(\boldsymbol{Z}^t\boldsymbol{R}^{-1}\boldsymbol{Z} + \boldsymbol{G}^{-1}\right)\tilde{\boldsymbol{u}} = \boldsymbol{0}. \tag{2.13}$$

Las ecuaciones (2.12) y (2.13) se conocen como las ecuaciones del modelo mixto (MMEs, Mixed Model Equations en inglés) propuestas por Henderson (1950). Estas ecuaciones se pueden escribir en forma matricial como se presenta a continuación:

$$\begin{bmatrix} \boldsymbol{X}^t\boldsymbol{R}^{-1}\boldsymbol{X} & \boldsymbol{X}^t\boldsymbol{R}^{-1}\boldsymbol{Z} \\ \boldsymbol{Z}^t\boldsymbol{R}^{-1}\boldsymbol{X} & \boldsymbol{Z}^t\boldsymbol{R}^{-1}\boldsymbol{Z}+\boldsymbol{G}^{-1} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \tilde{\boldsymbol{u}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{X}^t\boldsymbol{R}^{-1}\boldsymbol{y} \\ \boldsymbol{Z}^t\boldsymbol{R}^{-1}\boldsymbol{y} \end{bmatrix}$$

Las soluciones para $\boldsymbol{\beta}$ y $\boldsymbol{u}$ con $\boldsymbol{G}$ y $\boldsymbol{R}$ son conocidas, están dadas por (2.14) y (2.15) (para más detalles véase Gumedze y Dunne (2011)):

$$\widehat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^t\boldsymbol{V}^{-1}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^t\boldsymbol{V}^{-1}\boldsymbol{y}, \tag{2.14}$$

$$\tilde{\boldsymbol{u}} = \boldsymbol{G}\boldsymbol{Z}^t\boldsymbol{V}^{-1}\left(\boldsymbol{y}-\boldsymbol{X}\widehat{\boldsymbol{\beta}}\right), \tag{2.15}$$

con matrices de varianzas-covarianzas dadas por (2.16) y (2.17):

$$\text{var}\left(\widehat{\boldsymbol{\beta}}\right) = \sigma^2\left(\boldsymbol{X}^t\boldsymbol{V}^{-1}\boldsymbol{X}\right)^{-1} \tag{2.16}$$

$$\text{var}\left(\tilde{\boldsymbol{u}}\right) = \sigma^2\boldsymbol{G}\boldsymbol{Z}^t\boldsymbol{P}\boldsymbol{Z}\boldsymbol{G}, \tag{2.17}$$

donde

$$\boldsymbol{P} = \boldsymbol{V}^{-1} - \boldsymbol{V}^{-1}\boldsymbol{X}\left(\boldsymbol{X}^t\boldsymbol{V}^{-1}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^t\boldsymbol{V}^{-1}. \tag{2.18}$$

El estimador $\widehat{\boldsymbol{\beta}}$ dador por (2.14) se conoce como el estimador BLUE de $\boldsymbol{\beta}$. Si $\boldsymbol{X}$ no es de rango completo entonces se utiliza la inversa generazalida, $\left(\boldsymbol{X}^t\boldsymbol{V}^{-1}\boldsymbol{X}\right)^{-}$, para obtener la solución de $\boldsymbol{\beta}$. El predictor $\tilde{\boldsymbol{u}}$ es conocido como el mejor predictor lineal insesgado (BLUP, Best Linear Unbiased Predictor en inglés).

Para obtener los resultados (2.14) y (2.15) se supone que los parámetros de varianza son conocidos pero generalmente son desconocidos por lo que se tiene que estimar.

Existen diversos métodos para la estimación de los parámetros de varianza. Estos métodos incluyen el método ANOVA para datos balanceados, el método de estimación cuadrática de norma mínima (MINQUE, por sus siglas en inglés) para datos no balanceados propuesto por Rao (1971), los métodos de Henderson I, II y III (Gumedze y Dunne, 2011) propuesto por Henderson (1953), el método de Lee y Nelder (1998) el cual utiliza una cuasi-verosimilitud extendida, entre otros.

El método de máxima verosimilitud (ML, Maximum Likelihood en inglés) y el de máxima verosimilitud restringida o residual (REML, Restricted Maximum Likelihood en inglés) son métodos estándar en la estimación de los parámetros de varianza para datos balanceados y no balanceados (Gumedze y Dunne, 2011). Los estimadores obtenidos por los métodos de ML y REML tienen propiedades asintóticas que los hacen preferibles sobre los estimadores obtenidos con otros métodos (Caballero *et al.*, 2003).

2.2.2 Estimación por máxima verosimilitud

La distribución marginal de $\boldsymbol{y}$ esta dado por (2.11), por lo que la función log-verosimilitud marginal de $\boldsymbol{y}$ esta dado por (2.19):

$$\ell\left(\boldsymbol{\beta}, \sigma^2; \boldsymbol{y}\right) = -\frac{1}{2}\left\{n\log\left(2\pi\right) + n\log\sigma^2 + \log\mid \boldsymbol{V} \mid + \frac{\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}\right)^t \boldsymbol{V}^{-1}\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}\right)}{\sigma^2}\right\}. \tag{2.19}$$

Suponiendo que todos los componentes de varianza en $\boldsymbol{V}$ son conocidos, entonces realizando las derivaciones correspondientes en (2.19), igualando a cero y resolviendo las ecuaciones resultantes, se obtienen los estimadores de máxima verosimilitud (MLE, Maximum Likelihood Estimator en inglés) (veáse Gumedze y Dunne, 2011),

$$\widehat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^t\boldsymbol{V}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^t\boldsymbol{V}^{-1}\boldsymbol{y}, \tag{2.20}$$

$$\widehat{\sigma}^2 = \frac{1}{n}\left(\boldsymbol{y}-\boldsymbol{X}\widehat{\boldsymbol{\beta}}\right)^t\boldsymbol{V}^{-1}\left(\boldsymbol{y}-\boldsymbol{X}\widehat{\boldsymbol{\beta}}\right). \tag{2.21}$$

2.2.3 Estimación por REML

Para tamaños de muestra "pequeño" los estimadores obtenidos por máxima verosimilitud son generalmente sesgados, puesto que al estimar los efectos aleatorios no se toma en cuenta la pérdida en grados de libertad que resulta de la estimación de los efectos fijos (Caballero *et al.*, 2003). Para solucionar este problema, el método REML utiliza combinaciones lineales de los elementos del vector de datos $\boldsymbol{y}$ de tal manera que esas combinaciones no contienen efectos fijos (Caballero *et al.*, 2003). Esas combinaciones generan una función de verosimilitud que no depende de los efectos fijos (Searle *et al.*, 1992). El método REML maximiza la función de verosimilitud restringida con respecto a cada uno de los componentes de varianza.

Sea $\boldsymbol{y} \sim NM\left(\boldsymbol{X\beta}, \sigma^2\boldsymbol{V}\right)$, y $\boldsymbol{K}^t\boldsymbol{X} = \boldsymbol{0}$, utilizando las propiedades de la distribución normal multivariada se puede mostrar que:

$$\boldsymbol{K}^t\boldsymbol{y} \sim NM\left(\boldsymbol{0}, \sigma^2\boldsymbol{K}^t\boldsymbol{V}\boldsymbol{K}\right).$$

La función log-verosimilitud residual es:

$$\ell\left(\sigma^2; \boldsymbol{K}^t\boldsymbol{y}\right) = -\frac{1}{2}\left\{(n-p)\log\left(2\pi\right) + (n-p)\log\sigma^2 + \log\mid \boldsymbol{K}^t\boldsymbol{V}^{-1}\boldsymbol{K} \mid\right\} + \left\{\frac{1}{\sigma^2}\boldsymbol{y}\boldsymbol{K}\left(\boldsymbol{K}^t\boldsymbol{V}^{-1}\boldsymbol{K}\right)^{-1}\boldsymbol{K}^t\boldsymbol{y}\right\}.$$

Patterson y Thompson (1971) derivaron la distribución de probabilidad de $\boldsymbol{K}^t\boldsymbol{y}$ eligiendo cuidadosamente $\boldsymbol{K}^t$ como una matriz de dimensión $(n-p)\times n$, cuyas hileras son linealmente independiente de $\boldsymbol{I} - \boldsymbol{X}\left(\boldsymbol{X}^t\boldsymbol{X}\right)^{-1}\boldsymbol{X}^t$. Ya que la matriz $\boldsymbol{K}^t$ es simétrica e idempotente y con rango $n-p$, entonces se puede expresar $\boldsymbol{K}\boldsymbol{K}^t$ tal que $\boldsymbol{K}\boldsymbol{K}^t = \boldsymbol{I}$ (Gumedze y Dunne, 2011).

Patterson y Thompson (1971) argumentaron que si $E\left(\boldsymbol{K}^t\boldsymbol{y}\right) = \boldsymbol{0}$ entonces $\boldsymbol{K}^t\boldsymbol{y}$ se encuentra en el espacio de error y por lo tanto no contiene información sobre los efectos fijos, pero contiene información sobre los parámetros de varianza (Gumedze y Dunne, 2011). Entonces la nueva función log-verosimilitud residual es:

$$\ell\left(\sigma^2;\boldsymbol{y}\right) = -\frac{1}{2}\left\{(n-p)\log\sigma^2 + \log \mid \boldsymbol{V} \mid + \log \mid \boldsymbol{X}^t\boldsymbol{V}^{-1}\boldsymbol{X} \mid + \frac{\boldsymbol{y}^t\boldsymbol{P}\boldsymbol{y}}{\sigma^2}\right\}, \tag{2.22}$$

donde $\boldsymbol{P}$ está definido en (2.18). Diferenciando la función (2.22) con respecto a σ^2, se obtiene:

$$\frac{\partial\ell\left(\sigma^2;\boldsymbol{y}\right)}{\partial\sigma^2} = -\frac{n-p}{2\sigma^2} + \frac{\boldsymbol{y}^t\boldsymbol{P}\boldsymbol{y}}{2\sigma^4}. \tag{2.23}$$

Igualando a cero y resolviendo (2.23) se obtiene el estimador REML de la varianza del error, como

$$\widehat{\sigma}^2 = \frac{\boldsymbol{y}^t\boldsymbol{P}\boldsymbol{y}}{n-p}. \tag{2.24}$$

Generalmente se presentan más de un componente de varianza en el modelo lineal mixto y no se puede obtener analíticamente los estimadores por lo que es necesario obtenerlos mediante métodos iterativos como los algoritmos Newton-Raphson (NR), puntuaciones de Fisher (Fisher scoring), e información promedio (Average Information).

2.3 Inferencia bayesiana

El objetivo de la inferencia estadística es obtener conclusiones sobre una determinada cantidad de interés desconocida, $\boldsymbol{\theta}$, mediante el uso de modelos probabilísticos. Bajo la perspectiva bayesiana se consideran los datos observados e información previa existente, mientras que bajo el punto de vista clásico solamente se considera la información de los datos y cualquier otra información adicional no se incorpora en el proceso de decisión (Gelman *et al.*, 2004). La inclusión de la información previa en el proceso de inferencia es el punto principal del análisis bayesiano, esto se hace incluyendo una distribución *a priori*, el cual describe todo el conocimiento disponible que se puede tener sobre la cantidad de interés, y su elección puede influir en los resultados finales dependiendo de la cantidad de datos disponibles (Gelman *et al.*, 2004).

Sea $\boldsymbol{\theta}$ el vector de parámetros de interés y $\boldsymbol{y}$ el vector de datos observados entonces, para

hacer inferencia sobre $\boldsymbol{\theta}$ dado $\boldsymbol{y}$ se debe comenzar con un modelo que proporcione una distribución de probabilidad conjunta para $\boldsymbol{\theta}$ y $\boldsymbol{y}$. La función de probabilidad se puede escribir como el producto de dos densidades conocidas, la densidad *a priori* $p(\boldsymbol{\theta})$ y la distribución muestral $p(\boldsymbol{y}|\boldsymbol{\theta})$, tal y como indica Gelman *et al.* (2004), esto es:

$$p(\boldsymbol{\theta}, \boldsymbol{y}) = p(\boldsymbol{\theta})\, p(\boldsymbol{y}|\boldsymbol{\theta}).$$

Note que la distribución muestral $p(\boldsymbol{\theta})$ es una función de $\boldsymbol{\theta}$ y corresponde a la función de verosimilitud de $\boldsymbol{\theta}$, es decir $l(\boldsymbol{\theta}) = p(\boldsymbol{y}|\boldsymbol{\theta})$. La densidad *a priori* contiene toda la distribución de probabilidad de $\boldsymbol{\theta}$ antes de observar los valores de $\boldsymbol{y}$. Existen muchos tipos de distribuciones *a priori*, como no-informativas, conjugadas e impropias (Robert *et al.*, 2007, Cap. 1). La inferencia debe basarse en la distribución de probabilidad de $\boldsymbol{\theta}$ después de observar los valores de $\boldsymbol{y}$, esta distribución se llama distribución *a posteriori* la cual se obtiene mediante el teorema de Bayes definido por la ecuación (2.25),

$$p(\boldsymbol{\theta}|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{\theta})\, p(\boldsymbol{\theta})}{p(\boldsymbol{y})}, \tag{2.25}$$

donde $p(\boldsymbol{y})$ es la constante de normalización y se le conoce como la distribución marginal de los datos o la distribución predictiva *a priori* (Chen *et al.*, 2012), y se obtiene como sigue:

$$p(\boldsymbol{y}) = \sum_{\boldsymbol{\theta}} p(\boldsymbol{y}|\boldsymbol{\theta})\, p(\boldsymbol{\theta}), \tag{2.26}$$

si $\boldsymbol{\theta}$ es discreto y

$$p(\boldsymbol{y}) = \int p(\boldsymbol{y}|\boldsymbol{\theta})\, p(\boldsymbol{\theta})\, \partial\boldsymbol{\theta}, \tag{2.27}$$

si $\boldsymbol{\theta}$ es continuo. Una forma equivalente de obtener la distribución *a posteriori* (2.25) es omitir el factor $p(\boldsymbol{y})$ el cual no depende de $\boldsymbol{\theta}$ y, con $\boldsymbol{y}$ fijo, se puede considerar como constante,

$$p\left(\boldsymbol{\theta}|\boldsymbol{y}\right) \propto p\left(\boldsymbol{\theta}\right) p\left(\boldsymbol{y}|\boldsymbol{\theta}\right).$$

2.3.1 Predicción

Otro elemento importante de la inferencia bayesiana es la distribución predictiva o marginal de $\boldsymbol{y}$ con densidad $p\left(\boldsymbol{y}\right)$ dado por (2.27) y da la distribución de la esperanza para las observaciones $\boldsymbol{y}$, $p\left(\boldsymbol{y}\right) = E\left\{p\left(\boldsymbol{y}|\boldsymbol{\theta}\right)\right\}$, y la esperanza es tomada con respecto a la distribución *a priori* de $\boldsymbol{\theta}$ (Gamerman y Lopes, 2006).

De igual manera se puede realizar inferencia sobre una observación futura y_{nuevo} después de observar $\boldsymbol{y}$. Esta predicción debe basarse en la distribución de $y_{nuevo}|\boldsymbol{y}$. Si y_{nuevo} y $\boldsymbol{y}$ son condicionalmente independientes dado $\boldsymbol{\theta}$ entonces la distribución (2.28) se conoce como la distribución predictiva *a posteriori*:

$$\begin{aligned} p\left(\boldsymbol{y}_{nuevo}|\boldsymbol{y}\right) &= \int p\left(\boldsymbol{y}_{nuevo}, \boldsymbol{\theta} \mid \ \boldsymbol{y}\right) \partial\boldsymbol{\theta} \\ &= \int p\left(\boldsymbol{y}_{nuevo} \mid \boldsymbol{\theta}, \boldsymbol{y}\right) p\left(\boldsymbol{\theta} \mid \boldsymbol{y}\right) \partial\boldsymbol{\theta} \\ &= \int p\left(\boldsymbol{y}_{nuevo}|\boldsymbol{\theta}\right) p\left(\boldsymbol{\theta}|\boldsymbol{y}\right) \partial\boldsymbol{\theta}. \end{aligned} \tag{2.28}$$

La independencia condicional entre el vector de nuevas observaciones $\boldsymbol{y}_{nuevo}$ y $\boldsymbol{y}$ se obtiene si, $\boldsymbol{y} = \left(y_1, ..., y_n\right)^t$ y $\boldsymbol{y}_{nuevo} = \left(y_{n+1}, ..., y_{n+m}\right)^t$ son muestras de $p\left(\boldsymbol{y}|\boldsymbol{\theta}\right)$, entonces la distribución predictiva se usa para predecir valores futuros de una población.

2.3.2 Resumen de la información

Una vez que se tiene la distribución *a posteriori* se puede resumir la información de $\boldsymbol{\theta}$ a través de algunos elementos importantes como medidas de localización y de dispersión. Estas medidas se obtienen para tener una idea de los posibles valores centrales y variabilidad asociados con la distribución *a posteriori* (Gamerman y Lopes, 2006).

Las principales medidas de localidad son la media, moda y mediana, y las de dispersión

son la varianza, desviación estándar, y rango intercuartil. Todas estas medidas pueden evaluarse para las distribuciones conjunta, marginal y condicional (Gamerman y Lopes, 2006).

La mayoría de los procedimientos de resumen involucran integración de la distribución *a posteriori*. La inferencia exacta solo será posible si estas integrales pueden realizarse analíticamente, de otro modo se deben realizar aproximaciones. Uno de los métodos de aproximación más importante y utilizados son los métodos cadenas de Markov Monte Carlo (MCMC, Markov Chain Monte Carlo en inglés) (véase Cowles y Carlin, 1996; Gelfand y Smith, 1990; Geyer, 1992; Gilks *et al.*, 1995). El MCMC genera muestras de la distribución *a posteriori* simulada mediante una cadena de Markov.

2.4 Modelo de Regresión Lineal bayesiano

El modelo de regresión lineal bayesiano es un enfoque al modelo de regresión lineal en donde al análisis estadístico se realiza dentro de un contexto de inferencia bayesiana. La diferencia fundamental entre el MCO y la regresión bayesiana es que el último asocia una distribución de probabilidad para los parámetros de regresión $\boldsymbol{\beta}$ (DelSole, 2007). Esta distribución llamada *a priori*, $p(\boldsymbol{\beta})$, cuantifica el "grado de creencia" del valor de $\boldsymbol{\beta}$ antes del análisis de datos (DelSole, 2007).

El objetivo de la regresión lineal bayesiana es determinar la distribución *a posteriori* de los parámetros del modelo dado por (2.1), $\boldsymbol{\beta}$ y σ_e^2. Dada la distribución *a priori*, $p(\boldsymbol{\beta})$, la distribución *a posteriori* se obtiene a partir del teorema de Bayes (2.25):

$$p(\boldsymbol{\beta}|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{\beta})\ \ p(\boldsymbol{\beta})}{\int p(\boldsymbol{y}|\boldsymbol{\beta})\, p(\boldsymbol{\beta})\, \partial\boldsymbol{\beta}}. \tag{2.29}$$

La distribución (2.29) se interpreta como la distribución condicional de los parámetros de regresión dado los datos $\boldsymbol{y}$ (DelSole, 2007).

Una solución más general es suponer que el término de error $\boldsymbol{e}$ se distribuye normal con media cero y matriz de varianzas-covarianzas $\sigma^2\boldsymbol{I}$ (DelSole, 2007),

$$\boldsymbol{e} \sim NM\left(\mathbf{0}, \sigma_e^2 \boldsymbol{I}\right),$$

y el parámetro de regresión $\boldsymbol{\beta}$ tiene una distribución normal *a priori* con media $\boldsymbol{\mu}$ y matriz de covarianza $\gamma^2\boldsymbol{\Sigma}$,

$$\boldsymbol{\beta} \sim NM\left(\boldsymbol{\mu}, \gamma^2\boldsymbol{\Sigma}\right).$$

De acuerdo a Lindley y Smith (1972) esto es un resultado estándar en la teoría bayesiana, dado que todas las distribuciones anteriores son normales, la distribución *a posteriori* también es normal con su media y matriz de covarianza (DelSole, 2007) respectivamente:

$$\widehat{\boldsymbol{\beta}} = E\left(\boldsymbol{\beta} \mid \boldsymbol{y}\right) = \left(\boldsymbol{X}^t\boldsymbol{X} + \frac{\sigma^2}{\gamma^2}\boldsymbol{\Sigma}^{-1}\right)^{-1}\left(\boldsymbol{X}^t\boldsymbol{y} + \frac{\sigma^2}{\gamma^2}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right),$$

$$\Phi = Cov\left[\boldsymbol{\beta} \mid \boldsymbol{y}\right] = \left(\boldsymbol{X}^t\boldsymbol{X} + \frac{\sigma^2}{\gamma^2}\boldsymbol{\Sigma}^{-1}\right)^{-1}\sigma^2.$$

De acuerdo a DelSole (2007) se puede verificar que la media *a posteriori*, $E\left[\boldsymbol{\beta} \mid \boldsymbol{y}\right]$, converge a la solución de MCO cuando $\gamma \to \infty$ lo que demuestra que el MCO es equivalente al estimador bayesiano.

2.5 Método Monte Carlo basado en cadenas de Markov

Los métodos MCMC pertenecen a una clase de algoritmos de muestreo a partir de una distribución de probabilidad. La idea del muestreo por MCMC fue introducido por Metropolis *et al.* (1953), y es utilizado para resolver problemas como de optimización, cálculo de volúmenes complejos, minimización de costos, entre otros. Sin embargo, el gran impulso lo da el enfoque estadístico bayesiano. Una descripción más detallada de los métodos MCMC se presentan en Møller (1993), y en los textos de Chen *et al.* (2012) y Gamerman y Lopes (2006).

Supóngase que se tiene alguna distribución $\pi(x)$, $x \in E \subseteq \mathbb{R}$, la cual es conocida

excepto por una constante multiplicativa. Esta distribución es conocida como distribución objetivo. Si π es suficientemente compleja que no podamos muestrear directamente, un método indirecto para obtener muestras de π es construir una cadena de Markov aperiódico e irreducible con estado-espacio E, y cuya distribución estacionaria es $\pi(x)$ (Brooks y Gelman, 1998). Si implementamos la cadena durante un tiempo suficientemente largo, los valores simulados de la cadena pueden tratarse como una muestra dependiente de la distribución objetivo y usada como base para resumir características importante de π.

Dado las realizaciones $\{X^t : t = 0, 1, ...\}$ de tal cadena, los resultados asintóticos incluyen la convergencia de las realizaciones a la distribución estacionaria, es decir la distribución del estado de la cadena al tiempo t converge a π cuando $t \rightarrow \infty$, y la consistencia del promedio ergódico para cualquier escalar funcional θ (Brooks y Gelman, 1998),

$$\frac{1}{n}\sum_{t=1}^{n}\theta\left(X^t\right) \overset{n\rightarrow\infty}{\rightarrow} \mathbb{E}_\pi\{\theta\left(X\right)\}.$$

En las siguientes secciones se describen los dos algoritmos más utilizados del MCMC: el algoritmo Metropolis-Hasting (Hastings, 1970; Metropolis *et al.*, 1953) y el muestreador de Gibbs (Gelfand y Smith, 1990).

2.5.1 Algoritmo Metropolis-Hasting

El algoritmo Metropolis-Hasting pertenece a una familia de métodos de simulación MCMC que son útiles para extraer muestras de las distribuciones *a posteriori* (Gelman *et al.*, 2004). Este algoritmo fue desarrollado por Metropolis *et al.* (1953) y generalizado por Hastings (1970). Una descripción más detallada sobre el algoritmo Metropolis-Hasting se tiene en: Barker (1965), Peskin (1973), Tierney (1994), y Chib y Edward (1995).

Sea $q\left(\boldsymbol{\theta}, \boldsymbol{\vartheta}\right)$ una densidad propuesta, el cual se denomina como *densidad generadora de candidatos*, tal que:

$$\int q\left(\boldsymbol{\theta}, \boldsymbol{\vartheta}\right) d\boldsymbol{\vartheta} = 1.$$

Sea $U(0,1)$ que indica la distribución uniforme sobre $(0,1)$ entonces, una versión del algoritmo Metropolis-Hastings para el muestreo de la distribución *a posteriori* $\pi(\boldsymbol{\theta}|\boldsymbol{y})$ se describe a continuación (Chen *et al.*, 2012):

1. Seleccionar un punto inicial arbitrario $\boldsymbol{\theta}_0$ y sea $i = 0$.
2. Generar un punto candidato $\boldsymbol{\theta}^*$ de $q(\boldsymbol{\theta}_i, .)$ y u de $U(0,1)$.
3. Sea $\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}^*$ si $u \leq a(\boldsymbol{\theta}_i, \boldsymbol{\theta}^*)$, y de otro modo $\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i$, donde la probabilidad de aceptación está dado por:

$$a(\boldsymbol{\theta}, \boldsymbol{\vartheta}) = \text{mín}\left\{\frac{\pi(\boldsymbol{\vartheta}|\boldsymbol{y})\, q(\boldsymbol{\vartheta}, \boldsymbol{\theta})}{\pi(\boldsymbol{\theta}|\boldsymbol{y})\, q(\boldsymbol{\theta}, \boldsymbol{\vartheta})}, 1\right\}.$$

4. Sea $i = i + 1$ y repetir los pasos 2-3.

El algoritmo anterior es muy general ya que diversos algoritmos son un caso especial del algoritmo Metropolis-Hastings como el muestreador de Gibbs (Gelfand y Smith, 1990), el algoritmo "hit-and-run" (Smith, 1984), el algoritmo de ponderación dinámica (Liu *et al.*, 2001), entre otros (Chen *et al.*, 2012).

Este algoritmo se aplica en modelos lineales mixtos generalizados, modelos lineales dinámicos, modelos lineales dinámicos generalizados, modelos espaciales, entre otros (Gelman *et al.*, 2004).

2.5.2 Muestreador de Gibbs

El muestreador de Gibbs puede ser uno de los algoritmos de muestreo MCMC más conocidos en la literatura computacional bayesiana (Chen *et al.*, 2012). El término formal del muestreador de Gibbs fue introducido por Geman y Geman (1984). El principal punto de referencia bibliográfico del muestreador de Gibbs en problemas de inferencia bayesiana es Gelfand y Smith (1990). Tanner y Wong (1987) así como Casella y George (1992) presentan una descripción más detallada de este algoritmo.

Sea $\boldsymbol{\theta} = (\theta_1, \theta_2, ..., \theta_p)^t$ un vector de parámetros p-dimensional y sea $\pi(\boldsymbol{\theta}|\boldsymbol{y})$ la distribución *a posteriori* dado los datos $\boldsymbol{y}$. Entonces un esquema básico del algoritmo del muestreador de Gibbs es:

1. Seleccionar un punto inicial arbitrariamente

$$\boldsymbol{\theta}_0 = (\theta_{1,0}, \theta_{2,0}, ..., \theta_{p,0})^t$$

y sea $i = 0$

2. Generar $\boldsymbol{\theta}_{i+1} = (\theta_{1,i+1}, \theta_{2,i+1}, ..., \theta_{p,i+1})^t$ como sigue:

   - Generar $\theta_{1,i+1} \sim \pi(\theta_1 | \theta_{2i}, ..., \theta_{pi}, D)$;
   - Generar $\theta_{2,i+1} \sim \pi(\theta_2 | \theta_{1,i+1}, \theta_{3,i}, ..., \theta_{p,i}, D)$;
   - $\vdots$
   - Generar $\theta_{p,i+1} \sim \pi(\theta_p | \theta_{1,i+1}, \theta_{2,i+1}, ..., \theta_{p-1,i+1}, D)$,

   donde D denota los datos.

3. Sea $i = i + 1$, y repetir el paso 2.

Gelfand y Smith (1990) demuestran que bajo ciertas condiciones de regularidad, las secuencias $\{\boldsymbol{\theta}_i;\ i = 1, 2, ...\}$ tiene una distribución estacionaria $\pi(\boldsymbol{\theta}|D)$ (Chen *et al.*, 2012). La aplicación del muestreador de Gibbs se utiliza cuando es fácil obtener las distribuciones condicionales completas de los parámetros involucrados en el modelo. Se utiliza principalmente para modelos jerárquicos y modelos espaciales.

2.6 Algoritmo Esperanza-Maximización

El algoritmo Esperanza-Maximización (EM) se aplica en diversas áreas como en la genética, distribuciones de mezcla, econometría, estudios clínicos y sociológicos (Moon, 1996). En el campo de la estadística, se utiliza en aprendizaje automático, minería de datos, etc. Este algoritmo se utiliza para la estimación de parámetros de interés cuando existen datos perdidos (Dempster *et al.*, 1977). También se puede aplicar cuando se tienen datos latentes; es decir, datos que nunca se pretendieron ser observados. El algoritmo EM consiste en dos etapas: i) Esperanza y ii) Maximización.

2.6.1 El algoritmo clásico EM

Suponiendo que el conjunto de datos completos consiste de $\mathcal{Z} = (\mathcal{X}, \mathcal{Y})$ pero solamente $\mathcal{X}$ es observado.

La log-verosimilitud de los datos completos se denota por $\ell(\boldsymbol{\theta}; \mathcal{X}, \mathcal{Y})$ donde $\boldsymbol{\theta}$ es el vector de parámetros desconocidos para el cual se desea encontrar el estimador de máxima verosimilitud de $\boldsymbol{\theta}$.

El algoritmo consiste en (Haugh, 2015):

1) **Paso E**: Se calcula el valor esperado de $\ell(\boldsymbol{\theta}; \mathcal{X}, \mathcal{Y})$ dado los datos observado $\mathcal{X}$, y la estimación del parámetro actual, $\boldsymbol{\theta}_{old}$. Así, se define:

$$\begin{aligned} Q(\boldsymbol{\theta}; \boldsymbol{\theta}_{old}) &= E\left[\ell(\boldsymbol{\theta}; \mathcal{X}, \mathcal{Y}) \,|\mathcal{X}, \boldsymbol{\theta}_{old}\right] \\ &= \int l(\boldsymbol{\theta}; \mathcal{X}, y)\, p(y|\mathcal{X}, \boldsymbol{\theta}_{old})\, dy \end{aligned} \tag{2.30}$$

donde $p(.|\mathcal{X}, \boldsymbol{\theta}_{old})$ es la función de densidad condicional de $\mathcal{Y}$ dado los datos observados $\mathcal{X}$, y asumiendo $\boldsymbol{\theta} = \boldsymbol{\theta}_{old}$.

2) **Paso M**: Se maximiza para $\boldsymbol{\theta}$ la esperanza calculada en (2.30). Esto es,

$$\boldsymbol{\theta}_{new} := \underset{\boldsymbol{\theta}}{\text{máx}}\ Q(\boldsymbol{\theta}; \boldsymbol{\theta}_{old})$$

Se actualiza el parámetro $\boldsymbol{\theta}$, es decir $\boldsymbol{\theta}_{old} = \boldsymbol{\theta}_{new}$, y se repiten los dos pasos hasta que la secuencia $\boldsymbol{\theta}^t_{new}$ converja. Bajo condiciones muy generales, se puede garantizar la convergencia a un máximo local. Si se sospecha que la función de la log-verosimilitud tiene muchos máximos locales entonces el programa que implementa el algoritmo EM debe de ejecutarse varias veces utilizando diferentes valores iniciales de $\boldsymbol{\theta}_{old}$ en cada ocasión. El estimador de máxima verosimilitud de $\boldsymbol{\theta}$ se toma como el mejor de los conjuntos de máximos locales obtenidos de las distintas ejecuciones del programa que implementa el algoritmo.

2.6.2 Aplicación bayesiana

El algoritmo EM puede utilizarse para calcular la moda de la distribución *a posteriori* $p(\boldsymbol{\theta}|\mathcal{X})$, en un contexto bayesiano donde se da una distribución *a priori*, $p(\boldsymbol{\theta})$, en el parámetro desconocido, $\boldsymbol{\theta}$.

Se puede escribir la distribución *a posteriori*,

$$p(\boldsymbol{\theta}|\mathcal{X}) = \frac{p(\mathcal{X}|\boldsymbol{\theta})\, p(\boldsymbol{\theta})}{p(\mathcal{X})}, \tag{2.31}$$

tomando el logaritmo en (2.31) se tiene,

$$\log\ p(\boldsymbol{\theta}|\mathcal{X}) = \log\ p(\mathcal{X}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) - \log\ p(\mathcal{X}). \tag{2.32}$$

Realizando una re-formulación más general del algoritmo EM (vease Haugh, 2015) podemos escribir (2.32) como:

$$\ell(\boldsymbol{\theta}; \mathcal{X}) := \log\ p(\mathcal{X}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + KL\left(q \parallel p_{\mathcal{Y}|\mathcal{X}}\right), \tag{2.33}$$

donde $\mathcal{L}(q, \boldsymbol{\theta})$ y $KL\left(q \parallel p_{\mathcal{Y}|\mathcal{X}}\right)$ es la verosimilitud y la divergencia Kullback-Leibler (KL) respectivamente, y sustituyendo (2.32) en (2.33) se tiene,

$$\log\ p(\boldsymbol{\theta} \mid \mathcal{X}) = \mathcal{L}(q, \boldsymbol{\theta}) + KL\left(q \parallel p_{\mathcal{Y}|\mathcal{X}} + \log\ p(\boldsymbol{\theta})\right) - \log\ p(\mathcal{X})$$

Con esto se puede encontrar la moda *a posteriori* de $\log p(\boldsymbol{\theta} \mid \mathcal{X})$.

2.6.3 Estimación en el modelo lineal mixto

Considere el modelo mixto definido en (2.9), y sea $\boldsymbol{u} \mid \sigma_u^2 \boldsymbol{A} \sim NM(\mathbf{0}, \sigma_u^2 \boldsymbol{A})$ y $\boldsymbol{e} \mid \sigma_e^2 \sim NM(\mathbf{0}, \sigma_e^2 \boldsymbol{I})$, donde $\boldsymbol{u}$ y $\boldsymbol{e}$ son independientes. El interés se centra en la inferencia de $\boldsymbol{\theta}$, donde $\boldsymbol{\theta} = \left(\boldsymbol{\beta}^t, \sigma_e^2, \sigma_u^2\right)^t$, para detalle véase Sorensen y Gianola (2007).

La verosimilitud de los datos observados del modelo (2.9) es:

$$\mathcal{L}(\boldsymbol{\theta} \mid \boldsymbol{y}) = \int p\left(\boldsymbol{y} \mid \boldsymbol{\beta}, \boldsymbol{u}, \sigma_e^2\right)\ p\left(\boldsymbol{u} \mid \boldsymbol{A}\sigma_u^2\right) \partial \boldsymbol{u}, \tag{2.34}$$

donde $p(\boldsymbol{y} \mid \boldsymbol{\beta}, \boldsymbol{u}, \sigma_e^2)$ corresponde a la función de densidad de una variable aleatoria con distribución $NM(\boldsymbol{X\beta} + \boldsymbol{Zu}, \sigma_e^2\boldsymbol{I})$ mientras que $p(\boldsymbol{u} \mid \sigma_u^2\boldsymbol{A})$ corresponde a la densidad de una variable aleatoria con distribución $NM(\boldsymbol{0}, \sigma_u^2\boldsymbol{A})$. Desarrollando la ecuación (2.34) se tiene:

$$\mathcal{L}(\boldsymbol{\theta} \mid \boldsymbol{y}) \propto \mid \boldsymbol{V} \mid^{-1/2} \exp\left\{-\frac{1}{2}(\boldsymbol{y} - \boldsymbol{X\beta})^t \boldsymbol{V}^{-1}(\boldsymbol{y} - \boldsymbol{X\beta})\right\}, \tag{2.35}$$

donde $\boldsymbol{V} = \sigma_u^2\boldsymbol{ZAZ}^t + \sigma_e^2\boldsymbol{I}$ es la matriz de varianzas-covarianzas de los datos observados $\boldsymbol{y}$. La estimación de $\boldsymbol{\theta}$ se obtiene utilizando el algoritmo EM, considerando los efectos aleatorios $\boldsymbol{u}$ como datos faltantes, entonces la verosimilitud de los datos completos es:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{u} \mid \boldsymbol{y}) = \mid \sigma_e^2 \mid^{-1/2} exp\left[-\frac{1}{2\sigma_e^2}(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Zu})\right] \times \mid \sigma_u^2\boldsymbol{A} \mid^{-1/2} exp\left[-\frac{1}{2\sigma_u^2}\boldsymbol{u}^t\boldsymbol{Au}\right]. \tag{2.36}$$

Aplicando el logaritmo en (2.36), se tiene la log-verosimilitud de los datos completos,

$$\ell(\boldsymbol{\theta}, \boldsymbol{u} \mid \boldsymbol{y}) = cte - \frac{n}{2}\log\sigma_e^2 - \frac{q}{2}\log\sigma_u^2 - \frac{1}{2\sigma_u^2}\boldsymbol{uA}^{-1}\boldsymbol{u} - \frac{1}{2\sigma_e^2}(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Zu})^t(\boldsymbol{y} - \boldsymbol{X\beta} - \boldsymbol{Zu}). \tag{2.37}$$

Para el algoritmo EM se requiere tomar esperanzas respecto a $\boldsymbol{u} \mid \boldsymbol{\theta}^{[s]}$, de acuerdo a la definición de $Q\left(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{[s]}\right)$ en (2.30), donde s es una variable que indica el número de iteración del algoritmo. El paso E se define como:

$$Q\left(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{[s]}\right) = \int \log p(\boldsymbol{\theta}, \boldsymbol{u} \mid \boldsymbol{y})\ p\left(\boldsymbol{u} \mid \boldsymbol{\theta}^{[s]}, \boldsymbol{y}\right) \partial \boldsymbol{u} \tag{2.38}$$

Sustituyendo (2.37) y la distribución de $p\left(\boldsymbol{u} \mid \sigma_u^2 \boldsymbol{A}\right)$ en (2.38):

$$\begin{aligned} Q\left(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{[s]}\right) &= -\frac{n}{2}\log\sigma_e^2 - \frac{q}{2}\log\sigma_u^2 - \frac{1}{2\sigma_u^2}E_{\boldsymbol{u}|\boldsymbol{y},\boldsymbol{\theta}^{[s]}}\left[\boldsymbol{u}^t\boldsymbol{A}^{-1}\boldsymbol{u}\right] \\ &\quad -\frac{1}{2\sigma_e^2}E_{\boldsymbol{u}|\boldsymbol{y},\boldsymbol{\theta}^{[s]}}\left[(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{Z}\boldsymbol{u})^t(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{Z}\boldsymbol{u})\right]. \end{aligned} \tag{2.39}$$

Sean

$$\begin{aligned} \tilde{\boldsymbol{u}} &= E_{\boldsymbol{u}|\boldsymbol{y},\boldsymbol{\theta}}\left(\boldsymbol{u} \mid \boldsymbol{y},\boldsymbol{\theta}\right) \\ &= \sigma_u^2\boldsymbol{A}\boldsymbol{Z}^t\boldsymbol{V}^{-1}\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}\right) \\ &= \left(\boldsymbol{Z}^t\boldsymbol{Z}+\frac{\sigma_e^2}{\sigma_u^2}\boldsymbol{A}^{-1}\right)^{-1}\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}\right), \end{aligned} \tag{2.40}$$

y

$$\begin{aligned} \tilde{\boldsymbol{W}} &= Var_{\boldsymbol{u}|\boldsymbol{y},\boldsymbol{\theta}}\left(\boldsymbol{u} \mid \boldsymbol{y},\boldsymbol{\theta}\right) \\ &= \sigma_u^2\boldsymbol{A} - \sigma_u^4\boldsymbol{A}\boldsymbol{Z}^t\boldsymbol{V}^{-1}\boldsymbol{Z}\boldsymbol{A} \\ &= \sigma_e^2\left(\boldsymbol{Z}^t\boldsymbol{Z}+\frac{\sigma_e^2}{\sigma_u^2}\boldsymbol{A}^{-1}\right)^{-1}. \end{aligned} \tag{2.41}$$

Utilizando los resultados para la esperanza de formas cuadráticas y sustituyendo en (2.39) se tiene,

$$\begin{aligned} Q\left(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{[s]}\right) = &-\frac{n}{2}\log\sigma_e^2 - \frac{q}{2}\log\sigma_u^2 - \frac{1}{2\sigma_u^2}\left[\tilde{\boldsymbol{u}}^{[s]t}\boldsymbol{A}^{-1}\tilde{\boldsymbol{u}}^{[s]} + tr\left(\boldsymbol{A}^{-1}\tilde{\boldsymbol{W}}^{[s]}\right)\right] \\ &-\frac{1}{2\sigma_e^2}\left[\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{Z}\tilde{\boldsymbol{u}}^{[s]'}\right)^t\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{Z}\tilde{\boldsymbol{u}}^{[s]'}\right) + tr\left(\boldsymbol{Z}^t\boldsymbol{Z}\tilde{\boldsymbol{W}}^{[s]}\right)\right]. \end{aligned} \tag{2.42}$$

En el paso M hay que maximizar (2.42), usando técnicas de derivadas e igualando a cero,

se tiene:

$$\frac{\partial Q\left(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{[s]}\right)}{\partial \boldsymbol{\beta}} = \frac{1}{\sigma_e^2}\boldsymbol{X}^t\left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\tilde{\boldsymbol{u}}^{[s]}\right) = \boldsymbol{0}, \tag{2.43}$$

$$\frac{\partial Q\left(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{[s]}\right)}{\partial \sigma_u^2} = -\frac{q}{2\sigma_u^2} + \frac{1}{2\left(\sigma_u^2\right)^2}\left[\tilde{\boldsymbol{u}}^{[s]\,t}\boldsymbol{A}^{-1}\tilde{\boldsymbol{u}}^{[s]} + tr\left(\boldsymbol{A}^{-1}\tilde{\boldsymbol{W}}^{[s]}\right)\right] = 0, \tag{2.44}$$

$$\frac{\partial Q\left(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{[s]}\right)}{\partial \sigma_e^2} = -\frac{n}{2\sigma_e^2} + \frac{1}{2\left(\sigma_e^2\right)^2}\left[\left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\tilde{\boldsymbol{u}}^{[s]}\right)^t\left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Z}\tilde{\boldsymbol{u}}^{[s]}\right)\right] + \frac{1}{2\left(\sigma_e^2\right)^2}tr\left(\boldsymbol{Z}^t\boldsymbol{Z}\tilde{\boldsymbol{W}}^{[s]}\right) = 0. \tag{2.45}$$

Resolviendo el sistema de ecuaciones dadas por (2.43)-(2.45) se obtienen las ecuaciones (2.46)-(2.48), mismas que se utilizan para implementar el algoritmo iterativo para obtener los estimadores de máxima verosimilitud de los parámetros de interés,

$$\boldsymbol{\beta}^{[s+1]} = \left(\boldsymbol{X}^t\boldsymbol{X}\right)^{-1}\boldsymbol{X}^t\left(\boldsymbol{y} - \boldsymbol{Z}\tilde{\boldsymbol{u}}^{[s]}\right), \tag{2.46}$$

$$\sigma_u^{2[s+1]} = \frac{1}{q}\left[\tilde{\boldsymbol{u}}^{[s]\,t}\boldsymbol{A}^{-1}\tilde{\boldsymbol{u}}^{[s]} + tr\left(\boldsymbol{A}^{-1}\tilde{\boldsymbol{W}}^{[s]}\right)\right], \tag{2.47}$$

$$\sigma_e^{2[s+1]} = \frac{1}{n}\left[\left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}^{[s]} - \boldsymbol{Z}\tilde{\boldsymbol{u}}^{[s]}\right)^t\left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}^{[s]} - \boldsymbol{Z}\tilde{\boldsymbol{u}}^{[s]}\right) + tr\left(\boldsymbol{Z}^t\boldsymbol{Z}\tilde{\boldsymbol{W}}^{[s]}\right)\right]. \tag{2.48}$$

El algoritmo EM para este problema se describe a continuación.

Algoritmo EM

1. Definir los valores iniciales de $\boldsymbol{\theta}$,

$$\boldsymbol{\theta}^{[0]} = \left(\boldsymbol{\beta}^{[0]t}, \sigma_e^{2[0]}, \sigma_u^{2[0]}\right)^t$$

con $s = 0$

1. Calcular $\tilde{\boldsymbol{u}}^{[s]}$ y $\tilde{\boldsymbol{W}}^{[s]}$ mediante (2.40) y (2.41).

2. Sea $s = s + 1$, obtener $\boldsymbol{\theta}^{[s]} = \left(\boldsymbol{\beta}^{[s]t}, \sigma_e^{2[s]}, \sigma_u^{2[s]}\right)^t$ mediante (2.46), (2.47) y (2.48) respectivamente.

3. Repetir los pasos 1 y 2 hasta convergencia.

2.7 Bayes variacional

Como se describió anteriormente, los métodos MCMC se utilizan principalmente en la inferencia bayesiana para aproximar distribuciones *a posteriori* pero este método resulta intensivo desde el punto de vista computacional en modelos grandes y complejos que presentan una gran cantidad de parámetros desconocidos para inferir. El Bayes variacional (VB, Variational Bayes en inglés) es una alternativa atractiva a los métodos MCMC ya que son computacionalmente más eficientes y se pueden aplicar a una amplia gama de modelos probabilísticos. El VB es una técnica basada en la optimización para la inferencia bayesiana y pertenece a la clase más grande de los métodos de aproximación variacional o inferencia variacional los cuales se pueden utilizar en el contexto frecuentista para la estimación de máxima verosimilitud cuando hay datos perdidos (Tran *et al.*, 2021). En cuanto a la inferencia bayesiana, el VB aproxima la distribución *a posteriori* por una distribución de probabilidad con densidad $q(\boldsymbol{\theta})$ perteneciente a alguna familia de distribuciones tratables $\mathcal{Q}$.

De acuerdo a Tran *et al.* (2021) se tiene que la mejor aproximación VB, $q^* \in \mathcal{Q}$, se encuentra minimizando la divergencia KL de $q(\boldsymbol{\theta})$ a $p(\boldsymbol{\theta} \mid y)$,

$$q^* = \underset{q \in \mathcal{Q}}{\arg\min} \left\{ KL\left(q \mid\mid p(\cdot \mid y)\right) \right\} := \left\{ q(\boldsymbol{\theta}) \ \log \frac{q(\boldsymbol{\theta})}{p(\boldsymbol{\theta} \mid y)} d\boldsymbol{\theta} \right\}, \tag{2.49}$$

entonces la inferencia bayesiana se realiza con la distribución *a posteriori* intratable, $p(\boldsymbol{\theta} \mid y)$, reemplazada por la aproximación VB tratable, $q^*(\boldsymbol{\theta})$,

$$KL\left(q \mid\mid p(\cdot \mid y)\right) = -\int q(\boldsymbol{\theta}) \log \frac{p(\boldsymbol{\theta})\, p(y \mid \boldsymbol{\theta})}{q(\boldsymbol{\theta})} d\boldsymbol{\theta} + \log \ p(y).$$

Por lo tanto, minimizar KL equivale a maximizar el límite inferior en $\log p(y)$,

$$LB\left(q\right) := \int q\left(\boldsymbol{\theta}\right) \log \frac{p\left(\boldsymbol{\theta}\right) p\left(y \mid \boldsymbol{\theta}\right)}{q\left(\boldsymbol{\theta}\right)} d\boldsymbol{\theta} = E_q \left(\log \frac{p\left(\boldsymbol{\theta}\right) p\left(y \mid \boldsymbol{\theta}\right)}{q\left(\boldsymbol{\theta}\right)} \right).$$

Sin ninguna restricción en $\mathcal{Q}$, la solución de (2.49) es $q^*\left(\boldsymbol{\theta}\right) = p\left(\boldsymbol{\theta} \mid y\right)$, pero esta esta solución es inútil ya que es en si misma intratable (Tran *et al.*, 2021) por lo que se tiene que imponer alguna restricción en $\mathcal{Q}$.

Dependiendo de la restricción impuesta en $\mathcal{Q}$, los algoritmos VB pueden categorizarse en dos clases: Mean Field VB (MFVB) y Fixed Form VB (FFVB).

El MFVB asume la siguiente factorización de q, $q\left(\boldsymbol{\theta}\right) = q_1\left(\boldsymbol{\theta}_1\right) q_2\left(\boldsymbol{\theta}_2\right)$, el cual ignora la dependencia *a posteriori* entre $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$ e intenta aproximar $p\left(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \mid y\right)$ por $q\left(\boldsymbol{\theta}\right) = q_1\left(\boldsymbol{\theta}_1\right) q_2\left(\boldsymbol{\theta}_2\right)$. Para detalle veáse Tran *et al.* (2021).

Algoritmo MFVB

1. Inicializar el parámetro de $q_1\left(\boldsymbol{\theta}_1\right)$.
2. Dado $q_1\left(\boldsymbol{\theta}_1\right)$, actualizamos el parámetro de $q_2\left(\boldsymbol{\theta}_2\right)$ utilizando,

$$q_2\left(\boldsymbol{\theta}_2\right) \propto \exp\left(E_{-\theta_2}\left[\log\ p\left(y, \boldsymbol{\theta}\right)\right]\right) = \exp\left(\int q_1\left(\boldsymbol{\theta}_1\right) \log\ p\left(y, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2\right) d\boldsymbol{\theta}_1 \right).$$

3. Dado $q_2\left(\boldsymbol{\theta}_2\right)$, actualizamos el parámetro de $q_1\left(\boldsymbol{\theta}_1\right)$ utilizando,

$$q_1\left(\boldsymbol{\theta}_1\right) \propto \exp\left(E_{-\theta_1}\left[\log\ p\left(y, \boldsymbol{\theta}\right)\right]\right) = \exp\left(\int q_2\left(\boldsymbol{\theta}_2\right) \log\ p\left(y, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2\right) d\boldsymbol{\theta}_2 \right).$$

4. Repetimos el paso 2 y 3 hasta que se cumpla la condición de paro.

Una condición para parar el algoritmo es terminar la actualización si el cambio en los parámetros del VB *a posteriori*, $q\left(\boldsymbol{\theta}\right) = q_1\left(\boldsymbol{\theta}_1\right) q_2\left(\boldsymbol{\theta}_2\right)$ entre dos iteraciones consecutivas es menor a un umbral ϵ.

Este procedimiento es fácil de extender a un caso más general donde $\boldsymbol{\theta}$ se divide en k bloques, $\boldsymbol{\theta} = \left(\boldsymbol{\theta}_1^t, \boldsymbol{\theta}_2^t, \ldots \boldsymbol{\theta}_k^t\right)$ y se quiere aproximar la distribución *a posteriori*, $p\left(\boldsymbol{\theta}_1^t, \boldsymbol{\theta}_2^t, \ldots \boldsymbol{\theta}_k^t \mid y\right)$ por $q\left(\boldsymbol{\theta}\right) = q_1\left(\boldsymbol{\theta}_1\right) q_2\left(\boldsymbol{\theta}_2\right) \cdots q_k\left(\boldsymbol{\theta}_k\right)$.

El óptimo $q_j(\boldsymbol{\theta}_j)$ que maximiza $LB(q)$ cuando $q_1, \ldots, q_{j-1}, q_{j+1}, \ldots, q_k$ son fijos es,

$$q_j(\boldsymbol{\theta}_j) \exp\left(E_{-\boldsymbol{\theta}_j}[log\ p(y, \boldsymbol{\theta})]\right),\ j = 1, \ldots, k,$$

donde $E_{-\boldsymbol{\theta}_j}(\cdot)$ denota la esperanza,

$$E_{-\boldsymbol{\theta}_j}[log\ p(y, \boldsymbol{\theta})] := \int q_1(\boldsymbol{\theta}_1) \ldots q_{j-1}(\boldsymbol{\theta}_{j-1})\, q_{j+1}(\boldsymbol{\theta}_{j+1}) \ldots q_k(\boldsymbol{\theta}_k) \log\ p(y, \boldsymbol{\theta})\, d\boldsymbol{\theta}_1 \cdots d\boldsymbol{\theta}_{j-1} d\boldsymbol{\theta}_{j+1} \cdots d\boldsymbol{\theta}_k.$$

Se puede desarrollar un procedimiento similar al algoritmo anterior donde se inician los parámetros en los $k-1$ factores, $q_1, \ldots, q_{k-1}$, para después actualizar q_k y los otros factores de forma recursiva.

El FFVB asume una forma paramétrica fija para la aproximación VB de la densidad q, $q = q_\lambda$, el cual pertenece a una clase de distribuciones $\mathcal{Q}$ indexada por un vector λ llamado parámetro variacional. El FFVB encuentra el mejor q_λ en $\mathcal{Q}$ optimizando el límite inferior.

$$LB(\lambda) := LB(q_\lambda) = E_{q_\lambda}\left[log \frac{p(\boldsymbol{\theta})\, p(y \mid \boldsymbol{\theta})}{q_\lambda(\boldsymbol{\theta})}\right] = E_{q_\lambda}[h_\lambda(\boldsymbol{\theta})],$$

donde $h_\lambda(\boldsymbol{\theta}) := log \frac{p(\boldsymbol{\theta}) p(y|\boldsymbol{\theta})}{q_\lambda(\boldsymbol{\theta})}$.

Algoritmo FFBV

1. Inicializar $\lambda^{(0)}$ y detener la iteración siguiente si se cumple el criterio de paro.
2. Para $t = 0, 1, \ldots$
   - Generar $\theta_s \sim q_{\lambda(t)}(\boldsymbol{\theta}),\ s = 1, \ldots, S$,

- Calcular el estimador insesgado del gradiente LB,

$$\nabla_\lambda LB\left(\lambda^{(t)}\right) := \frac{1}{S}\sum_{s=1}^{S} \nabla_\lambda log\ q_\lambda\left(\theta_s\right)\ \times h_\lambda\left(\theta_s\right)|_{\lambda=\lambda^{(t)}},$$

- Actualizar

$$\lambda^{(t+1)} = \lambda^{(t)} + a_t \widehat{\nabla_\lambda LB}\left(\lambda^{(t)}\right).$$

donde $\nabla_\lambda LB\left(\lambda\right) = E_{q\lambda}\left[\nabla_\lambda log\ q_\lambda\left(\boldsymbol{\theta}\right) \times h_\lambda\left(\boldsymbol{\theta}\right)\right]$. El parámetro S del algoritmo se refiere al número de muestras Monte Carlo.

2.7.1 Bayes variacional aplicado al modelo lineal mixto

De acuerdo a Ormerod y Wand (2010), la versión bayesiana del modelo lineal mixto es de la forma general,

$$\boldsymbol{y} \mid \boldsymbol{\beta}, \boldsymbol{u}, \boldsymbol{G}, \boldsymbol{R} \sim NM\left(\boldsymbol{X\beta} + \boldsymbol{Zu}, \boldsymbol{R}\right), \boldsymbol{u} \mid \boldsymbol{G} \sim NM(\boldsymbol{0}, \boldsymbol{G}),$$

donde $\boldsymbol{y}$ es el vector de dimensión n de la variable respuesta, $\boldsymbol{\beta}$ es el vector de dimensión p de los efectos fijos, $\boldsymbol{u}$ es el vector de dimensión q de los efectos aleatorios, $\boldsymbol{X}$ y $\boldsymbol{Z}$ son las matrices diseño correspondiente a los efectos fijos y aleatorios, $\boldsymbol{G}$ y $\boldsymbol{R}$ son las matrices de varianzas-covarianzas.

De acuerdo a Ormerod y Wand (2010) las matrices de varianzas-covarianzas se definen como $\boldsymbol{G} = blockdiag\left(\sigma^2_{u1}\boldsymbol{I}_{k1}, \ldots, \sigma^2_{ur}\boldsymbol{I}_{kr}\right)$ y $\boldsymbol{R} = \sigma^2_e\boldsymbol{I}$, y asignando las siguientes distribuciones *a priori*:

$$\boldsymbol{\beta} \sim NM\left(\boldsymbol{0}, \sigma^2_\beta\boldsymbol{I}\right)$$

$$\sigma^2_{ul} \sim IG\left(A_{ul}, B_{ul}\right),\ 1 \leq l \leq r$$

$$\sigma^2_e \sim IG\left(A_e, B_e\right)$$

para algún σ^2_β, A_{ul}, B_{ul}, A_e, $B_e > 0$.

Una solución se da con el producto de dos componentes:

$$q\left(\boldsymbol{\beta}, \boldsymbol{u}, \sigma^2_{u1}, \ldots, \sigma^2_{ur}, \sigma^2_e\right) = q_{\boldsymbol{\beta},\boldsymbol{u}}\left(\boldsymbol{\beta}, \boldsymbol{u}\right) q_{\sigma^2}\left(\sigma^2_{u1}, \ldots, \sigma^2_{ur}, \sigma^2_e\right).$$

Aplicando el VB obtenemos las densidades óptimas de la siguiente forma, $q^*_{\boldsymbol{\beta},\boldsymbol{u}}\left(\boldsymbol{\beta}, \boldsymbol{u}\right)$ es la función de densidad normal multivariada, y $q^*_{\sigma^2}$ es el producto de $r+1$ funciones de la densidad gamma inversa.

Sean $\boldsymbol{\mu}_{q(\boldsymbol{\beta},\boldsymbol{u})}$ y $\boldsymbol{\sigma}_{q(\boldsymbol{\beta},\boldsymbol{u})}$ la media y la matriz de covarianza para la densidad $q^*_{\boldsymbol{\beta},\boldsymbol{u}}$ y sea el conjunto $\boldsymbol{C}\left[\boldsymbol{X}\ \boldsymbol{Z}\right]$. Para la densidad $q^*_{\sigma^2}$, el parámetro de forma para los $r+1$ componentes pueden mostrarse como determinístico, $A_{u1} + \frac{1}{2}K_1, \ldots, A_{ur} + \frac{1}{2}K_r$ y $A_e + \frac{1}{2}n$. Sean $B_{q\left(\sigma^2_{u1}\right)}, \ldots, B_{q(\sigma^2_{ur})}, B_{q(\sigma^2_e)}$ los parámetros de la tasa de acompañamientos. La relación entre $\left(\boldsymbol{\mu}_{q(\boldsymbol{\beta},\boldsymbol{u})}, \boldsymbol{\sigma}_{q(\boldsymbol{\beta},\boldsymbol{u})}\right)$ y $\left(B_{q\left(\sigma^2_{u1}\right)}, \ldots, B_{q(\sigma^2_{ur})}, B_{q(\sigma^2_e)}\right)$ conduce al siguiente esquema iterativo en el algoritmo (Ormerod y Wand, 2010).

Algoritmo.

- Inicializar: $B_{q(\sigma^2_e)}, B_{q\left(\sigma^2_{u1}\right)}, \ldots, B_{q(\sigma^2_{ur})}$
- Ciclo:

$$\boldsymbol{\sigma}_{q(\boldsymbol{\beta},\boldsymbol{u})} \leftarrow \left\{\frac{A_e+\frac{n}{2}}{B_{q(\sigma^2_e)}} C^tC + blockdiag\left(\sigma^2_{\boldsymbol{\beta}}\boldsymbol{I}_p, \frac{A_{u1}+\frac{1}{2}K_1}{B_{q\left(\sigma^2_{u1}\right)}}\boldsymbol{I}_{K_1}, \ldots, \frac{A_{ur}+\frac{1}{2}K_r}{B_{q(\sigma^2_{ur})}}\right)\right\}^{-1},$$

$$\boldsymbol{\mu}_{q(\boldsymbol{\beta},\boldsymbol{u})} \leftarrow \left(\frac{A_e+\frac{n}{2}}{B_{q(\sigma^2_e)}}\right)\boldsymbol{\Sigma}_{q(\boldsymbol{\beta},\boldsymbol{u})}C^t\boldsymbol{y},$$

$$B_{q(\sigma^2_e)} \leftarrow B_e + \frac{1}{2}\left\{\parallel \boldsymbol{y} - C_{\boldsymbol{\mu}_{q(\boldsymbol{\beta},\boldsymbol{u})}} \parallel^2 + tr\left(C^tC\Sigma_{q(\boldsymbol{\beta},\boldsymbol{u})}\right)\right\},$$

$$B_{q(\sigma^2_{ur})} \leftarrow B_{ul} + \frac{1}{2}\left\{\parallel \boldsymbol{\mu}_{q(\boldsymbol{u}_l)} \parallel^2 + tr\left(\boldsymbol{\Sigma}_{q(\boldsymbol{u}_l)}\right)\right\}, \text{ para } 1 \leq l \leq r.$$

Una vez que converga el $\boldsymbol{\mu}^*_{q(\boldsymbol{\mu},\boldsymbol{u})}$, $\Sigma^*_{q(\boldsymbol{\beta},\boldsymbol{u})}$, $B^*_{q\sigma^2_{u1}}$, …, $B^*_{q(\sigma^2_{ur})}$ y $B^*_{q(\sigma^2_e)}$, las distribuciones *a posteriori* aproximadas son:

$p\left(\boldsymbol{\beta}, \boldsymbol{u} \mid \boldsymbol{y}\right) \approx N\left(\boldsymbol{\mu}^*_{q(\boldsymbol{\beta},\boldsymbol{u})}, \boldsymbol{\Sigma}^*_{q(\boldsymbol{\beta},\boldsymbol{u})}\right)$ y

el producto de funciones de densidades gama inversa, $p\left(\sigma^2_{u1}, \ldots, \sigma^2_{ur}, \sigma^2_e \mid \boldsymbol{y}\right) \approx IG\left(A_{ul} + \frac{1}{2}K_l, B^*_{q\left(\sigma^2_{ul}\right)}\right)$, $1 \leq l \leq r$, junto con la función de densidad $IG\left(A_\epsilon + \frac{1}{2}n, B^*_{q(\sigma^2_e)}\right)$.

2.8 Aproximación anidada integrada de Laplace (INLA)

La aproximación anidada integrada de Laplace (INLA, Integrated Nested Laplace Approximation en inglés) es un enfoque reciente para la inferencia bayesiana en el modelo Gaussiano latente (LGM, Latent Gaussian model en inglés) introducidos por Rue y Martino (2007), Rue *et al.* (2009) , Martino y Rue (2009).

El INLA se basa en una combinación de aproximaciones analíticas y esquemas eficientes de integración numérica para lograr aproximaciones deterministas de alta precisión para las cantidades *a posteriori* de interés. El beneficio principal de utilizar el INLA en lugar de los métodos MCMC es computacional ya que es más rápido en modelos grandes y complejos. El marco teórico detrás del INLA está detallado en Rue *et al.* (2009) (Martino y Rue, 2009). El INLA proporciona un método rápido para la inferencia bayesiana utilizando aproximaciones precisas para la densidad marginal *a posteriori* de los hiperparámetros $\boldsymbol{\theta}$, $p(\boldsymbol{\theta} \mid \boldsymbol{y})$, y la densidad marginal *a posteriori* de las variables latentes x_i, $i = 1, \ldots, n$, $p(x_i \mid \boldsymbol{y})$. Las aproximaciones de estas densidades pueden utilizarse para calcular los estadísticos de resumen de interés, tales como la media, varianza o cuantiles *a posteriori*. Los diferentes tipos de aproximaciones están disponibles en Rue y Martino (2007).

La clase de modelos que pueden ajustarse mediante el INLA es amplio, por ejemplo modelos de series de tiempo, modelos aditivos generalizados (Hastie y Tibshirani, 1987), modelos mixtos aditivos generalizados (Lin y Zhang, 1999), modelos geoaditivos (Kammann y Wand, 2003), modelos univariados de volatilidad (Taylor, 1994), entre otros (Martino y Rue, 2009).

Esta método está implementado en un paquete de R, el R-INLA (Martins *et al.*, 2013; Rue *et al.*, 2009) el cual representa una herramienta amigable con el usuario por su versatilidad para realizar inferencia bayesiana. El paquete INLA se encuentra en `www.r-inla.org`. Se tiene documentación detallada para su uso, ejemplos y tutoriales, además grupos de discusión y una bibliografía extensa.

Modelo Gaussiano Latente (LGM)

El INLA se aplica principalmente a una clase de modelos bayesianos conocido como modelo Gaussiano latente. Este modelo consiste de tres elementos: el modelo de verosimilitud, el campo Gaussiano latente y el vector de hiperparámetros (Martino y Rue, 2009). El vector de datos $\boldsymbol{y}$ se asume condicionalmente independiente dado el campo Gaussiano latente $\boldsymbol{x}$, por lo que el modelo de verosimilitud describe la distribución marginal de las observaciones. La media, u otra medida de tendencia central del LGM, de las observaciones y_i está ligado a un predictor lineal Gaussiano η_i a través de una función liga conocida:

$$\eta_i = \mu + \sum_j \beta_j z_{ij} + \sum_k w_k f^k \left(u_{ik}\right),$$

donde μ es el intercepto, $\boldsymbol{z}$ son covariables conocidas con efecto lineal $\boldsymbol{\beta}$, $\boldsymbol{w}$ es el vector de pesos conocidos y el término $\boldsymbol{f}^k$ se utiliza para modelar los efectos aleatorios de las covariable $\boldsymbol{u}$. Asignando una distribución *a priori* Gaussiana a μ, $\boldsymbol{\beta}$ y $\boldsymbol{f}^k$, $k = 1, \ldots, K$, el campo Gaussiano latente es $\boldsymbol{x} = \left\{\boldsymbol{\eta}, \mu, \boldsymbol{\beta}, \boldsymbol{f}^1, \boldsymbol{f}^2, \ldots\right\}$. El modelo completo es de la siguiente forma,

$$\begin{aligned}
\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta} &\sim \prod \pi \left(y_i \mid \eta_i, \boldsymbol{\theta}\right), \\
\boldsymbol{x} \mid \boldsymbol{\theta} &\sim \mathcal{N}\left(\mathbf{0}, Q^{-1}\left(\boldsymbol{\theta}\right)\right), \\
\boldsymbol{\theta} &\sim \pi\left(\boldsymbol{\theta}\right),
\end{aligned}$$

donde $Q\left(\boldsymbol{\theta}\right)$ es la matriz de precisión (inversa de la matriz de varianzas-covarianzas) del campo Gaussiano latente.

Mucho de los modelos que se utilizan comúnmente como *a priori* para los términos f^k pertenecen a la clase del campo aleatorio Markov Gaussiano (GMRF, Gaussian Markov Random Fields en inglés). Estos se pueden utilizar para modelar efectos suaves de covariables, efectos aleatorios, errores de medición, dependencia temporales, entre otros (Martino y Rue, 2009).

Para aplicar el INLA, el LGM debe cumplir los siguientes supuestos:

1. Cada punto de dato depende solo de un elemento $\boldsymbol{x}$ de modo que la verosimilitud se escribe como:

$$\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta} \sim \pi_i \left(y_i \mid \eta_i, \boldsymbol{\theta}\right).$$

2. El tamaño del vector de los hiperparámetros $\boldsymbol{\theta}$ es pequeño (< 15).

3. El campo latente $\boldsymbol{x}$, puede ser grande pero está dotado por algunas propiedades de independencia condicional de modo que la matriz de precisión $Q\left(\boldsymbol{\theta}\right)$ es dispersa.

4. El predictor lineal depende linealmente de la función suave desconocida de las covariables.

5. El interés inferencial radica en las densidades marginales *a posteriori*,

$$p\left(\theta_j \mid \boldsymbol{y}\right) = \int \int p\left(\boldsymbol{x}, \boldsymbol{\theta} \mid \boldsymbol{x}\right) d\ x d\boldsymbol{\theta}_{-j} = \int p\left(\boldsymbol{\theta} \mid \boldsymbol{y}\right) d\boldsymbol{\theta}_j \tag{2.50}$$

y

$$p\left(x_i \mid \boldsymbol{y}\right) = \int \int p\left(\boldsymbol{x}, \boldsymbol{\theta} \mid \boldsymbol{x}\right) dx_{-i} d\boldsymbol{\theta} = \int p\left(x_i \mid \boldsymbol{\theta}, \boldsymbol{y}\right) p\left(\boldsymbol{\theta} \mid \boldsymbol{y}\right) d\boldsymbol{\theta} \tag{2.51}$$

más que en la densidad conjunta *a posteriori* $p\left(\boldsymbol{x}, \boldsymbol{\theta} \mid \boldsymbol{y}\right)$.

De acuerdo a (Martino y Rue, 2009) el esquema de cálculo del INLA es el siguiente:

1. Explorar el espacio de $\boldsymbol{\theta}$ a través de la aproximación $\tilde{p}\left(\boldsymbol{\theta} \mid \boldsymbol{y}\right)$. Encontrar la moda de $\tilde{p}\left(\boldsymbol{\theta} \mid \boldsymbol{y}\right)$ y localizar la serie de puntos $\left\{\boldsymbol{\theta}^1, \ldots, \boldsymbol{\theta}^K\right\}$ en la zona de alta densidad de $\tilde{p}\left(\boldsymbol{\theta} \mid \boldsymbol{y}\right)$.

2. Para los K puntos de apoyos seleccionados calcular $\tilde{p}\left(\boldsymbol{\theta}^1 \mid \boldsymbol{y}\right), \ldots, \tilde{p}\left(\boldsymbol{\theta}^K \mid \boldsymbol{y}\right)$ utilizando,

$$\tilde{p}\left(\boldsymbol{\theta}^k \mid \boldsymbol{y}\right) \propto \frac{p\left(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{\theta}^k\right) p\left(\boldsymbol{x} \mid \boldsymbol{\theta}^k\right) p\left(\boldsymbol{\theta}^k\right)}{\tilde{p}_G\left(\boldsymbol{x} \mid \boldsymbol{\theta}^k, \boldsymbol{y}\right)}, \tag{2.52}$$

donde $\tilde{p}_G\left(\boldsymbol{x} \mid \boldsymbol{\theta}^k, \boldsymbol{y}\right)$ es una aproximación Gaussiana de la condicional completa $\boldsymbol{x} \mid \boldsymbol{\theta}^K, \boldsymbol{y}$.

3. Para cada punto seleccionado $\boldsymbol{\theta}^K$ aproximar la densidad de $x_i \mid \boldsymbol{\theta}, \boldsymbol{y}$ como $\tilde{p}\left(x_i \mid \boldsymbol{\theta}^k, \boldsymbol{y}\right), \ k = 1, \ldots, K$ utilizando una de las tres posibles aproximaciones: Laplace, Laplace simplificado o Gaussiano.

4. Resolver (2.51) vía integración numérica como:

$$\tilde{p}\left(x_i \mid \boldsymbol{y}\right) = \sum_{k=1}^{K} \tilde{p} = \int p\left(x_i \mid \boldsymbol{\theta}^k, \boldsymbol{y}\right) \tilde{p}\left(\boldsymbol{\theta}^k \mid \boldsymbol{y}\right) \Delta_k$$

   donde Δ_k son ponderaciones apropiadas, y la integral (2.50) se resuelve similarmente.

El INLA regresa valores de la ordenada predictiva condicional (CPO, por siglas en inglés) (Geisser y Eddy, 1979) y de la transformada integral de probabilidad (PIT, por sus siglas en inglés) (Marshall y Spiegelhalter, 2003) que se pueden utilizar para evaluar el modelo. También provee estimaciones por el criterio de información de la devianza (DIC, por sus siglas en inglés) (Spiegelhalter *et al.*, 2002), el criterio de información de Watanabe-Akaike (WAIC, por sus siglas en inglés) (Watanabe y Opper, 2010) y la verosimilitud marginal. La verosimilitud marginal es un criterio de selección de modelo bien establecido en las estadísticas bayesianas y se puede utilizar para promediar el modelo bayesiano. Más recientemente, Hubin y Storvik (2016) han estudiado la precisión de la estimación de la verosimilitud marginal proporcionada por el INLA encontrándola muy precisa (Martino y Riebler, 2014).

El paquete R-INLA (Martins *et al.*, 2013; Rue *et al.*, 2009) provee una implementación fácil de usar el método INLA. La definición del modelo en el R-INLA es similar a otros paquetes en R. Existen dos pasos esenciales:

1. Definir el predictor lineal a través del objeto `formula`.

2. Completar la definición del modelo y ajustar el modelo usando la función `inla()`.

El modelo ajustado se regresa como un objeto `inla`. La fórmula puede incluir efectos fijos y efectos aleatorios. Los términos no lineales y los efectos aleatorios se incluyen en la fórmula mediante la función `f()`. La especificación de diferentes modelos Gaussianos latentes, hiperprioris y opciones de ajuste de modelos es sencilla. Los resultados incluyen

las distribuciones marginales *a posteriores* de los efectos latentes y los hiperparámetros, así como estadísticas de resumen. Además, se pueden obtener estimaciones posteriores de combinaciones lineales o transformaciones del campo latente (Martins *et al.*, 2013). En R-INLA no existe la función "predecir" como en el caso de `glm` o lm. Las predicciones deben realizarse como parte del ajuste del modelo. Como se puede considerar que la predicción se ajusta a un modelo con datos faltantes, simplemente podemos establecer y[i] = NA para lo que se quiere predecir. Sin embargo, las distribuciones predictivas, que a menudo son de interés, no se devuelven directamente. En su lugar, se devuelven los marginales *a posteriori* para los efectos aleatorios y el predictor lineal en las ubicaciones faltantes (Martino y Riebler, 2014).

CAPÍTULO 3. lme4GS: UN PAQUETE DE R PARA SELECCIÓN GENÓMICA

La selección genómica (GS, Genomic Selection en inglés) es una tecnología utilizada para el mejoramiento genético, la cual tiene diversas ventajas sobre la selección basada en los fenotipos. Existen diversos modelos estadísticos que abordan los problemas estadísticos en la selección genómica como el modelo lineal mixto (LMM, linear mixed model en inglés). El desarrollo de softwares para ajustar este modelo es un área activa para muchas investigaciones, utilizado principalmente para las predicciones genómicas. El `lme4` es una paquete estándar para el ajuste del modelo lineal mixto y mixto generalizado en el paquete R, pero su uso para el análisis genético está limitado ya que no permite definir la correlación entre individuos o grupos de individuos.

En este capítulo se describe un nuevo paquete para R, `lme4GS`, el cual está enfocado en ajustar el LMM con estructuras de covarianza definidas por el usuario y en la predicción genómica de modelos utilizados para selección genómica. Se presentan varios ejemplos de modelos de selección genómica utilizando este paquete, así como el análisis de datos reales.

3.1 Introducción

En la última década han surgido tecnologías nuevas de bajo costos para el genotipado de alto rendimiendo, por lo que ha emergido un paradigma en selección genómica (Meuwissen *et al.*, 2001). La selección genómica combina datos moleculares y datos fenotípicos para obtener el valor de cría genómico estimado (GEBV, Genomic Estimated Breeding Values en inglés) de individuos que se han genotipado pero no fenotipado (Bernardo y Yu, 2007; Crossa *et al.*, 2010; de los Campos *et al.*, 2009; Hayes *et al.*, 2009; VanRaden *et al.*, 2009). La principal ventaja de la selección genómica sobre la selección basada en el fenotipo es

que se reduce el costo por ciclo y el tiempo requerido para el desarrollo de variedades. Sin embargo, diversos factores complican la aplicación práctica de la selección genómica, estos ocurren a diferentes niveles y están influenciados por muchos factores genéticos, ambientales y e incluso los modelos estadísticos utilizados.

Las dificultades en la selección genómica surgen cuando se determina (i) el tamaño y la diversidad de la población en el conjunto de entrenamiento, (ii) la relación entre el conjunto de entrenamiento y prueba, (iii) el desequilibrio del ligamiento, y (iv) la heredabilidad de los rasgos para ser predichos. Los desafíos en la selección genómica están relacionados con la alta dimensión de los datos de los marcadores, donde el número de marcadores es más grande que el número de observaciones, la multicolinealidad de los marcadores, la interacción entre marcadores, la complejidad de los rasgos, el tamaño de muestra, y la interacción genotipo-ambiente. Estas complejidades requieren de modelos estadísticos paramétricos y semi-paramétricos, especialmente de modelos mixtos, estimación bayesiana y recientemente, de métodos de aprendizaje automático profundo que pueden lidiar adecuadamente con los conjuntos de datos usualmente grandes (Crossa *et al.*, 2017). Esto ha llevado a desafíos computacionales debido al tamaño de los datos y desafíos estadísticos que incluyen el ajuste de modelos y la optimización de parámetros. Por lo tanto, el desarrollo de paquetes informáticos completos y sencillos para estimar los valores de cría de los individuos a ser seleccionados bajo estos complejos escenarios es crucial para una aplicación eficiente de la selección genómica.

El primer software en R (R Core Team, 2021) desarrollado para la predicción basado en el genoma fue presentado por de los Campos *et al.* (2009). Poco después, Pérez *et al.* (2010) describe formalmente el software `BLR` (Bayesian Linear Regression en inglés) que permite ajustar modelos de regresión lineal de alta dimensión que incluyen marcadores moleculares densos, información del pedigrí y otras covariables distintas a los marcadores. El paquete `BLR` en R descrito por Pérez *et al.* (2010) permite no solo incluir marcadores, sino también datos genealógicos de forma conjunta. Además, Pérez *et al.* (2010) explican los desafíos que surgen al evaluar la precisión de la predicción genómica a través de la validación cruzada aleatoria, así como también seleccionar la mejor opción de hiperparámetros en los modelos bayesianos.

Los modelos mixtos lineales juegan un papel fundamental en la selección genómica y en la predicción genómica. Este tipo de modelos se utiliza ampliamente para las predicciones, aunque otros modelos tales como los no lineales, redes neuronales y

modelos de aprendizaje automático, podrían utilizarse para este propósito. El modelo lineal mixto es de la forma $\boldsymbol{y} = \boldsymbol{X\beta} + \boldsymbol{Zu} + \boldsymbol{e}$, donde $\boldsymbol{y}$ es el vector de respuesta de dimensión $n \times 1$, $\boldsymbol{X}$ y $\boldsymbol{Z}$ son las matrices diseños para los efectos fijos ($\boldsymbol{\beta}$) y los efectos aleatorios para los genotipos ($\boldsymbol{u}$), respectivamente, $\boldsymbol{u} \sim NM(\boldsymbol{0}, \sigma_u^2\boldsymbol{K})$ con $\boldsymbol{K}$ una matriz semidefinida de varianza-covarianza conocida y $\boldsymbol{e} \sim NM(\boldsymbol{0}, \sigma_e^2\boldsymbol{I})$. En el contexto de la selección genómica, $\boldsymbol{K}$ puede ser la matriz de relaciones aditivas derivada de los coeficientes de co-ancestro (matriz de relaciones numéricas $\boldsymbol{A}$) o puede ser la matriz de relación genómica obtenidas de los marcadores ($\boldsymbol{G}$). Como se muestra a continuación, existen diversas formas de expresar la matriz de incidencia $\boldsymbol{Z}$ y del vector de efectos aleatorios $\boldsymbol{u}$ cuando se utiliza la matriz de relación numérica ($\boldsymbol{A}$). Las versiones bayesianas de los modelos de regresión lineal se han desarrollado ampliamente y el software para su ajuste se ha distribuido y utilizado en gran medida para la investigación, y se han extendido a casos más complicados como, por ejemplo, la introducción de la interacción genotipo × ambiente que incorpora covariables de pedigrí y ambientales (Jarquín *et al.*, 2014).

Endelman (2011) desarrolló un paquete para R denominado `rrBLUP`, que es capaz de ajustar el modelo lineal mixto básico con dos componentes de varianza (σ_u^2 y σ_e^2) descrito anteriormente, con los métodos de máxima verosimilitud o REML. Como una facilidad extra, el `rrBLUP` calcula el kernel Gaussiano y el kernel Exponencial que, generalmente explican los pequeños efectos epistáticos crípticos entre los marcadores. El `rrBLUP` incluye un algoritmo de validación cruzada para medir la precisión de la predicción de los modelos y muestra soluciones rápidas de las ecuaciones del modelo mixto para tamaño de datos moderados e intermedios. Algunos softwares más especializados se desarrollaron posteriormente como el `synbreed` de Wimmer *et al.* (2012) y `GEMMA` (Zhou y Stephens, 2012).

A pesar de que los softwares mencionados resuelven problemas de predicciones genómicas (por ejemplo, predicción en conjuntos de entrenamiento y prueba, validación cruzada y estimación de parámetros de varianza), son piezas de software separadas sin un marco estadístico e informático unificado. Por lo tanto, desde la perspectiva del usuario, tener un solo paquete que implemente todos los modelos que se instalarán ahorrará tiempo de preparación y de análisis de datos. Pérez y de los Campos (2014) extendieron el paquete `BLR` original desarrollado por Pérez *et al.* (2010) a un marco más general, la Regresión Lineal Generalizada Bayesiana (BGLR) el cual ofrece a los usuarios diversas alternativas para el análisis de datos en un software unificado. El

paquete `BGLR` está disponible en CRAN (`https://cran.r-project.org`). El paquete `BGLR` incluye varios modelos de regresión bayesiana, incluyendo métodos paramétricos para selección de variables y métodos de encogimiento, y procedimientos semi-paramétricos (Bayesian reproducing kernel Hilbert spaces regressions, RKHS). También se implementan diversas rutinas que pueden ser aplicadas en problemas distintos a la predicción genómica, además de esto las variables respuesta pueden ser continuas o categóricas (binario u ordinal). Los modelos son ajustados utilizando métodos de cadenas de Markov Monte Carlo, principalmente el muestreador de Gibbs y el algoritmo de Metropolis implementados en los lenguajes de programación R (R Core Team, 2021) y C (Kernighan y Ritchie, 1988). Además, el paquete `BGLR` puede ajustar modelos más complejos como por ejemplo el modelo que incluyen la interacción genotipo × ambiente, pedigrí, covariables ambientales, etc. (Jarquín *et al.*, 2014). El paquete `BGLR` también se utiliza para evaluar el efecto marcador × ambiente (Lopez-Cruz *et al.*, 2015), ajuste de modelos para selección de variables como el BayesB o BayesC (Crossa *et al.*, 2017), o para ajustar modelos de alta dimension con variable respuesta ordinal (Montesinos-López *et al.*, 2016).

Aunque el modelo lineal mixto es una herramienta importante para ajustar modelos de selección genómica, Covarrubias-Pazaran (2016) menciona que el software actual para realizar predicción genómica incluye solo un único componente de varianza además del asociado al error, y por lo tanto, la predicción para situaciones más complicadas como la predicción de rendimiento de híbridos con efectos aditivos, dominancia y epistáticos no es tarea sencilla utilizando los modelos disponibles. Covarrubias-Pazaran (2016) desarrolló un software que ajusta modelos mixtos con múltiples efectos aleatorios denominado `sommer`. El paquete `sommer` estima los parámetros de varianza utilizando los algoritmos siguientes: (i) Información Promedio (Average Information en inglés), (ii) Esperanza-Maximización (Expectation-Maximization en inglés) y (iii) asociación eficiente en el modelo mixto (EMMA). El software `sommer` es competitivo respecto a otros programas de cómputo y ajusta los modelos de forma más rápida que los programas de cómputo que ajustan los modelos usando algoritmos basados en cadenas de Markov Monte Carlo.

El desarrollo de software para el ajuste del modelo lineal mixto es un área activa de investigación ya que el uso de este modelo para la predicción genómica basada en marcadores moleculares e información del pedigrí es crucial para el avance en la aplicación del mejoramiento genético. El paquete `lme4` (Bates *et al.*, 2014) para R

incluye funciones eficientes para ajustar el modelo lineal mixto y mixto generalizado (GLMMs, por sus siglas en inglés). Algunas de las características principales del paquete `lme4` son las siguientes: (i) es eficiente para manejo de conjuntos de datos grandes; (ii) manejo de número arbitrario de factores de agrupación, anidados, cruzados, etc., y (iii) puede usar una combinación de representaciones matriciales densas y ralas para facilitar el procesamiento de datos de alta dimensión de forma rápida.

Sin embargo, el uso del paquete `lme4` para el análisis genético ha sido limitado ya que no permite usar la correlación entre individuos o grupos de individuos. Cuando los individuos están relacionados a nivel genético, es necesario considerarla al momento de construir la función de verosimilitud marginal, tal información se puede incluir calculando covarianzas entre los individuos relacionados. Vazquez *et al.* (2010) desarrollaron un paquete para R denominado `pedigreemm` que utiliza los algoritmos de ajuste de modelos implementados en `lme4`, pero este permite incluir correlaciones entre niveles de efectos aleatorios tales como las relaciones genéticas entre parientes expresadas como relaciones de pedigrí. La metodología de Vazquez *et al.* (2010) consiste en re-expresar la matriz $\boldsymbol{Z}$ en el modelo mixto lineal en función de la descomposición de Cholesky ($\boldsymbol{L}$) de la matriz de relaciones genómicas aditivas $\boldsymbol{A}$ obtenida a partir de la información del pedigrí.

Dadas las consideraciones anteriores y algunas limitaciones en términos de la eficiencia computacional de algunos modelos de predicción genómica existentes, describimos el nuevo paquete de R, `lme4GS`, el cual está basado en el software `lme4` de Bates *et al.* (2014) que se encuentra disponible en CRAN (`https://cran.r-project.org`). El paquete `lme4GS` se centra en la predicción basada en marcadores con énfasis en el problema de selección genómica y puede ajustar modelos mixtos con diferentes matrices de varianza-covarianza. El paquete `lme4GS` incluye efectos fijos y aleatorios, y matrices de varianza-covarianza asociadas, de donde se obtienen las matrices para efectos fijos y aleatorios ($\boldsymbol{X}, \boldsymbol{Z}_1, ..., \boldsymbol{Z}_q$, respectivamente). Las matrices originales de varianza-covarianza se introducen y se transforman mediante la factorización de Cholesky o la descomposición espectral y luego se utilizan para definir la función objetivo (función de deviance). Una vez definida la función objetivo, el módulo de optimización maximiza la función objetivo y proporciona estimaciones REML de los parámetros de interés.

3.2 Materiales y Métodos

Considere el modelo lineal mixto:

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{u} + \boldsymbol{e}, \tag{3.1}$$

donde $\mathbf{y}$ es el vector de respuesta de dimensión $n \times 1$, $\boldsymbol{X}$ es la matriz de efectos fijos de dimensión $n \times p$, $\boldsymbol{\beta}$ es el vector de efectos fijos de dimensión $p \times 1$, $\boldsymbol{Z}$ es la matriz de incidencia de dimensión $n \times r$, y $\boldsymbol{u}$ es el vector de efectos aleatorios. Se supone que $\boldsymbol{u} \sim NM\left(\mathbf{0}, \sigma_a^2 \boldsymbol{K}\right)$ y $\boldsymbol{e} \sim NM\left(\mathbf{0}, \sigma_e^2 \boldsymbol{I}\right)$, con $\boldsymbol{K}$ conocida como la matriz de varianza-covarianza, σ_a^2 y σ_e^2 son los parámetros de varianza asociadas con $\boldsymbol{u}$ y $\boldsymbol{e}$, respectivamente. Además se supone que $\boldsymbol{u}$ y $\boldsymbol{e}$ se distribuyen independientemente. En el caso de la selección genómica, la matriz de varianza-covarianza se pueden derivar de los marcadores o del pedigrí.

El modelo lineal mixto (3.1) se puede re-escribir como:

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}^*\boldsymbol{u}^* + \boldsymbol{e}, \tag{3.2}$$

donde $\boldsymbol{Z}^* = \boldsymbol{Z}\boldsymbol{L}$, con $\boldsymbol{L}$ obtenido de la factorización de Cholesky de $\boldsymbol{K}$, alternativamente, $\boldsymbol{Z}^* = \boldsymbol{Z}\boldsymbol{\Gamma}\boldsymbol{\Lambda}^{1/2}$ con $\boldsymbol{\Gamma}$ y $\boldsymbol{\Lambda}$ las matrices de eigen-vectores y eigen-valores obtenidas de la descomposición espectral de $\boldsymbol{K}$, y $\boldsymbol{u}^* \sim NM\left(\mathbf{0}, \sigma_u^2 \boldsymbol{I}\right)$. Tenemos que $\boldsymbol{Z}^*\boldsymbol{u}^*$ tiene la misma distribución que $\boldsymbol{Z}\boldsymbol{u}$, esto es $\boldsymbol{Z}^*\boldsymbol{u}^* \overset{d}{=} \boldsymbol{Z}\boldsymbol{u} \sim NM(\mathbf{0}, \sigma_a^2 \boldsymbol{Z}\boldsymbol{K}\boldsymbol{Z}^t)$.

3.2.1 BLUPs

Una vez que el modelo mixto en (3.2) se ajusta, pueden obtenerse las medias condicionales de los efectos aleatorios, esto es, $\widehat{\boldsymbol{u}}^*$. El BLUP para $\boldsymbol{u}^*$ se obtiene como sigue: $\widehat{\boldsymbol{u}}^* = \widehat{\sigma}_u^2 \boldsymbol{Z}^{*t} \widehat{\boldsymbol{V}}^{*-1} \left(\boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}\right)$ donde $\widehat{\boldsymbol{V}}^* = \widehat{\sigma}_u^2 \boldsymbol{Z}^* \boldsymbol{Z}^{*t} + \hat{\sigma}_e^2 \boldsymbol{I}$, con $\hat{\sigma}_e^2$, $\hat{\sigma}_u^2$ y $\widehat{\boldsymbol{\beta}}$ son estimaciones REML de los parámetros de varianza y del vector de efectos fijos, respectivamente. Las medias condicional de los efectos aleatorios para el modelo en la ecuación (3.1) se obtiene como sigue: $\widehat{\boldsymbol{u}} = \boldsymbol{L}\widehat{\boldsymbol{u}}^*$ si se utiliza la factorización de Cholesky, alternativamente, $\widehat{\boldsymbol{u}} = \boldsymbol{\Gamma}\boldsymbol{\Lambda}^{1/2}\widehat{\boldsymbol{u}}^*$ si se utiliza la descomposición espectral.

3.2.2 Predicción de nuevas observaciones

El objetivo principal de la selección genómica es predecir nuevas observaciones (valores fenotípicos) o simplemente obtener los BLUPs para los efectos aleatorios no presentes en los datos observados, pero que se muestrean de la misma población de $\boldsymbol{u}$ y $\boldsymbol{e}$ (Gilmour *et al.*, 2004). Suponiendo que el vector aleatorio $\boldsymbol{u}$ y la matriz $\boldsymbol{K}$ se particiona de la siguiente manera:

$$\boldsymbol{u} = \begin{bmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{bmatrix}, \ \boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_{11} & \boldsymbol{K}_{12} \\ \boldsymbol{K}_{21} & \boldsymbol{K}_{22} \end{bmatrix},$$

entonces el BLUP para $\boldsymbol{u}_2$ se obtiene como:

$$E\left(\boldsymbol{u}_2|\boldsymbol{y}_1\right) = \boldsymbol{K}_{21}\boldsymbol{K}_{11}^{-1}\boldsymbol{u}_1. \tag{3.3}$$

En el caso más general, el modelo (3.1) se puede extender para incluir más efectos aleatorios, esto es:

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \sum_{j=1}^{q} \boldsymbol{Z}_j\boldsymbol{u}_j + \boldsymbol{e}, \tag{3.4}$$

donde $\boldsymbol{Z}_j$ es la matriz diseño de los efectos aleatorios, y $\boldsymbol{u}_j$ es el vector de los efectos aleatorios, $j = 1, ..., q$, donde q corresponde al número de términos aleatorios incluidos en el modelo. Se supone que $\boldsymbol{u}_j \sim NM\left(\boldsymbol{0}, \sigma_j^2\boldsymbol{K}_j\right)$ se distribuyen independientemente. Note que el modelo (3.1) es una caso especial del modelo (3.4) obtenido al establecer $q = 1$, $\boldsymbol{Z} = \boldsymbol{Z}_1$, $\boldsymbol{u} = \boldsymbol{u}_1$, $\boldsymbol{K} = \boldsymbol{K}_1$, $\sigma_a^2 = \sigma_1^2$. Basado en la misma estrategia computacional utilizada para reescribir el modelo (3.1) como el modelo en (3.2), el modelo (3.4) se puede reescribir como:

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \sum_{j=1}^{q} \boldsymbol{Z}_j^*u_j^* + \boldsymbol{e}. \tag{3.5}$$

3.2.3 Implementación

El paquete `lme4GS` es una extensión del paquete `lme4` de R (Bates *et al.*, 2014). El desarrollo del paquete `lme4GS` se inspiró en los paquetes R existentes: 1) `pedigreem` (Vazquez *et al.*, 2010) y 2) `lme4qtl` (Ziyatdinov *et al.*, 2018), el cual se centran en estudios de asociación QTL y mapas de ligamiento, mientras que `lme4GS` se centra en el problema de predicción en selección genómica (Meuwissen *et al.*, 2001) con modelos tipo GBLUP, aunque los modelos pueden aplicarse en otras áreas de investigación. El software `lme4GS` utiliza el motor computacional proporcionado por el paquete `lme4` el cual ha sido ampliamente probado y utiliza las rutinas computacionales para ajustar modelos mixtos con matrices de varianza-covarianza proporcionadas por el usuario. El paquete `lme4GS` puede considerarse como una generalización del paquete `rrBLUP` (Endelman, 2011), ya que es capaz de ajustar el modelo (3.4), mientras que `rrBLUP` ajusta el modelo (3.1). El paquete también implementa algunos de los modelos incluidos en el software `sommer` (Covarrubias-Pazaran, 2016). El software `lme4GS` hace uso del diseño modular de la función `lmer` (módulo de fórmulas, módulo de función objetivo, módulo de optimización y módulo salida) para ajustar los modelos con matrices de varianza-covarianza dadas por el usuario. El módulo fórmula nos permite especificar los efectos fijos y aleatorios, y las matrices de varianza-covarianza asociadas, de donde se obtienen las matrices para los efectos fijos y aleatorios ($\boldsymbol{X}, \boldsymbol{Z}_1, ..., \boldsymbol{Z}_q$, respectivamente). Después de esto, las matrices de varianza-covarianza son utilizadas en el cálculo de las las matrices de incidencias transformadas ($\boldsymbol{Z}_j^*, j = 1, ..., q$) utilizando la descomposición de Cholesky o descomposición espectral de las matrices de varianza-covarianza proporcionadas por el usuario, las cuales se toman como entrada para definir la función objetivo (deviance function). Una vez que la función objetivo ha sido definida, el módulo de optimización se utiliza para optimizar la función objetivo y proporcionar las estimaciones REML de los parámetros de interés. Finalmente, el módulo de salida proporciona los resultados que serán interpretadas por el usario. Se desarrollaron tres funciones principales en R:

- `lmerUvcov`: Ajusta el modelo lineal mixto con matriz de varianza-covarianza definida por el usuario. Esta función toma como entrada una fórmula para especificar la respuesta `y`, los efectos fijos (fixed) y los efectos aleatorios (random), una tabla de datos (`data.frame`) y una lista (`Uvcov`) para especificar las matrices de varianza-covarianza de los efectos aleatorios. Una vez que el modelo se ajusta, la rutina regresa un objeto de la clase `merMod` para los cual hay muchos métodos

disponibles en R para su posterior procesamiento (ejemplos `summary`, `print`, `predict`, `VarCorr`, etc.).

- `ranefUvcov`: Extrae las medias condicionales de los efectos aleatorios. Esta función toma como entrada un objeto regresado por la función `lmerUvcov`. Las medias condicionales para los efectos aleatorios en el modelo (3.4) se obtienen como se explicó en la sección 3.2.1. La función `ranef` en `lme4` está sobreescrita con `ranefUvcov` de tal forma que el usuario pueda llamar a cualquiera de estas dos rutinas y obtener los mismos resultados.

- `ranefUvcovNew`: Obtiene los BLUPs para nuevos niveles de efectos aleatorios con matrices de varianza-covarianza especificados. La función toma como entrada un objeto proporcionado por la función `lmerUvcov` y una lista de dos niveles con matrices de varianza-covarianza que contiene información de los identificadores de los individuos que se van a predecir y los que se incluyeron al ajustar el modelo. Los BLUPs se obtienen utilizando particiones similares a las usadas para derivar la ecuación (3.4).

El software se encuentra disponible en el repositorio de github, https://github.com/perpdgo/lme4GS.

3.3 Ejemplos

En esta sección se ilustra el uso del paquete `lme4GS` con diversos ejemplos usando datos de muestra incluidos en el paquete. En los ejemplos se considera solamente la predicción de los efectos aleatorios y la estimación de los parámetros de varianza, aunque el software también puede estimar efectos fijos.

3.3.1 Ejemplo 1: Predicción utilizando marcadores y pedigrí

En este ejemplo se analiza el conjunto de datos de 599 líneas de trigo desarrollado por el programa global del mejoramiento del trigo del CIMMyT. El conjunto de datos se ha analizado varias veces en la literatura (ej. Crossa *et al.*, 2010; de los Campos *et al.*, 2009; Pérez *et al.*, 2010). El conjunto de datos incluye información del rendimiento de grano, del

pedigrí y de 1477 marcadores generados por Triticarte Pty., Ltd. (Canberra, Australia; https://www.diversityarrays.com). Los datos fenotípicos a analizar corresponden a los datos originales e incluyen las réplicas en cada ambiente y también la información del pedigrí con la finalidad de mostrar la forma de obtener la matriz de relaciones genéticas aditiva utilizando otras herramientas de R y utilizar esa matriz para el ajuste de los modelos. El conjunto de datos puede cargarse dentro del ambiente de R con los comandos mostrados en el listado 3.1.

Listado de código 3.1: Cargando datos de trigo.

```
1 library(lme4GS)
2 data(wheat599)
3 ls() #list objects
```

Una vez que los comandos se ejecutan, los siguientes objetos se cargan a memoria:

- `wheat.Pheno`: Tabla de datos (`data.frame`) con 4 columnas, `Env` para los ambientes, `Rep` para las réplicas, `GID` para identificadores de los genotipos, y `Yield` para el rendimiento en grano.
- `wheat.Pedigree`: Tabla de datos (`data.frame`) con 3 columnas: `gpid1` y `gpid2`, el cual corresponden para el GID de los padres 1 y 2, respectivamente, y la `progenie`, que corresponden los GIDs de la progenie.
- `wheat.X`: Una matriz de dimensión 599×1279, el cual corresponde a los marcadores DArT codificados como 0 y 1.

Un modelo mixto lineal que permite predecir el rendimiento de grano en uno de los ambientes utilizando los marcadores y el pedigrí está dado por:

$$\boldsymbol{y} = \mathbf{1}\mu + \boldsymbol{Z}_1\boldsymbol{u}_1 + \boldsymbol{Z}_2\boldsymbol{u}_2 + \boldsymbol{e}, \tag{3.6}$$

donde $\boldsymbol{y}$ es el vector de fenotipos observados en un ambiente en particular, $\mathbf{1}$ es un vector de unos, μ es el intercepto, $\boldsymbol{u}_1 \sim NM\left(\mathbf{0}, \sigma^2_m \boldsymbol{G}\right)$, $\boldsymbol{G} = \dfrac{\boldsymbol{W}\boldsymbol{W}^t}{p}$ (véase Lopez-Cruz *et al.*, 2015) es la matriz de relaciones genómicas, $\boldsymbol{W}$ es la matriz de marcadores centrados y estandarizados, p es el número de marcadores, σ^2_m es el parámetro de varianza asociado

con los marcadores; $\boldsymbol{u}_2 \sim NM\left(\mathbf{0}, \sigma_a^2 \boldsymbol{A}\right)$, $\boldsymbol{A}$ es una matriz de relaciones genéticas aditiva derivada del pedigrí, y σ_a^2 es el parámetro de varianza asociado a $\boldsymbol{A}$, $\boldsymbol{Z}_1$ y $\boldsymbol{Z}_2$ son matrices que conectan los fenotipos con los genotipos, y $\boldsymbol{e}$ se distribuye como se describió en el modelo (3.1). La matriz de relaciones genéticas aditivas $\boldsymbol{A}$ pueden calcularse fácilmente en R utilizando el paquete `pedigreemm` (Vazquez *et al.*, 2010), la descomposición de Cholesky correspondiente se puede calcular de manera muy eficiente y el paquete puede almacenar el resultado como una matriz rala. El código en el listado 3.2 calcula las matrices $\boldsymbol{A}$ y $\boldsymbol{G}$ y después se ajusta el modelo mixto utilizando la función `lmerUvcov`. Después de esto, se obtienen los BLUPs utilizando la función `ranefUvcov`.

Listado de código 3.2: Calcular las matrices $\boldsymbol{A}$ y $\boldsymbol{G}$.

```
## Complete and sort incomplete Pedigree using editPed
PedEdit<-editPed(sire=wheat.Pedigree$gpid1,dam=wheat.Pedigree$gpid2,
                 label=wheat.Pedigree$progenie, verbose=TRUE)

## Converted the data frame PedEdit into an S4 object of formal
## class 'Pedigree'
PedFinal<-with(PedEdit,pedigree(label=label,sire=sire,dam=dam))

#A
AFull<-getA(PedFinal)
GID<-unique(wheat.Pheno$GID)
selected<-rownames(AFull)%in%GID
A<-AFull[selected,selected]
A<-matrix(A,599,599)
rownames(A)<-colnames(A)<-rownames(AFull[selected,selected])

W<-scale(wheat.X,center=TRUE,scale=TRUE)
G<-tcrossprod(W)/ncol(W)

#Environment 1
e1<-which(wheat.Pheno$Env==1)
y<-wheat.Pheno[e1,]$Yield
GID<-as.character(wheat.Pheno[e1,]$GID)

wheat<-data.frame(y=y,mrk=GID,ped=GID)
random<-list(mrk=list(K=G),ped=list(K=A))

fmGA<-lmerUvcov(y~(1|mrk)+(1|ped),data = wheat,Uvcov = random)
summary(fmGA)

#BLUPs
```

```
ranefUvcov(fmGA)

#or equivalently
ranef(fmGA)
```

El proceso para ajustar el modelo tomó alrededor de 81 segundos en una computadora con un procesador Intel Core i7 @ 2.8 GHz. Después de ajustar el modelo, la función `summary` puede utilizarse para mostrar algunos de los resultados. Las estimaciones de los parámetros de varianza fueron: $\widehat{\sigma}^2_m = 0.2218$, $\widehat{\sigma}^2_a = 0.2213$ y $\widehat{\sigma}^2_e = 0.0349$ (ver listado 3.3). Las funciones `predict`, `residuals`, etc., que se utilizan habitualmente después de ajustar el modelo con la función `lmer` también se puede utilizar con el objeto resultante.

Listado de código 3.3: Resultados parciales del listado 3.2.

```
Linear mixed model fit by REML ['lmerUvcov']
Formula: y ~ (1 | mrk) + (1 | ped)
   Data: wheat

REML criterion at convergence: 1103.5

Scaled residuals:
     Min       1Q   Median       3Q      Max
-2.51852 -0.44430  0.00982  0.42390  2.60030

Random effects:
 Groups   Name        Variance Std.Dev.
 mrk      (Intercept) 0.22189  0.4711
 ped      (Intercept) 0.21138  0.4598
 Residual             0.03496  0.1870
Number of obs: 1198, groups:  mrk, 599; ped, 599

Fixed effects:
            Estimate Std. Error t value
(Intercept)  4.81719    0.08757   55.01
```

3.3.2 Ejemplo 2: Conjuntos de entrenamiento y prueba

En este ejemplo se simula el problema de selección genómica que enfrentan los mejoradores; se evalúa el poder predictivo del modelo (3.6) particionando los datos de forma aleatoria en dos conjuntos disjuntos, se asignó el 80 % de las líneas al conjunto de

entrenamiento y el restante 20 % al conjunto de prueba. El código en el listado 3.4 particiona los datos en el conjunto de entrenamiento y prueba, y define dos vectores, `y_tnr` y `y_tst`, con los valores fenotípicos para ambos conjuntos. Después se crea una lista para especificar los efectos aleatorios que se van a incluir en el modelo lineal mixto. El modelo lineal mixto se ajusta utilizando la función `lmerUvcov`. En el siguiente paso, se define una lista de los efectos aleatorios incluyendo las matrices de varianzas-covarianzas $\boldsymbol{G}$ y $\boldsymbol{A}$, y los Identificadores de los Genotipos (GIDs por sus siglas en inglés) de las líneas a predecir; los nombres de las hileras y columnas de las matrices de varianzas-covarianzas corresponden a los GIDs. La función `ranefUvcovNew` se utiliza para predicción, y proporciona una lista de los BLUPs para cada uno de los términos aleatorios como resultado. Finalmente, las predicciones para los individuos en el conjunto de prueba se obtienen simplemente sumando el intercepto a los BLUPs. Los valores observados y predichos se guardan en una tabla con tres columnas: `GID`, `y` (valores fenotípicos observados), e `yHat` (valores fenotípicos predichos) mismos que pueden ser utilizados para generar algunas gráficas. La Figura 3.1 muestra un diagrama de dispersión con los valores fenotípicos observados vs predichos en los conjuntos de entrenamiento y prueba. El coeficiente de correlación de Pearson entre los valores observados y predichos fue 0.5638, y el Error Cuadrado Medio (MSE) 0.2581.

Listado de código 3.4: Partición única de entrenamiento y prueba.

```
set.seed(456)
trn<-sample(unique(GID),size=as.integer(0.80*599))
tst<-setdiff(unique(GID),trn)

#Phenotypes in training and testing
y_trn<-y[GID%in%trn]
y_tst<-y[GID%in%tst]

A_trn<-A[rownames(A)%in%trn,colnames(A)%in%trn]
G_trn<-G[rownames(G)%in%trn,colnames(G)%in%trn]
GID_trn<-GID[GID%in%trn]
GID_tst<-GID[!(GID%in%trn)]

pheno_trn<-data.frame(y_trn=y_trn,mrk=GID_trn,
                      ped=GID_trn)

random<-list(mrk=list(K=G_trn),ped=list(K=A_trn))

fmGA_trn<-lmerUvcov(y_trn~(1|mrk)+(1|ped), data = pheno_trn,
                    Uvcov = random)
```

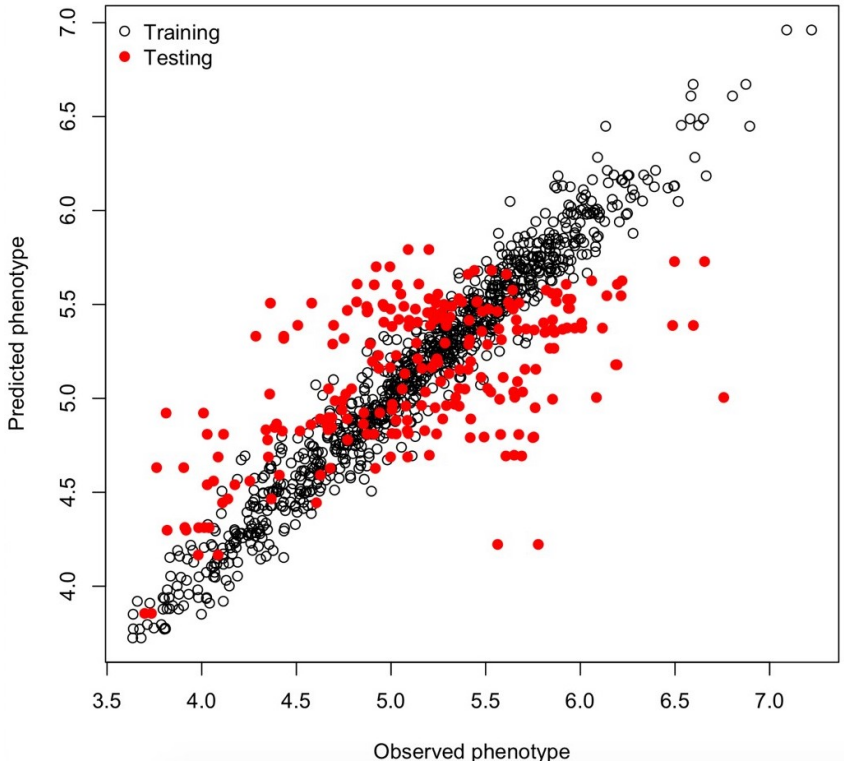

Figura 3.1: Diagrama de dispersión entre los valores observados y predichos en los conjuntos de entrenamiento y prueba.

```
21
22 plot(pheno_trn$y_trn, predict(fmGA_trn),
23      xlab="Observed phenotype",ylab="Predicted phenotype")
24
25 #Predict for new levels
26 blup_tst<-ranefUvcovNew(fmGA_trn,
27                         Uvcov=list(mrk=list(K=G),ped=list(K=A)))
28 i1<-match(GID_tst,rownames(blup_tst$mrk))
29 i2<-match(GID_tst,rownames(blup_tst$ped))
30 blup_mrk<-blup_tst$mrk[i1,1]
31 blup_ped<-blup_tst$ped[i2,1]
32 yHat_tst<-fixef(fmGA_trn)[1] + blup_mrk + blup_ped
33
34 points(y_tst,yHat_tst,col="red",pch=19)
35 legend("topleft",legend=c("Training","Testing"),
36        pch=c(1,19),col=c("black","red"),bty="n")
37
38 #Correlation in testing set
39 cor(y_tst,yHat_tst)
```

```
40
41 #MSE
42 var(y_tst-yHat_tst)
43
44 #Data frame with prediction for further processing
45 predictions<-data.frame(GID=GID_tst,y=y_tst,yHat=yHat_tst)
```

En el listado de código 3.5 se muestran las instrucciones de R para realizar una validación cruzada con 5 grupos (folds) que se usa ampliamente para estudiar la precisión de la predicción (ej. Crossa *et al.*, 2010). Se dividen aleatoriamente los datos en 5 conjuntos disjuntos basados en el GID, $\{S_1, \ldots, S_5\}$. Cada conjunto se utiliza para medir la precisión de la predicción. Utilizando estos conjuntos, los datos se dividen en las poblaciones de entrenamiento y prueba, por ejemplo, los datos en $\{S_2., \ldots, S_5\}$ son los datos de entrenamiento y S_1 son los datos de prueba. El modelo se ajusta utilizando los datos de entrenamiento, y los fenotipos para S_1 se predicen y se obtiene la precisión de la predicción. El mismo ejercicio se puede realizar tomando S_f como los datos de prueba, $f = 2, \ldots, 5$. En el cuadro 3.1 se muestra los resultados de la validación cruzada, la columna 1 corresponde al campo, la columna 2 muestra el coeficiente de correlación de Pearson entre los valores observados y predichos para los individuos del conjunto de entrenamiento, la columna 3 corresponde al MSE del conjunto de entrenamiento, las columnas 4 y 5 muestran las correlaciones y el MSE para los individuos del conjunto de prueba. La correlación promedio del conjunto de entrenamiento fue de 0.9768 mientras que la correlación del conjunto de prueba fue de 0.5192. La media del MSE en el conjunto de entrenamiento fue de 0.0187 y del conjunto de prueba fue de 0.2897. Los resultados son los esperados: la correlación en el conjunto de entrenamiento es más alto que en el conjunto de prueba y el MSE es más alto en conjunto de prueba que en el conjunto de entrenamiento.

Listado de código 3.5: Validación cruzada.

```
1 set.seed(789)
2 uGID<-unique(GID)
3 nFolds<-5
4 sets<-sample(1:nFolds,size=length(uGID),replace=TRUE)
5 resultsCV<-matrix(NA,nrow=nFolds,ncol=4)
6 colnames(resultsCV)= c("r_trn","MSE_trn","r_tst","MSE_tst")
7
8 for(f in 1:nFolds)
9 {
10   #Training and testing
```

```
trn<-(uGID[sets!=f])
tst<-(uGID[sets==f])

#Phenotypes in training and testing
y_trn<-y[GID%in%trn]
y_tst<-y[GID%in%tst]

A_trn<-A[rownames(A)%in%trn,colnames(A)%in%trn]
G_trn<-G[rownames(G)%in%trn,colnames(G)%in%trn]
GID_trn<-GID[GID%in%trn]
GID_tst<-GID[!(GID%in%trn)]

pheno_trn<-data.frame(y_trn=y_trn,mrk=GID_trn,
          ped=GID_trn)

random<-list(mrk=list(K=G_trn),ped=list(K=A_trn))

fmGA_trn<-lmerUvcov(y_trn~(1|mrk)+(1|ped), data = pheno_trn,
Uvcov = random)

yHat_trn<-predict(fmGA_trn)

#Correlation in training set
resultsCV[f,1]<-cor(y_trn,yHat_trn)

#MSE in training set
resultsCV[f,2]<-var(y_trn-yHat_trn)


#Predict for new levels
blup_tst<-ranefUvcovNew(fmGA_trn,Uvcov=list(mrk=list(K=G),
          ped=list(K=A)))
i1<-match(GID_tst,rownames(blup_tst$mrk))
i2<-match(GID_tst,rownames(blup_tst$ped))
blup_mrk<-blup_tst$mrk[i1,1]
blup_ped<-blup_tst$ped[i2,1]
yHat_tst<-fixef(fmGA_trn)[1] + blup_mrk + blup_ped

#Correlation in testing set
resultsCV[f,3]<-cor(y_tst,yHat_tst)

#MSE
resultsCV[f,4]<-var(y_tst-yHat_tst)
```

Cuadro 3.1: Resultados de una validación cruzada de 5 conjuntos.

Conjunto	Entrenamiento		Prueba	
	r	MSE	r	MSE
1	0.9752	0.0201	0.5290	0.2778
2	0.9775	0.0181	0.5680	0.2729
3	0.9755	0.0197	0.5096	0.3035
4	0.9786	0.0173	0.4179	0.3280
5	0.9775	0.0182	0.5714	0.2663
avg	0.9769	0.0187	0.5192	0.2897
sd	0.0015	0.0012	0.0624	0.0256

```
54 }
55
56 resultsCV
```

3.3.3 Ejemplo 3: Predicción de rendimiento de híbridos

La predicción del rendimiento híbrido es una tarea importante en los programas de mejoramiento agrícola. Technow *et al.* (2014) y Acosta-Pech *et al.* (2017) emplearon los modelos tipo G-BLUP para predecir el rendimiento de híbridos de maíz. El modelo lineal utilizando para este propósito está dado por:

$$\boldsymbol{y} = \mathbf{1}\mu + \boldsymbol{W}\boldsymbol{\theta} + \boldsymbol{Z}_1\boldsymbol{u}_1 + \boldsymbol{Z}_2\boldsymbol{u}_2 + \boldsymbol{Z}_3\boldsymbol{u}_3 + \boldsymbol{e}, \tag{3.7}$$

donde $\boldsymbol{y}$ es el vector de respuesta, $\mathbf{1}$ es el vector de unos, μ es el intercepto, $\boldsymbol{W}$ es la matriz diseño para los ambientes, $\boldsymbol{\theta}$ es el vector de efectos ambientales (fijos), $\boldsymbol{Z}_1$, $\boldsymbol{Z}_2$, $\boldsymbol{Z}_3$ son las matrices de incidencia de los padres, madres e híbridos, respectivamente; $\boldsymbol{u}_1$, $\boldsymbol{u}_2$ son vectores de la habilidad combinatoria general para líneas paternas y maternas, respectivamente, $\boldsymbol{u}_1 \sim NM\left(\mathbf{0}, \sigma_1^2\boldsymbol{K}_1\right)$, $\boldsymbol{u}_2 \sim NM\left(\mathbf{0}, \sigma_2^2\boldsymbol{K}_2\right)$ con $\boldsymbol{K}_1$ y $\boldsymbol{K}_2$ matrices de relaciones para las líneas paternas y maternas asociadas con los parámetros de varianzas σ_1^2, σ_2^2; $\boldsymbol{u}_3 \sim NM\left(\mathbf{0}, \sigma_3^2\boldsymbol{K}_3\right)$ con $\boldsymbol{K}_3 = \boldsymbol{K}_1 \otimes \boldsymbol{K}_2$, σ_3^2 el parámetro de varianza asociada con la habilidad combinatoria específica de los híbridos, $\otimes$ denota el producto Kronecker de dos matrices, y $\boldsymbol{e} \sim NM\left(\mathbf{0}, \sigma_e^2\boldsymbol{I}\right)$. Note que el modelo (3.7) puede reescribirse como $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}_1\boldsymbol{u}_1 + \boldsymbol{Z}_2\boldsymbol{u}_2 + \boldsymbol{Z}_3\boldsymbol{u}_3 + \boldsymbol{e}$, donde $\boldsymbol{X} = [\mathbf{1}\ \boldsymbol{W}]$ y $\boldsymbol{\beta} = \left(\mu, \boldsymbol{\theta}^t\right)^t$, el cual corresponde

al modelo (3.5) discutido anteriormente. Para ejemplificar el ajuste de este modelo en el paquete `lme4GS`, se utiliza el conjunto de datos `DT_cornHybrids` incluidos en el paquete `sommer` de R (Covarrubias-Pazaran, 2016). El paquete `lme4GS` incluye una copia de los datos originales y se denominan `cornHybrids`. El conjunto de datos contiene información del rendimiento en grano y altura de la planta para 100 de 400 posibles cruzas originadas a partir de 40 líneas endogámicas pertenecientes a dos grupos heteróticos, 20 líneas en cada grupo. Solamente se evaluaron 100 híbridos en cuatro localidades, entonces el problema consiste en estimar las habilidades combinatorias generales, habilidades combinatorias específicas y la predicción del rendimiento de los híbridos no probados en cada localidad. El conjunto de datos se puede cargar en R usando los comandos que se muestran en listado 3.6.

Listado de código 3.6: Cargando datos de maíz.

```
1 library(lme4GS)
2 data(cornHybrids)
3 ls() #list objects
```

Una vez cargados los datos, los objetos siguientes estarán disponibles en memoria:

- `maize.Pheno`: Tabla de datos con 6 columnas, `Location` (localidad), `GCA1` (padre 1), `GCA2` (padre 2), `SCA` (híbrido), `Yield` (rendimiento) y `PlantHeight` (altura de planta). Los registros con valores perdidos en las dos últimas columnas corresponden a híbridos (identificado con la etiqueta `Parent 1: Parent 2`) que no fueron evaluados en el campo y que es necesario predecir.

- `maize.G`: Una matriz con relaciones entre individuos para padres de ambos grupos heteróticos ($\boldsymbol{K}_1$ y $\boldsymbol{K}_2$). La matriz se calculó utilizando 511 SNPs utilizando la función `A.mat` incluida en el paquete rrBLUP (Endelman, 2011). El nombre de las hileras y el nombre de las columnas de las matrices corresponden a los identificadores de los genotipos para el padre 1 y el padre 2.

El código R que se muestra en el listado 3.7 calcula las matrices $\boldsymbol{K}_1$, $\boldsymbol{K}_2$ y $\boldsymbol{K}_3$ y después ajusta el modelo (3.7) utilizando la función `lmerUvcov` usando como variable respuesta los valores fenotípicos observados para la altura de las plantas.

Listado de código 3.7: Ajustando el modelo para predicción de híbridos.

```
1 maize.Pheno$GCA1<-as.character(maize.Pheno$GCA1)
```

```
maize.Pheno$GCA2<-as.character(maize.Pheno$GCA2)
maize.Pheno$SCA<-as.character(maize.Pheno$SCA)

#Genomic relationship matrix for parent 1
GCA1<-unique(maize.Pheno$GCA1)
selected<-rownames(maize.G)%in%GCA1
K1<-maize.G[selected,selected]

#Genomic relationship matrix for parent 2
GCA2<-unique(maize.Pheno$GCA2)
selected<-rownames(maize.G)%in%GCA2
K2<-maize.G[selected,selected]

#kronecker, make.dimmanes is necessary to identify the hybrids
#with the label Parent 1:Parent 2
K3<-kronecker(K1,K2,make.dimnames=TRUE)

#Training set
trn<-which(!is.na(maize.Pheno$PlantHeight))

hybrid<-data.frame(y=maize.Pheno$PlantHeight[trn],
                   loc=maize.Pheno$Location[trn],
                   P1=maize.Pheno$GCA1[trn],
                   P2=maize.Pheno$GCA2[trn],
                   H=maize.Pheno$SCA[trn])

random<-list(P1=list(K=K1),
             P2=list(K=K2),
             H=list(K=K3))

#Fit the model
fm<-lmerUvcov(y~loc+(1|P1)+(1|P2)+(1|H), data=hybrid, Uvcov=random)

summary(fm)
```

El proceso para ajustar el modelo tomó alrededor de un segundo en una computadora con procesador Intel Core i7 @ 2.8 GHz. Una vez que el modelo ha sido ajustado, la función `summary` puede utilizarse para mostrar alguna información relevante. La salida de la función `summary` se muestra en el listado 3.8, de donde se obtienen las estimaciones para las habilidades combinatorias general y específica así como el parámetro de varianza asociado a los residuales, $\widehat{\sigma}_1^2 = 0.016385$, $\widehat{\sigma}_2^2 = 0.000841$, $\widehat{\sigma}_3^2 = 0.002047$ y $\widehat{\sigma}_e^2 = 0.001182$.

Listado de código 3.8: Resultados del ajuste del modelo en listado 3.7.

```
#...
Random effects:
 Groups   Name        Variance Std.Dev.
 H        (Intercept)  0.016385 0.12800
 P2       (Intercept)  0.000841 0.02900
 P1       (Intercept)  0.002047 0.04525
 Residual              0.001182 0.03438
Number of obs: 400, groups:  H, 100; P2, 20; P1, 20
#...
```

El rendimiento esperado para los híbridos no evaluados en el campo puede obtenerse combinando los resultados de las funciones `ranefUvcov` y `ranefUvcovNew`. El listado 3.9 muestra las instrucciones para obtener los BLUPs para la habilidad combinatoria específica de los híbridos. La función `ranefUvcov` es llamada internamente en `ranefUvcovNew`. En el listado 3.9 también se muestra cómo obtener los parámetros de varianza utilizando la función `VarCorr` y luego calcular la heredabilidad usando los resultados. Siguiendo a Covarrubias-Pazaran (2016), la heredabilidad puede calcularse usando la expresión $h^2 = \frac{\sigma_1^2 + \sigma_2^2}{\sigma_1^2 + \sigma_2^2 + \sigma_e^2}$, que da como resultado una herabilidad estimada de 0.70.

Listado de código 3.9: Predecir el rendimiento de los híbridos.

```
#Unobserved hybrid performance
blup_tst<-ranefUvcovNew(fm,Uvcov=list(H=list(K=K3)))
blup_tst$H

#variance parameters
vc<-VarCorr(fm)
print(vc,comp=c("Variance","Std.Dev."),digits=4)
variances<-as.data.frame(vc)$vcov
variances

#Heritability
h2<-sum(variances[2:3])/sum(variances[2:4])
h2
```

3.3.4 Ejemplo 4: Selección del parámetro ancho de banda con los kernel Gaussiano y exponencial

Gianola *et al.* (2006) introdujo el kernel Gaussiano en la genética cuantitativa con la idea de capturar los efectos genéticos en el problema de predicción genómica. El kernel Gaussiano se define como (ej., Morota y Gianola, 2014; Pérez y de los Campos, 2014):

$$\boldsymbol{K}\left(\mathbf{x}_i,\mathbf{x}_j\right)=\exp\left\{-\theta\frac{d_{ij}^2}{m}\right\}=\exp\left\{-\theta\frac{\sum_{k=1}^{m}\left(x_{ik}-x_{jk}\right)^2}{m}\right\}, \tag{3.8}$$

donde $\theta > 0$ es el parámetro ancho de banda, d_{ij} es la distancia Euclidiana, y x_{ik}, $(i,j=1,...,n,\ k=1,...,n)$ corresponde al marcador k para el individuo i y m corresponde al número de marcadores. El parámetro ancho de banda puede escogerse mediante validación cruzada, REML, máxima verosimilitud o métodos bayesianos. El kernel Gaussiano ha sido usado por diversos autores para la predicción genómica (ej., de los Campos *et al.*, 2010; Endelman, 2011; Pérez-Elizalde *et al.*, 2015). La selección del ancho de banda no es un problema fácil debido al alto costo computacional; de los Campos *et al.* (2010) y Endelman (2011) propusieron evaluar el desempeño del modelo el cual incluye el kernel Gaussiano sobre una malla de valores de θ. Dado que $\theta > 0$, definiendo $\rho = \exp(-\theta)$, entonces $\rho \in (0,1)$, con ello es posible definir una malla de valores para ρ y entonces, utilizando esos valores, se pueden mapear a los valores para $-\theta$, esto es $\theta = -\log\rho$ de modo que la ecuación (3.8) se puede reescribir como $\boldsymbol{K}\left(\mathbf{x}_i,\mathbf{x}_j\right) = \exp\left\{\log\rho\frac{d_{ij}^2}{m}\right\}$. Otro kernel que también se utiliza en la predicción genómica es el kernel exponencial (ej. Endelman, 2011):

$$\boldsymbol{K}\left(\mathbf{x}_i,\mathbf{x}_j\right)=\exp\left\{-\theta\frac{d_{ij}}{\sqrt{m}}\right\}. \tag{3.9}$$

Se programó la función `theta_optim` la cual ajusta el modelo (3.6) cuando uno de los términos aleatorios $(\boldsymbol{u}_j, j=1,...,q)$ incluye como matriz de varianza-covarianza el kernel Gaussiano o exponencial. Esta función toma como entrada los mismos objetos de la función `lmerUvcov` y una lista (kernel) que contiene (i) una matriz con distancias $\left(\left\{\frac{d_{ij}}{\sqrt{m}}\right\}, i,j=1,...,n\right)$ o la matriz de marcadores $(\{x_{ij}\}, i=1,...,n, j=1,...,m)$, (ii) el tipo de kernel (ya sea "gaussian" o "exponential"), y (iii) una secuencia de valores

para θ; los IDs de los individuos se toman directamente del nombre de las filas de las matrices que proporcionan las distancias o los marcadores. Si la secuencia de valores para θ no se proporcionan, entonces esta se genera automáticamente. El software ajusta el modelo mixto en (3.6) utilizando la función `lmerUvcov` para cada uno de los distintos valores de θ. El valor de θ que maximiza la log-verosimilitud se elige como la óptima. La función proporciona una lista con los siguientes elementos: un vector de valores de la log-verosimilitud, el valor óptimo de θ, el modelo ajustado y el kernel calculado con el valor óptimo de θ.

En el siguiente ejemplo se muestra como predecir el rendimiento de grano en trigo (`wheat599`) utilizando la matriz de relaciones derivadas del kernel Gaussiano o exponencial y la matriz de relaciones derivada del pedigrí. El modelo lineal para la predicción del rendimiento del grano para el ambiente uno es análogo al modelo (3.1):

$$\boldsymbol{y} = \mathbf{1}\mu + \boldsymbol{Z}_1\boldsymbol{u}_1 + \boldsymbol{Z}_2\boldsymbol{u}_2 + \boldsymbol{e}, \tag{3.10}$$

donde $\boldsymbol{y}$ es el rendimiento del grano, $\mathbf{1}$ es el vector de unos, μ es el intercepto, $\boldsymbol{u}_1 \sim NM\left(\mathbf{0}, \sigma_m^2\boldsymbol{K}\right)$, con $\boldsymbol{K}$ es el kernel que puede ser Gaussiano o exponencial, σ_m^2 es el parámetro de varianza asociado con los marcadores; $\boldsymbol{u}_2 \sim NM\left(\mathbf{0}, \sigma_a^2\boldsymbol{A}\right)$, $\boldsymbol{A}$ es la matriz de relaciones genéticas aditivas derivadas del pedigrí, y σ_a^2 es el parámetro de varianza asociado a $\boldsymbol{A}$, $\boldsymbol{Z}_1$, $\boldsymbol{Z}_2$ son matrices que conectan los fenotipos con los genotipos, y $\boldsymbol{e}$ es el término aleatorio distribuido como en el modelo (3.1).

El código en el listado 3.10 se usa para ajustar el modelo (3.10) con los kernels Gaussiano y exponencial. La Figura 3.2 muestra la log-verosimilitud perfil para diferentes valores de θ. Para el kernel Gaussiano, el máximo de la log-verosimilitud es igual a -513.0865, y se alcanza cuando $\widehat{\theta} = 1.1779$, mientras que para el kernel exponencial el máximo de la log-verosimilitud es igual a -511.585, y se alcanza cuando $\widehat{\theta} = 0.4107$. El código en el listado 3.11, muestra como obtener un resumen del ajuste del modelo con el valor óptimo del parámetro ancho de banda de donde se obtienen las estimaciones de los parámetros de varianza. El tiempo de ajuste del modelo es de aproximadamente 1608 segundos para el modelo con el kernel Gaussiano y 1701 segundos para el modelo con el kernel exponencial utilizando el mismo procesador descrito anteriormente. Note que la selección del parámetro ancho de banda es una tarea computacional muy intensiva, pero diversos autores (ej. Endelman, 2011; Pérez-Elizalde *et al.*, 2015) han reportado que la precisión

de la predicción con kernels no aditivos es más alta que la precisión de la predicción de Regresión Ridge (o equivalentemente al GBLUP).

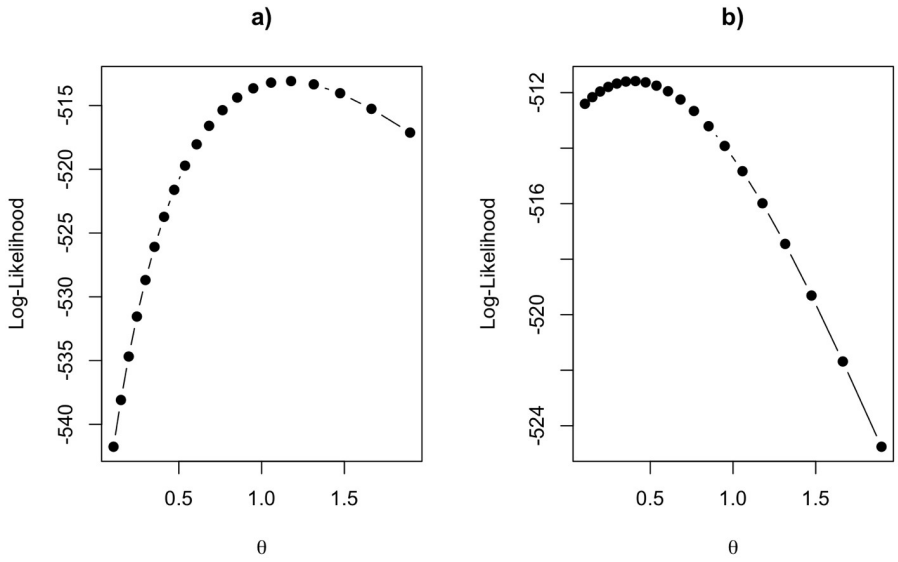

Figura 3.2: Perfil de la log-verosimilitud para diferentes valores de θ. a)Kernel Gaussiano, b)Kernel exponencial.

Listado de código 3.10: Kernel Gaussiano y exponencial.

```
library(lme4GS)
library(pedigreemm)

#Load data
data(wheat599)

## Complete and sort incomplete Pedigree using editPed
PedEdit<-editPed(sire=wheat.Pedigree$gpid1,dam=wheat.Pedigree$gpid2,
                 label=wheat.Pedigree$progenie, verbose=TRUE)

## Converted the data frame PedEdit into an S4 object of formal
## class 'Pedigree'
PedFinal<-with(PedEdit,pedigree(label=label,sire=sire,dam=dam))

#A
AFull<-getA(PedFinal)
GID<-unique(wheat.Pheno$GID)
selected<-rownames(AFull)%in%GID
A<-AFull[selected,selected]
A<-matrix(A,599,599)
rownames(A)<-colnames(A)<-rownames(AFull[selected,selected])
```

```
22
23 #X (markers)
24 X<-scale(wheat.X,center=TRUE,scale=TRUE)
25
26 #Phenotypes environment 1
27 e1<-which(wheat.Pheno$Env==1)
28 y<-wheat.Pheno[e1,]$Yield
29 GID<-as.character(wheat.Pheno[e1,]$GID)
30
31 wheat <- data.frame(y=y, ped=GID,k_id=GID)
32
33 fm1 <-theta_optim(y~(1|k_id)+(1|ped), Uvcov = list(ped=list(K=A)),
34                    kernel = list(kernel_type = "gaussian",MRK = X),
35                    data = wheat)
36
37 fm2<-theta_optim(y~(1|k_id)+(1|ped), Uvcov = list(ped=list(K=A)),
38                  kernel =list(kernel_type = "exponential",MRK = X),
39                  data = wheat)
40
41 par(mfrow=c(1,2))
42 plot(fm1$theta,fm1$LL,xlab=expression(theta),ylab="Log-Likelihood",
43      type="b",pch=19,main="a)")
44 plot(fm2$theta,fm2$LL,xlab=expression(theta),ylab="Log-Likelihood",
45      type="b",pch=19,main="b)")
```

Listado de código 3.11: Resumen de los resultados para los modelos ajustados.

```
1 summary(fm1$fm)
2 #Output (edited)
3 Random effects:
4  Groups   Name        Variance Std.Dev.
5  k_id     (Intercept) 0.29043  0.5389
6  ped      (Intercept) 0.07751  0.2784
7  Residual             0.03434  0.1853
8 Number of obs: 1198, groups: k_id, 599; ped, 599
9
10 Fixed effects:
11             Estimate Std. Error t value
12 (Intercept)   4.6510     0.1314    35.4
13
14 summary(fm2$fm)
15 #Output (edited)
16 Random effects:
17  Groups   Name        Variance Std.Dev.
```

```
 ped      (Intercept) 0.05154  0.2270
 k_id     (Intercept) 0.62952  0.7934
 Residual             0.03408  0.1846
Number of obs: 1198, groups:  k_id, 599; ped, 599
Fixed effects:
            Estimate Std. Error t value
(Intercept)   4.5311     0.5599   8.093
```

3.3.5 Tiempos de cómputo y comparación con otros software

Se ajustaron los modelos (3.6) y (3.7) utilizando las librerías de funciones `sommer` (Covarrubias-Pazaran, 2016) y BGLR (Pérez y de los Campos, 2014). En el caso del BGLR, el número de iteraciones para el muestreador de Gibbs se estableció a 30,000. No fue posible ajustar los modelos mediante la libreria rrBLUP (Endelman, 2011) ya que no es posible incluir más de una matriz de covarianza así como para el modelo (3.10). Las predicciones proporcionadas por los diferentes programas fueron similares. A continuación se presenta una pequeña comparación de los tiempos de cómputo necesarios para el ajuste de los modelos (3.6), (3.7) y (3.10) con los kernels Gaussiano y exponencial. Los modelos fueron ajustados utilizando un procesador Intel Core i7 @ 2.8-GHz en R-4.0.5 R Core Team (2021). En el cuadro 3.2 se presenta el tiempo resultante (en segundos) que se toma para el ajuste de los diferentes modelos. Algunas entradas en el Cuadro 3.2 están vacías ya que corresponden a los modelos que no pudieron ser ajustados en los paquetes de software correspondientes. De los resultados del Cuadro 3.2, se concluye que el paquete `sommer` fue el más rápido, seguido de los paquetes `lme4GS` y `BGLR`.

Cuadro 3.2: Comparación de tiempos (segundos) entre diferentes software para ajuste de modelos con kernels Gaussiano y exponencial.

Software	Versión	Ejemplos			
		Modelo (3.6)	Modelo (3.7)	Modelo (3.10) Gaussiano	Modelo (3.10) exponencial
`lme4GS`	0.1	81.5	1.3	1,608.8	1,701.1
`BGLR`	0.8	143.0	20.2	–	–
`sommer`	4.1.3	46.0	2.7	–	–

CAPÍTULO 4. AUMENTACIÓN ORTOGONAL DE DATOS PARA EL AJUSTE DEL MODELO LINEAL MIXTO

En este capítulo se presenta el uso de la técnica aumentación ortogonal de datos en los modelos de regresión ridge bayesiana y modelo lineal mixto mediante los algoritmos muestreador de Gibbs y Esperanza-Maximización para el ajuste de los modelos, utilizando datos de alta dimensión ($p \gg n$). Para ejemplificar los algoritmos se usaron datos del programa de mejoramiento de Maíz y Trigo (CIMMyT, https://www.cimmyt.org).

4.1 Introducción

Diversos problemas modernos de aprendizaje estadístico involucran el análisis de datos de alta dimensión, esto es común en estudios genéticos donde, por ejemplo, los fenotipos se relacionan con un gran número de variables predictoras al mismo tiempo (Pérez y de los Campos, 2014), así como en estudios médicos (van Wieringen, 2015). La implementación de modelos de regresión donde p es grande y n pequeño, plantea varios desafíos estadísticos y computacionales; incluyendo cómo enfrentar el problema de la llamada "maldición de la dimensionalidad" (Pérez y de los Campos, 2014). Los desarrollos recientes en procedimientos de estimación penalizada y selección de variables han hecho factible la implementación de regresiones con datos altamente dimensionales.

La Regresión Ridge (RR) (Hoerl y Kennard, 1970; Marquardt y Snee, 1975; McDonald, 2009; van Wieringen, 2015) es un método de contracción o penalización para la estimación del modelo de regresión lineal múltiple donde existe problemas de colinealidad entre las covariables x_j, $j = 1, ..., p$. La Regresión Ridge reduce los coeficientes introduciendo un término de penalización, λ, el cual varía de 0 a 1 (Marquardt y Snee, 1975). Cuando $\lambda \to 0$ la penalización es nula y el resultado es equivalente al estimador por MCO en el modelo de regresión lineal, por el contrario si $\lambda \to 1$ mayor es la penalización y menor es el valor

de los predictores β_j, $j = 1, ..., p$. Desde el punto de vista bayesiano, la regresión Ridge resulta de asignar un valor *a priori* a los parámetros de regresión β_j y de acuerdo a Pérez-Elizalde *et al.* (2022) suponer que son condicionalmente independientes. Este modelo bayesiano se conoce como Regresión Ridge Bayesiana (BRR, Bayesian Ridge Regression en inglés), y se asigna una distribución *a priori* Gaussiana al vector de parámetros $\boldsymbol{\beta}$, es decir $\boldsymbol{\beta}|\sigma^2_\beta \sim NM(\mathbf{0}, \sigma^2_\beta \boldsymbol{I})$.

La inferencia bayesiana en los modelos de regresión se basa principalmente en los métodos MCMC para la estimación de los parámetros de interés. En particular el muestreador de Gibbs es el algoritmo más utilizado ya que se pueden obtener la distribuciones condicionales completas *a posteriori* en forma cerrada. Sin embargo, dado que la estimación bayesiana de esos parámetros es un problema de integración de alta dimensión resulta ser una tarea computacional muy exigente para los métodos MCMC (Pérez-Elizalde *et al.*, 2022).

Zhao *et al.* (2020) proponen un algoritmo para resolver el problema computacional en el muestreador de Gibbs utilizando técnicas de aumentación de datos y cómputo paralelo. Los modelos que implementan estas estrategias computacionales son denominados "modelos de regresión bayesianos intensivos independientes (BayesXII)". Zhao *et al.* (2020) muestran que es posible muestrear las distribuciones condicionales completas de cada una de las covariables en un modelo de regresión de forma independiente en cada paso del muestreador de Gibbs. El método de Zhao *et al.* (2020) consiste en incluir p nuevos renglones en la matriz de covariables, de tal forma que la misma sea ortogonal e incluye la generación de variables respuesta "faltantes" y de esta forma se simplifican enormemente los cálculos. El procedimiento utilizado por Zhao *et al.* (2020) fue propuesto por Ghosh y Clyde (2011), se conoce como aumentación ortogonal de datos. Ghosh y Clyde (2011) mostraron que el enfoque de aumentación ortogonal de datos mantiene inalterada la distribución *a posteriori* original de interés, y esto hace que los cálculos en cada paso del algoritmo basado en cadenas de Markov Monte Carlo, pueda acelerarse k veces (teóricamente), donde k es el número de procesadores (CPUs) disponibles en un equipo de cómputo o clúster de cómputo.

Con un enfoque clásico, Xiong *et al.* (2016) introdujeron un algoritmo eficiente basado en la idea de Ghosh y Clyde (2011) sobre aumentación ortogonal de datos y lo aplicaron en la solución de problemas de optimización penalizada con el método de mínimos cuadrados. El algoritmo resultante fue denominado EM Ortogonal (OEM por sus siglas en inglés). Xiong *et al.* (2016) mostraron que el algoritmo OEM es altamente

eficiente para problemas de mínimos cuadrados penalizados y mínimos cuadrados a gran escala, y es considerablemente más rápido que los métodos cuando n es mucho mayor que p, $n \gg p$.

En este capítulo se proponen dos algoritmos basados en el método de aumentación ortogonal de Ghosh y Clyde (2011) para ajustar un modelo de regresión ridge bayesiana y el modelo lineal mixto, y poder eficientizar el cómputo en el muestreador de Gibbs y el algoritmo Esperanza-Maximización utilizando datos de alta dimensión, $p \gg n$. Los algoritmos propuestos se programaron en los lenguajes de programación R (R Core Team, 2021) y C (Kernighan y Ritchie, 1988).

4.2 Métodos

4.2.1 Modelo de regresión ridge bayesiana con aumentación ortogonal de datos

Considere el modelo de regresión lineal:

$$\boldsymbol{y} = \mathbf{1}\mu + \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{e},$$

donde $\boldsymbol{e} \sim NM(\mathbf{0}, \sigma_e^2\boldsymbol{I})$. La verosimilitud está dada por:

$$p(\boldsymbol{y}|\mu, \boldsymbol{\beta}, \sigma_e^2) = NM(\mathbf{1}\mu + \boldsymbol{X}\boldsymbol{\beta}, \sigma_e^2\boldsymbol{I}).$$

Asignando las siguientes distribuciones *a priori*:

$p(\mu) \propto 1$, $\boldsymbol{\beta}|\sigma_\beta^2 \sim NM(\mathbf{0}, \sigma_\beta^2\boldsymbol{I})$, $\sigma_e^2|df_e, S_e \sim \chi^{-2}(df_e, S_e)$, $\sigma_\beta^2|df_\beta, S_\beta \sim \chi^{-2}(df_\beta, S_\beta)$.

La distribución conjunta de $\boldsymbol{\theta} = (\mu, \boldsymbol{\beta}^t, \sigma_e^2, \sigma_\beta^2)^t$, con $H = \{df_e, S_e, df_\beta, S_\beta\}$, está dada por:

$$p(\boldsymbol{\theta}|H) = p(\mu) \times p(\boldsymbol{\beta}|\sigma_\beta^2) \times p(\sigma_e^2|df_e, S_e) \times p(\sigma_\beta^2|df_\beta, S_\beta).$$

La distribución conjunta de $p(\boldsymbol{y}, \boldsymbol{\theta})$ está dada por:

$$p(\boldsymbol{y}, \boldsymbol{\theta}) = p(\boldsymbol{y}|\boldsymbol{\beta}, \sigma_e^2) \times p(\mu) \times p(\boldsymbol{\beta}|\sigma_\beta^2) \times p(\sigma_e^2|df_e, S_e) \times p(\sigma_\beta^2|df_\beta, S_\beta).$$

Aplicando el teorema de Bayes definido en (2.25) se puede obtener la distribución conjunta *a posteriori*. La distribución conjunta no corresponde a una distribución conocida pero se pueden obtener muestras de la distribución *a posteriori* utilizando el algoritmo del muestreador de Gibbs, por lo que se requieren las distribuciones condicionales completas:

- $p(\mu|resto)$,
- $p(\boldsymbol{\beta}|resto)$,
- $p(\sigma_\beta^2|resto)$,
- $p(\sigma_e^2|resto)$.

La distribución condicional de $\beta_j|resto$, $j = 1, ..., p$ es normal con media y varianza igual a la solución (inversa de los coeficientes en el lado izquierdo de la igualdad) de la siguiente ecuación:

$$\left(\frac{1}{\sigma_e^2}\boldsymbol{x}_j^t\boldsymbol{x}_j + \frac{1}{\sigma_\beta^2}\right)\beta_j = \frac{1}{\sigma_e^2}\boldsymbol{x}_j^t\boldsymbol{e}_j,$$

donde $\boldsymbol{x}_j$ es la j-ésima columna de $\boldsymbol{X}$ y $\boldsymbol{e}_j = \boldsymbol{y} - \boldsymbol{1}\mu - \boldsymbol{X}_{-j}\boldsymbol{\beta}_{-j}$.

El algoritmo del muestreador de Gibbs con actualización escalar es:

1. Muestrear $\mu|resto \rightarrow \mu^{(t)}$
2. Muestrear $\boldsymbol{\beta}^{(t)}|resto$
   - Muestrear $\beta_1|resto \rightarrow \beta_1^{(t)}$.

   $\vdots$

- Muestrear $\beta_p|resto \rightarrow \beta_p^{(t)}$.

3. Muestrear $\sigma_\beta^2|resto \rightarrow \sigma_\beta^{2(t)}$.
4. Muestrear $\sigma_e^2|resto \rightarrow \sigma_e^{2(t)}$.
5. Repetir pasos 1-4 para $t = 1, ..., B$, donde B es el número de iteraciones MCMC.

Es claro que si n y p son grandes, entonces se requieren recursos computacionales considerables y el proceso de ajuste del modelo puede llegar a ser muy lento. El "cuello de botella" en el algoritmo del muestreador de Gibbs es el muestreo de $\beta_j|resto$, ya que no se puede realizar en "paralelo" por la naturaleza misma del algoritmo (cadena de Markov), para actualizar el valor β_j en la iteración $(t+1)$ es necesario que $\mu^{(t)}$, $\boldsymbol{\beta}_{-j}^{(t)}$, $\sigma_\beta^{2(t)}$ y $\sigma_e^{2(t)}$ hayan sido actualizados previamente. Note que el muestreo de $\boldsymbol{\beta}|resto$ tampoco es una opción, ya que es necesario generar muestras de una distribución normal multivariada de dimensión p, con una matriz de varianzas-covarianzas que cambia de iteración a iteración y que, por ejemplo tendría que factorizarse utilizando la descomposición de Cholesky para obtener muestras de manera eficiente.

Ahora se considera el caso en el que se realiza una "aumentación ortogonal" de los datos para el problema recién descrito. Sea $\boldsymbol{W}_o = (\mathbf{1}\ \boldsymbol{X})$ y sea

$$\boldsymbol{W}_c = \begin{pmatrix} \boldsymbol{W}_o \\ \boldsymbol{W}_a \end{pmatrix}, \tag{4.1}$$

entonces,

$$\boldsymbol{W}_c^t \boldsymbol{W}_c = (\boldsymbol{W}_o^t \boldsymbol{W}_a^t) \begin{pmatrix} \boldsymbol{W}_o \\ \boldsymbol{W}_a \end{pmatrix} = \boldsymbol{W}_o^t \boldsymbol{W}_o + \boldsymbol{W}_a^t \boldsymbol{W}_a.$$

Ghosh y Clyde (2011) mostraron que es posible construir $\boldsymbol{W}_a$, de tal forma que las columnas de $\boldsymbol{W}_c^t$ sean ortogonales, es decir:

$$\boldsymbol{W}_c^t \boldsymbol{W}_c = \boldsymbol{W}_o^t \boldsymbol{W}_o + \boldsymbol{W}_a^t \boldsymbol{W}_a = \boldsymbol{D},$$

Entonces, $\boldsymbol{W}_a^t \boldsymbol{W}_a = \boldsymbol{D} - \boldsymbol{W}_o^t \boldsymbol{W}_o$.

Sean $o_1 \geq \cdots \geq o_{p+1}$ los eigen-valores de $\boldsymbol{W}_o^t\boldsymbol{W}_o$ y sea $\boldsymbol{D} = diag(\delta_1, ..., \delta_{p+1})$, $\delta_1 \geq \cdots \geq \delta_{p+1}$. Sin pérdida de generalidad se puede fijar δ_j a o_1 y entonces $\boldsymbol{W}_a$ se obtiene al realizar la descomposición de Cholesky de $\boldsymbol{D} - \boldsymbol{W}_o^t\boldsymbol{W}_o$. No es necesario calcular todos los eigen-valores de la matriz $\boldsymbol{W}_o^t\boldsymbol{W}_o$, sino solamente el más grande de estos. El algoritmo de Lanczos (Golub y Van Loan, 2013; Lanczos, 1950) proporciona un procedimiento eficiente que permite calcular los eigenvalores más grandes y más pequeños de una matriz simétrica, así como los correspondientes eigen-vectores.

Zhao *et al.* (2020) empleó la aumentación ortogonal de datos recién descrita para acelerar el proceso de muestreo de las distribuciones condicionales en modelos de regresión con datos de alta dimensión en el contexto de predicción genómica, y fue ejemplificado en el caso donde se asigna una distribución de mezcla de distribuciones de dos componentes a $\boldsymbol{\beta}$ (spike-slab), como en el caso del modelo BayesCπ, las mismas ideas pueden ser empleadas en el caso de otros modelos de la familia del alfabeto bayesiano (Gianola *et al.*, 2009),

$$\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{y}_a \end{pmatrix} = \begin{pmatrix} \boldsymbol{1} \\ \boldsymbol{\eta}_a \end{pmatrix} \mu + \begin{pmatrix} \boldsymbol{X} \\ \boldsymbol{X}_a \end{pmatrix} \boldsymbol{\beta} + \begin{pmatrix} \boldsymbol{e} \\ \boldsymbol{e}_a \end{pmatrix},$$

donde $(\boldsymbol{\eta}_a \ \boldsymbol{X}_a) = \boldsymbol{W}_a$. El modelo puede ser escrito como:

$$\boldsymbol{y}_c = \boldsymbol{W}_c \begin{pmatrix} \mu \\ \boldsymbol{\beta} \end{pmatrix} + \boldsymbol{e}_c$$

donde $\boldsymbol{W}_c$ es conocido, y se supone que $\boldsymbol{e}_c \sim NM(\boldsymbol{0}, \sigma_e^2\boldsymbol{I})$. Note que $\boldsymbol{y}_a$ es un vector de "pseudo datos" faltantes mismo que es imputado en cada paso del muestreador de Gibbs que se realiza de la siguiente forma:

1. Muestrear de $\boldsymbol{y}_a|resto$,
2. Muestrear de $\mu|resto$,
3. Muestrear de $\boldsymbol{\beta}|resto$,
4. Muestrear de $\sigma_\beta^2|resto$,
5. Muestrear de $\sigma_e^2|resto$.

6. Repetir paros 1-5 B veces.

Utilizando la aumentación ortogonal de datos es posible mostrar que $\boldsymbol{y}_a|resto \sim NM(\boldsymbol{\eta}_a\mu + \boldsymbol{X}_a\boldsymbol{\beta}, \sigma_e^2\boldsymbol{I})$. Las distribuciones condicionales de $\beta_j|resto$ se obtienen como sigue:

$$\beta_j|resto \sim N\left(\frac{\boldsymbol{x}_j^t\boldsymbol{y} + \boldsymbol{x}_{ja}^t\boldsymbol{y}_a}{\delta_1 + \sigma_e^2/\sigma_\beta^2}, \frac{\sigma_e^2}{\delta_1 + \sigma_e^2/\sigma_\beta^2}\right), j = 1, ..., p.$$

La distribución condicional de $\mu|resto$ es como sigue:

$$\mu|resto \sim N\left(\frac{\mathbf{1}^t\boldsymbol{y} + \boldsymbol{\eta}_a\boldsymbol{y}_a}{\delta_1}, \frac{\sigma_e^2}{\delta_1}\right),$$

Finalmente, $\sigma_\beta^2|resto \sim \chi^{-2}(df_\beta + p, \boldsymbol{\beta}^t\boldsymbol{\beta} + S_\beta)$, $\sigma_e^2|resto \sim \chi^{-2}(df_e + n + p, \boldsymbol{y}_{corr}^t\boldsymbol{y}_{corr})$, donde $\boldsymbol{y}_{corr} = \boldsymbol{y}_c - \boldsymbol{W}_c(\mu,\ \boldsymbol{\beta}^t)^t$. Si se define $RSS := \boldsymbol{y}_{corr}^t\boldsymbol{y}_{corr}$, entonces se puede mostrar que:

$$RSS = \boldsymbol{y}^t\boldsymbol{y} + \boldsymbol{y}_a^t\boldsymbol{y}_a + \delta_1\mu^2 + \delta_1\boldsymbol{\beta}^t\boldsymbol{\beta} - 2\mu(\mathbf{1}^t\boldsymbol{y} + \boldsymbol{\eta}_a^t\boldsymbol{y}_a) - 2\boldsymbol{\beta}^t(\boldsymbol{X}^t\boldsymbol{y} + \boldsymbol{X}_a\boldsymbol{\eta}_a).$$

En la implementación del muestreador de Gibbs, la generación de las muestras de las distribuciones condicionales es sencilla y en términos computacionales el muestrear $\sigma_e^2|resto$ es uno de los más costosos, ya que se debe calcular RSS, pero se pueden ahorrar cálculos si se observa que RSS puede actualizarse cada vez que se muestrea μ, algún β_j o bien algún elemento de $\boldsymbol{y}_a$.

Actualización de RSS al muestrear $\mu|resto$

$$RSS_{new} = RSS_{old} + \delta_1(\mu_{new}^2 - \mu_{old}^2) - 2(\mathbf{1}^t\boldsymbol{y} + \boldsymbol{\eta}_a^t\boldsymbol{y}_a)(\mu_{new} - \mu_{old}).$$

Actualización de RSS al muestrear $\beta_j|resto$

$$RSS_{new} = RSS_{old} + \delta_1(\beta^2_{j,new} - \beta^2_{j,old}) - 2(\boldsymbol{x}^t_j\boldsymbol{y} + \boldsymbol{x}^t_{ja}\boldsymbol{\eta}_a).$$

Actualización de RSS al imputar $y_{ak}|resto$

$$RSS_{new} = RSS_{old} + y^2_{ak,new} - y^2_{ak,old} - 2\boldsymbol{l}_k(\mu\ \boldsymbol{\beta}^t)^t(y_{ak,new} - y_{ak,old}).$$

Selección de los hiper parámetros de las distribuciones a priori

Para la selección de los hiper parámetros se utiliza la estrategia descrita en Pérez y de los Campos (2014), quienes particionan la varianza de $\boldsymbol{y}$ en componentes atribuibles al error y los predictores la cual se controla por el parámetro $R2$ (proporción de varianza atribuida a predictores y residuales). Se fija $R2 = 0.5$, $df_\beta = 5$, $df_e = 5$, $S_e = var(y) \times (1 - R2) \times (df_e + 2)$, $S_\beta = var(y) \times R2 \times (df_\beta + 2)/MSx$, con MSx la suma de varianzas muestrales de las columnas de $\boldsymbol{X}$.

Implementación

El algoritmo recién descrito fue implementado en un programa escrito en los lenguajes de programación R (R Core Team, 2021) y C (Kernighan y Ritchie, 1988). Los listados de código en los apéndices A y B muestran como realizar la implementación del muestreador de Gibbs.

4.2.2 Modelo lineal mixto con aumentación ortogonal de datos

El modelo de Regresión Ridge es ampliamente utilizado en predicción con datos de alta dimensión. El modelo GBLUP se obtiene haciendo el cambio de variable $\boldsymbol{u} = \boldsymbol{X}\boldsymbol{\beta}$ y utilizando las propiedades de la distribución normal multivariada es posible mostrar que $\boldsymbol{u} \sim NM(\boldsymbol{0}, \sigma^2_\beta \boldsymbol{G})$, donde $\boldsymbol{G}$ es una matriz de relaciones genómicas de dimensión $n \times n$, para $\boldsymbol{G} = \boldsymbol{X}\boldsymbol{X}^t$. Aquí se propone el algoritmo para ajustar el modelo lineal mixto (4.2) basado en la aumentación ortogonal de datos y el algoritmo EM.

Considere el modelo:

$$\boldsymbol{y} = \mu\mathbf{1} + \boldsymbol{u} + \boldsymbol{e}, \tag{4.2}$$

con $\boldsymbol{u} \sim NM(\mathbf{0}, \sigma_u^2\boldsymbol{G})$ y $\boldsymbol{e} \sim NM(\mathbf{0}, \sigma_e^2\boldsymbol{I})$.

Sea $\boldsymbol{G} = \boldsymbol{\Gamma}\boldsymbol{\Lambda}\boldsymbol{\Gamma}^t$ la descomposición espectral de $\boldsymbol{G}$, entonces como se mostró en el capítulo anterior, el modelo puede ser re-escrito como:

$$\boldsymbol{y} = \mu\mathbf{1} + \boldsymbol{Z}_s\boldsymbol{u}^* + \boldsymbol{e},$$

donde $\boldsymbol{Z}_s = \boldsymbol{\Gamma}\boldsymbol{\Lambda}^{1/2}$ y $\boldsymbol{u}^* \sim NM(\mathbf{0}, \sigma_u^2\boldsymbol{I})$.

Siguiendo una estrategia similar al caso de aumentación ortogonal de datos en el modelo de Regresión Ridge, se propone hacer una aumentación de datos en el modelo mixto, es decir:

$$\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{y}_a \end{pmatrix} = \begin{pmatrix} \mathbf{1} \\ \boldsymbol{\eta}_a \end{pmatrix} \mu + \begin{pmatrix} \boldsymbol{Z}_s \\ \boldsymbol{Z}_a \end{pmatrix} \boldsymbol{\beta} + \begin{pmatrix} \boldsymbol{e} \\ \boldsymbol{e}_a \end{pmatrix},$$

donde $(\boldsymbol{\eta}_a \ \boldsymbol{Z}_a) = \boldsymbol{W}_a$. El modelo puede ser escrito como:

$$\boldsymbol{y}_c = \boldsymbol{W}_c \begin{pmatrix} \mu \\ \boldsymbol{\beta} \end{pmatrix} + \boldsymbol{e}_c,$$

donde $\boldsymbol{W}_o = (\mathbf{1} \ \ \boldsymbol{Z}_s)$ y $\boldsymbol{W}_c$ como se define en la ecuación (4.1).

Los pasos necesarios para ajustar el modelo mixto con aumentación de datos son los siguientes, suponiendo σ_u^2 y σ_e^2 fijos:

1. Calcular $\boldsymbol{W}_o^t\boldsymbol{W}_o$.
2. Obtener el eigen-valor más grande de $\boldsymbol{W}_o^t\boldsymbol{W}_o$, digamos δ_1.
3. Construir la matriz $\boldsymbol{D} = diag(\delta_1, ..., \delta_1)$.

4. Construir la matriz $\boldsymbol{D}-\boldsymbol{W}_o^t\boldsymbol{W}_o$ y obtener su descomposición de Cholesky, a partir de la cual se obtiene $\boldsymbol{\eta}_a$ y $\boldsymbol{Z}_a$, es decir $\boldsymbol{L}=(\boldsymbol{\eta}_a \;\; \boldsymbol{Z}_a)$.
5. Imputar $\boldsymbol{y}_a^{[s]}$ con $\boldsymbol{L}(\mu^{[s]} \;\; \boldsymbol{u}^{*[s]})^t$.
6. Actualizar μ.
7. Actualizar $\boldsymbol{u}^*$.
8. Repetir pasos 5-7 hasta convergencia, $[s]=1,2,...$

Para actualizar μ se utiliza la expresión:

$$\mu^{[s+1]}=(\mathbf{1}^t\mathbf{1}+\boldsymbol{\eta}_a^t\boldsymbol{\eta}_a)^{-1}(\mathbf{1}^t \;\; \boldsymbol{\eta}_a^t)\left[\begin{pmatrix}\boldsymbol{y}\\ \boldsymbol{y}_a\end{pmatrix}-\begin{pmatrix}\boldsymbol{Z}_s\\ \boldsymbol{Z}_a\end{pmatrix}\boldsymbol{u}^{*[s]}\right].$$

Para actualizar $\boldsymbol{u}^*$ se utiliza la expresión:

$$\begin{aligned}
\boldsymbol{u}^{*[s+1]} &= \left[\begin{pmatrix}\boldsymbol{Z}_s\\ \boldsymbol{Z}_a\end{pmatrix}^t\begin{pmatrix}\boldsymbol{Z}_s\\ \boldsymbol{Z}_a\end{pmatrix}+\frac{\sigma_e^{2[s]}}{\sigma_u^{2[s]}}\boldsymbol{I}\right]^{-1}\begin{pmatrix}\boldsymbol{Z}_s\\ \boldsymbol{Z}_a\end{pmatrix}^t\left[\begin{pmatrix}\boldsymbol{y}\\ \boldsymbol{y}_a\end{pmatrix}-\begin{pmatrix}\mathbf{1}\\ \boldsymbol{\eta}_a\end{pmatrix}\mu^{[s]}\right]\\
&= \left[\boldsymbol{Z}_s^t\boldsymbol{Z}_s+\boldsymbol{Z}_a^t\boldsymbol{Z}_a+\frac{\sigma_e^{2[s]}}{\sigma_u^{2[s]}}\boldsymbol{I}\right]^{-1}\begin{pmatrix}\boldsymbol{Z}_s\\ \boldsymbol{Z}_a\end{pmatrix}^t\left[\begin{pmatrix}\boldsymbol{y}\\ \boldsymbol{y}_a\end{pmatrix}-\begin{pmatrix}\mathbf{1}\\ \boldsymbol{\eta}_a\end{pmatrix}\mu^{[s]}\right]\\
&= \left[\delta_1\boldsymbol{I}+\frac{\sigma_e^{2[s]}}{\sigma_u^{2[s]}}\boldsymbol{I}\right]^{-1}\begin{pmatrix}\boldsymbol{Z}_s\\ \boldsymbol{Z}_a\end{pmatrix}^t\left[\begin{pmatrix}\boldsymbol{y}\\ \boldsymbol{y}_a\end{pmatrix}-\begin{pmatrix}\mathbf{1}\\ \boldsymbol{\eta}_a\end{pmatrix}\mu^{[s]}\right]\\
&= \left(\delta_1+\frac{\sigma_e^{2[s]}}{\sigma_u^{2[s]}}\right)^{-1}\begin{pmatrix}\boldsymbol{Z}_s\\ \boldsymbol{Z}_a\end{pmatrix}^t\left[\begin{pmatrix}\boldsymbol{y}\\ \boldsymbol{y}_a\end{pmatrix}-\begin{pmatrix}\mathbf{1}\\ \boldsymbol{\eta}_a\end{pmatrix}\mu^{[s]}\right].
\end{aligned}$$

El criterio de parada del algoritmo EM se puede definir en términos de las diferencias entre evaluaciones sucesivas de la función de log-verosimilitud o bien en términos de las diferencias entre parámetros entre dos evaluaciones sucesivas. En el presente trabajo se utiliza la segunda estrategia y el programa se detiene cuando el máximo de las diferencias en valor absoluto entre iteraciones sucesivas de los parámetros a estimar es menor que una tolerancia fijada de antemano, por ejemplo 1×10^{-4}. Una vez el algoritmo converge se tiene $\hat{\mu}$ y $\widehat{\boldsymbol{u}}^*$, de donde se obtiene $\widehat{\boldsymbol{u}}=\boldsymbol{Z}_s\widehat{\boldsymbol{u}}^*$.

El algoritmo supone que los parámetros de varianza σ_e^2 y σ_u^2 son conocidos, lo cual en la práctica no es cierto y los parámetros deben estimarse usando los datos. Zhou y Stephens (2012) propusieron un algoritmo basado en la descomposición espectral de $\boldsymbol{G}$ que permite estimar estos parámetros utilizando el algoritmo de Brent (Brent, 1973), el método se describe brevemente a continuación para el caso del modelo de interés.

Partiendo del modelo (4.2) y re-escribiendo las distribuciones de $\boldsymbol{u}$ y $\boldsymbol{e}$ de la forma siguiente, $\boldsymbol{u} \sim N(\boldsymbol{0}, \lambda\tau^{-1}\boldsymbol{G})$, y $\boldsymbol{e} \sim NM(\boldsymbol{0}, \tau^{-1}\boldsymbol{I})$ con $\lambda = \frac{\sigma_u^2}{\sigma_e^2}$ y $\tau^{-1} = \sigma_e^2$. Como se mostró en el capítulo de revisión de literatura (2), $\boldsymbol{y} \sim NM(\boldsymbol{1}\mu, \boldsymbol{V})$, donde:

$$\begin{aligned} \boldsymbol{V} &= \sigma_u^2\boldsymbol{G} + \sigma_e^2\boldsymbol{I} \\ &= \tau^{-1}(\lambda\boldsymbol{G} + \boldsymbol{I}) \\ &= \tau^{-1}\boldsymbol{H}, \end{aligned}$$

donde $\boldsymbol{H} = \lambda\boldsymbol{G} + \boldsymbol{I}$. La función de log-verosimilitud para los parámetros de interés está dada por:

$$\ell(\lambda, \tau, \mu; \boldsymbol{y}) = \frac{n}{2}\log(\tau) - \frac{n}{2}\log(2\pi) - \frac{1}{2}\log|\boldsymbol{H}| - \frac{1}{2}\tau(\boldsymbol{y} - \boldsymbol{1}\mu)^t\boldsymbol{H}^{-1}(\boldsymbol{y} - \boldsymbol{1}\mu).$$

Suponiendo λ es conocido, entonces los estimadores para μ y τ están dados por:

$$\begin{aligned} \hat{\mu} &= \left(\boldsymbol{1}^t\boldsymbol{H}^{-1}\boldsymbol{1}\right)^{-1}\boldsymbol{1}^t\boldsymbol{H}^{-1}\boldsymbol{y} \\ \hat{\tau} &= \frac{n}{(\boldsymbol{y} - \boldsymbol{1}\hat{\mu})^t\boldsymbol{H}^{-1}(\boldsymbol{y} - \boldsymbol{1}\hat{\mu})} = \frac{n}{\boldsymbol{y}^t\boldsymbol{P}\boldsymbol{y}}, \end{aligned}$$

donde $\boldsymbol{P} = \boldsymbol{H}^{-1} - \boldsymbol{H}^{-1}\boldsymbol{1}(\boldsymbol{1}^t\boldsymbol{H}^{-1}\boldsymbol{1})^{-1}\boldsymbol{1}^t\boldsymbol{H}^{-1}$. Sustituyendo las soluciones anteriores en la función de log-verosimilitud se obtiene una función que depende solamente del parámetro λ y está dada por:

$$\ell(\lambda; \boldsymbol{y}) = \frac{n}{2}\log\left(\frac{n}{2\pi}\right) - \frac{n}{2} - \frac{1}{2}\log|\boldsymbol{H}| - \frac{n}{2}\log(\boldsymbol{y}^t\boldsymbol{P}\boldsymbol{y}).$$

Derivando la función anterior con respecto a λ se obtiene:

$$\frac{\partial\ell(\lambda; \boldsymbol{y})}{\partial\lambda} = -\frac{1}{2}tr(\boldsymbol{H}^{-1}\boldsymbol{G}) + \frac{n}{2}\frac{\boldsymbol{y}\boldsymbol{P}\boldsymbol{G}\boldsymbol{P}\boldsymbol{y}}{\boldsymbol{y}^t\boldsymbol{P}\boldsymbol{y}} \tag{4.3}$$

La función (4.3) puede evaluarse de forma muy rápida y eficiente re-expresando los téminos en función de la descomposición espectral de las matrices $\boldsymbol{G} = \boldsymbol{\Lambda}\boldsymbol{\Gamma}\boldsymbol{\Lambda}^t$ y $\boldsymbol{H}$, la cual puede ser re-escrita como:

$$\begin{aligned}
\boldsymbol{H} &= \lambda\boldsymbol{G} + \boldsymbol{I} \\
&= \lambda\boldsymbol{\Lambda}\boldsymbol{\Gamma}\boldsymbol{\Lambda}^t + \boldsymbol{I} \\
&= \lambda\boldsymbol{\Lambda}\boldsymbol{\Gamma}\boldsymbol{\Lambda}^t + \boldsymbol{\Lambda}\boldsymbol{\Lambda}^t \\
&= \lambda\boldsymbol{\Lambda}\boldsymbol{\Gamma}\boldsymbol{\Lambda}^t + \boldsymbol{\Lambda}\boldsymbol{I}\boldsymbol{\Lambda}^t \\
&= \boldsymbol{\Lambda}(\lambda\boldsymbol{\Gamma} + \boldsymbol{I})\boldsymbol{\Lambda}^t.
\end{aligned}$$

La función (4.3) se iguala a 0 y se encuentra la raíz de la misma empleando métodos numéricos. Zhou y Stephens (2012) emplearon el algoritmo de Brent (Brent, 1973) para dar solución al problema suponiendo que $\lambda \in (1 \times 10^{-5}, 1 \times 10^{5})$.

Implementación

El algoritmo que permite ajustar el modelo mixto con aumentación ortogonal de datos fue implementado en un programa escrito en el lenguaje de programación R (R Core Team, 2021). El listado de código en el apéndice C muestra como realizar la implementación del algoritmo EM.

4.3 Costo computacional de la implementación de los algoritmos con aumentación de datos

El costo computacional se puede dividir en dos partes: 1) Uso de memoria RAM y 2) Complejidad del algoritmo. Se toma como referencia el algoritmo para ajustar el modelo BRR. Para la complejidad del algoritmo, en ciencias de la computación es usual comparar la eficiencia de los algoritmos basados en el número de operaciones primitivas (asignación de un identificador a un objeto, determinar objeto asociado a un identificador, operaciones aritméticas, comparación de dos números, acceder a un elemento de un arreglo, llamada de una función entre las más comunes). El tiempo de ejecución de las operaciones primitivas se considera constante y el número de operaciones primitivas realizadas en la implementación de un algoritmo es proporcional al tiempo de ejecución y para realizar las comparaciones se utiliza la notación O (Big-Oh) (Weiss, 2012).

En el algoritmo convencial de la regresión ridge bayesiana las operaciones que consumen más tiempo son las relacionadas con el muestreo de $\beta_j|resto$ y la complejidad es $O(n \times p)$ por cada ciclo del muestreador de Gibbs. Para el modelo de regresión ridge bayesiano con aumentación ortogonal de datos el muestreo de $\beta_j|resto$ tiene una complejidad $O((n+p) \times p)/k)$, donde k es el número de CPUs cuando el proceso de muestreo se realiza en paralelo. En este segundo caso debe recordarse que además se tienen que imputar los datos faltantes cuya complejidad es $O(p \times p/k)$ (Zhao *et al.*, 2020), en este mismo caso se debe añadir el costo computacional para construir $\boldsymbol{W}_a$, de complejidad $O(n \times p^2 + p^3)$ que incluye el cálculo de $\boldsymbol{W}_o^t\boldsymbol{W}_o$, el cálculo del máximo de los eigenvalores de esta misma matriz, así como la descomposición de Cholesky de $\boldsymbol{D} - \boldsymbol{W}_o^t\boldsymbol{W}_o$. La gran mayoría de estas operaciones se pueden realizar de forma paralela en múltiples procesadores de un mismo equipo de cómputo usando las librerías de álgebra de matrices especializadas como openBLAS (`https://www.openblas.net`), Intel MKL, etc. En el caso de problemas más complejos, es posible usar varios equipos de cómputo que realicen el trabajo en forma colaborativa usando tecnologías como OpenMPI (`https://www.open-mpi.org`) y bibliotecas de álgebra de matrices desarrolladas para esos entornos, por ejemplo ScaLAPACK (`http://www.netlib.org/scalapack/`). Finalmente en el caso del algoritmo EM con aumentación ortogonal de datos es necesario realizar la descomposición espectral de la matriz $\boldsymbol{G}$ con una complejidad $O(n^3)$, misma que se utiliza en el cálculo de los parámetros de varianza y re-parametrizació del modelo. En este último caso tiene que

añadirse el costo del cálculo de $\boldsymbol{W}_a$ de complejidad $O(n^3)$. En cada iteración del EM se deben imputar los datos faltantes con complejidad $O(n^2)$, actualizar μ con complejidad $O(n^2)$ y actualizar $\boldsymbol{u}^*$ con complejidad $O(n^2)$.

4.4 Aplicación con datos reales

En esta sección se ejemplifica el uso de los modelos con aumentación ortogonal de datos discutidos en la sección previa usando datos reales. Los datos corresponden a rendimiento de 561 líneas elite del programa de mejoramiento de trigo del Centro Internacional de Mejoramiento de Maíz y Trigo (CIMMyT, `https://www.cimmyt.org`). Las líneas fueron genotipeadas con 5500 marcadores moleculares y se tiene también la información del pedigrí. Adicionalmente el conjunto de datos incluye datos del espectroscopia de infrarrojo cercano (NIR, Near Infrared Spectroscopy en inglés) para $p = 1059$ longitudes de onda. El problema consiste en predecir el rendimiento de las líneas de trigo usando la información disponible (marcadores, pedigrí, NIR). Los datos fueron previamente analizados por Cuevas *et al.* (2019) quienes ajustaron modelos que incluyen como predictores los marcadores, la matriz de relaciones aditivas derivadas de pedigrí y los datos de NIR. Los modelos considerados incluyen modelos lineales así como otros modelos más sofisticados para aprendizaje profundo. En este ejemplo solo se consideran los datos de NIR, el ajuste de modelos lineales con las otras fuentes de información ya fue discutido en el capítulo previo y los modelos de aprendizaje profundo considerados por los autores no es cubierto por el presente trabajo de investigación.

Cuevas *et al.* (2019) normalizó los datos de NIR tomando la primera (NIR1) y segunda (NIR2) derivada con la finalidad de maximizar las diferencias entre las señales muestrales. Los datos fueron normalizados empleando la función `savitzkyGolay` en la librería de funciones `prospectr` (Stevens y Ramirez-Lopez, 2022) que implementa la metodología de Savitzky y Golay (1964) para separar la señal del ruido y eliminar datos atípicos en los datos de espectrometría. En lo sucesivo se utilizan solo los datos de NIR2 y la matriz de predictores correspondientes será referida como $\boldsymbol{X}$. La Figura 4.1 muestra un mapa de calor y dendograma generada a partir de la matriz de relaciones entre las líneas de trigo calculada usando la ecuación $\boldsymbol{G} = \boldsymbol{X}^*\boldsymbol{X}^{*t}/p$, donde $\boldsymbol{X}^*$ corresponde a la matriz NIR2 centrada y estandarizada por columnas y p es el número de longitudes de onda. En esta figura se pueden observar claramente grupos de líneas con perfiles de NIR

similares, los grupos aparecen como bloques en el mapa de calor y se muestran también en los dendogramas que aparecen en los márgenes del mismo. La Figura 4.2 muestra el histograma para los datos de rendimiento de trigo, de donde se observa que la distribución es aproximadamente simétrica y bien podría aproximarse utilizando el modelo normal.

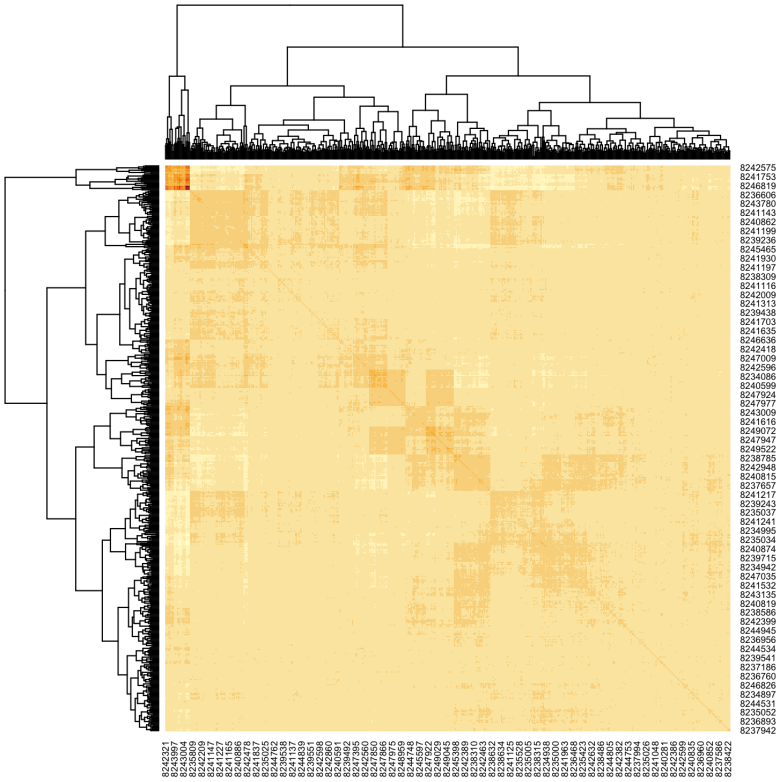

Figura 4.1: Mapa de calor para matriz de relaciones de líneas de trigo con datos de NIR.

Para el análisis de los datos se ajustaron 3 modelos: 1) Modelo de Regresión Ridge usando el paquete BGLR (Pérez y de los Campos, 2014), 2) Modelo de Regresión Ridge con aumentación ortogonal de datos y 3) Modelo mixto con aumentación ortogonal de datos. Se calculó primero una matriz de relaciones entre las líneas usando los datos de NIR2, obteniendo la matriz $\boldsymbol{G}$ descrita previamente, posteriormente se calculó la descomposición espectral de la misma, $\boldsymbol{G} = \boldsymbol{\Gamma\Lambda\Gamma}^t$, a partir de la cual se obtiene $\boldsymbol{\Gamma\Lambda}^{1/2}$ de tal manera que el modelo de Regresión Ridge a ajustar es $\boldsymbol{y} = \boldsymbol{1}\mu + \boldsymbol{\Gamma\Lambda}^{1/2}\boldsymbol{u}^* + \boldsymbol{e}$, con $\boldsymbol{u}^*|\sigma_u^2 \sim NM(\boldsymbol{0}, \sigma_u^2\boldsymbol{I})$ y $\boldsymbol{e}|\sigma_e^2 \sim NM(\boldsymbol{0}, \sigma_e^2\boldsymbol{I})$. El modelo mixto a ajustar es el descrito en las secciones previas, $\boldsymbol{y} = \boldsymbol{1}\mu + \boldsymbol{u} + \boldsymbol{e}$ con $\boldsymbol{u} \sim N(\boldsymbol{0}, \sigma_u^2\boldsymbol{G})$.

Para el ajuste de los modelos la variable respuesta $\boldsymbol{y}$ se centró y estandarizó, de tal forma que los parámetros de varianza σ_u^2 y σ_e^2 puedan interpretarse como "proporción de la

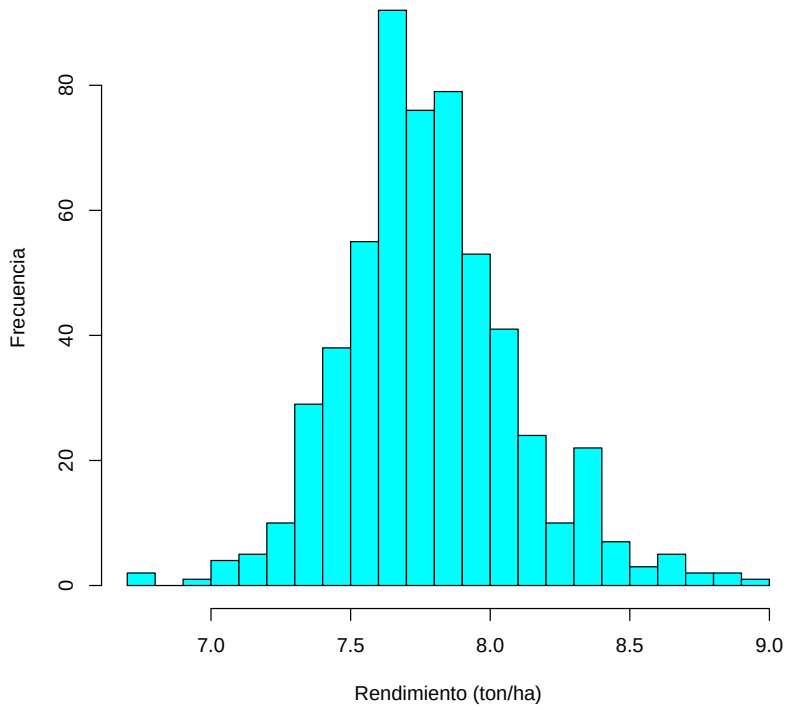

Figura 4.2: Histograma para datos de rendimiento de trigo.

varianza de los fenotipos" explicada por los predictores y el ambiente. Para los modelos de Regresión Ridge, la inferencia para los parámetros de interés fue basada en 25,000 muestras MCMC obtenidas después de descartar 25,000 muestras MCMC que sirvieron como calentamiento. De las 25,000 muestras utilizadas en el proceso de inferencia se tomaron sub-muestras con un adelgazamiento (thin) de 10. El cuadro 4.1 muestra los estimadores puntuales para σ_e^2 y σ_u^2. En el caso de los modelos de regresión Ridge se tomó como estimador puntual la media de la distribución *a posteriori* y en el caso del algoritmo EM corresponde a los estimadores de máxima verosimilitud calculados usando la metodología de Zhou y Stephens (2012).

Los estimadores puntuales obtenidos utilizando los 3 modelos son bastante similares, en el caso de σ_e^2 hay diferencias en la segunda cifra decimal y para σ_u^2 la diferencia máxima entre estimadores es de 0.04. En el caso de los modelos de Regresión Ridge dado que se utilizaron los mismos hiper-parámetros en las distribuciones "a priori" las diferencias son atribuibles al error de muestreo Monte Carlo. La Figura 4.3 muestra la gráfica de traza (panel izquierdo) para los parámetros de varianza σ_e^2 y σ_u^2 construidas partiendo de las cadena MCMC después de eliminar las iteraciones de calentamiento y haber realizado el adelgazamiento, el panel derecho muestra la función de densidad estimada mediante

Cuadro 4.1: Estimadores puntuales de σ_e^2, σ_u^2.

Modelo	$\hat{\sigma}_e^2$	$\hat{\sigma}_u^2$
BRR	0.7769	0.4656
BRR ortogonal	0.7843	0.4341
EM	0.7803	0.3957

suavizamiento de kernel (Silverman, 1986). La figura 4.4 muestra la gráfica de traza (panel izquierdo) y la función de densidad estimada mediante suavizamiento de kernel (panel derecho) para las muestras MCMC obtenidas usando aumentación ortogonal de datos. En el caso σ_u^2 la gráfica de la traza revela que es necesario aumentar el número de iteraciones MCMC o incluso pudiera existir algún problema de convergencia. La Figura 4.5 muestra el histograma de la media para los datos imputados para el modelo de Regresión Ridge con aumentación de datos, de donde se observan algunos valores extremos de -20 ó 30, pero debe recordarse que no hay restricción alguna para los valores que toman los componentes del vector $\boldsymbol{y}_a$ y cada uno de los elementos puede tomar valores en $\mathbb{R}$. La figura 4.6 presenta el máximo del valor absoluto de las diferencias para los parámetros a estimar con el algoritmo EM con aumentación ortogonal de datos en cada iteración. En el algoritmo EM se detuvo cuando estas diferencias fueron menores a 1×10^{-4} lo cual se logró después de 785 iteraciones. Finalmente la Figura 4.7 presenta una gráfica de $\widehat{\boldsymbol{u}}_{BRR} = \boldsymbol{\Gamma}\boldsymbol{\Lambda}^{1/2}\widehat{\boldsymbol{u}}^*$ para los modelos de Regresión Ridge vs los valores $\widehat{\boldsymbol{u}}_{EM}$ obtenidos mediante el algoritmo EM. Estos valores son los "valores de cría" asociados a las líneas de trigo y son los utilizados en los procesos de selección. De la figura 4.7 es claro que los resultados coinciden para las tres metodologías utilizadas, los coeficientes de correlación de Pearson entre estos valores para dos modelos dados en todos los casos es superior a 0.99.

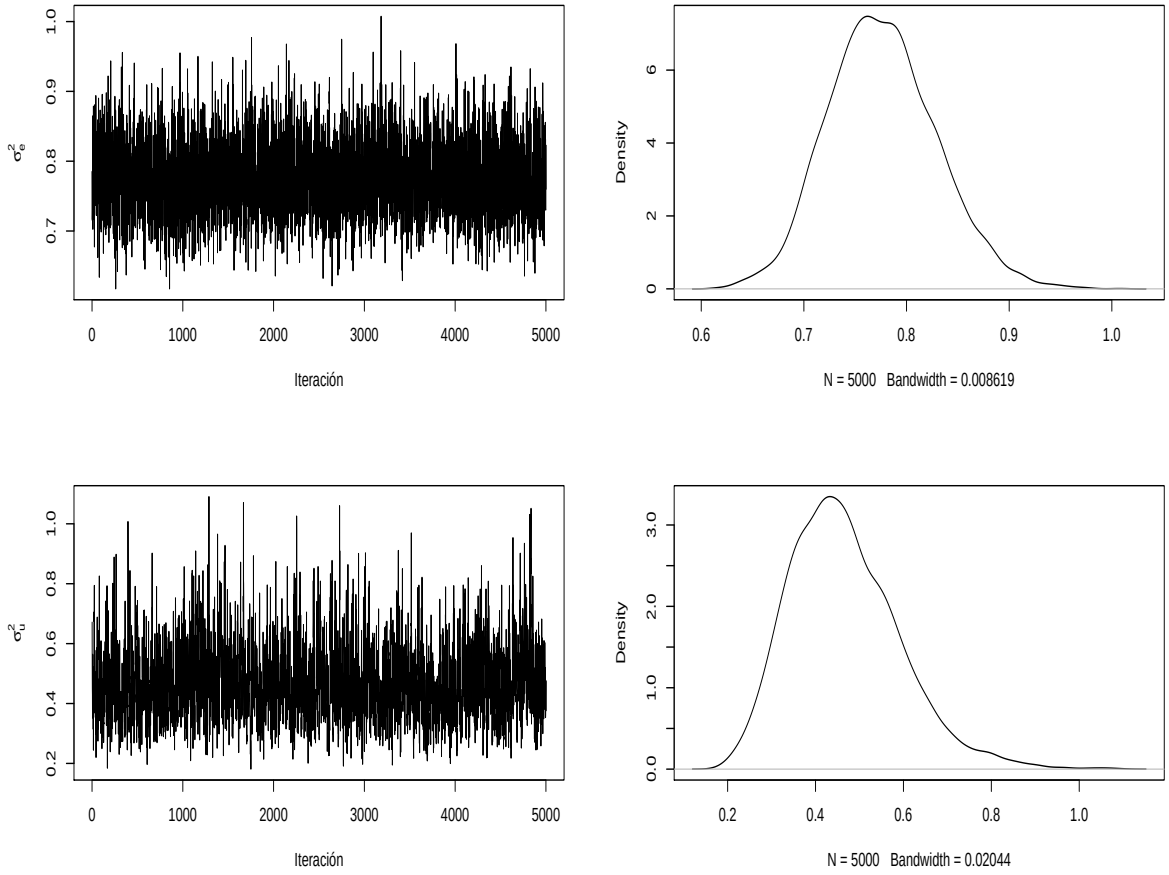

Figura 4.3: Parámetros de varianza obtenidos mediante BGLR.

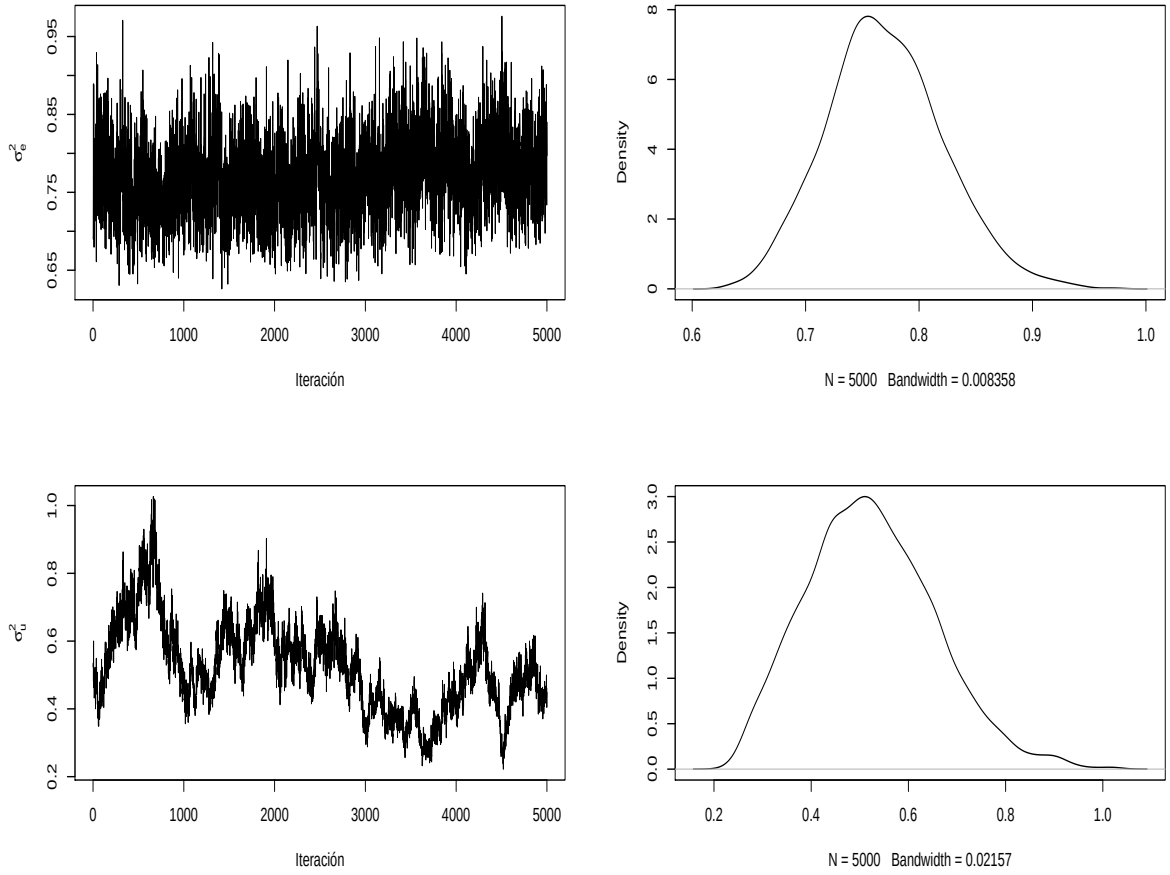

Figura 4.4: Parámetros de varianza con aumentación ortogonal de datos.

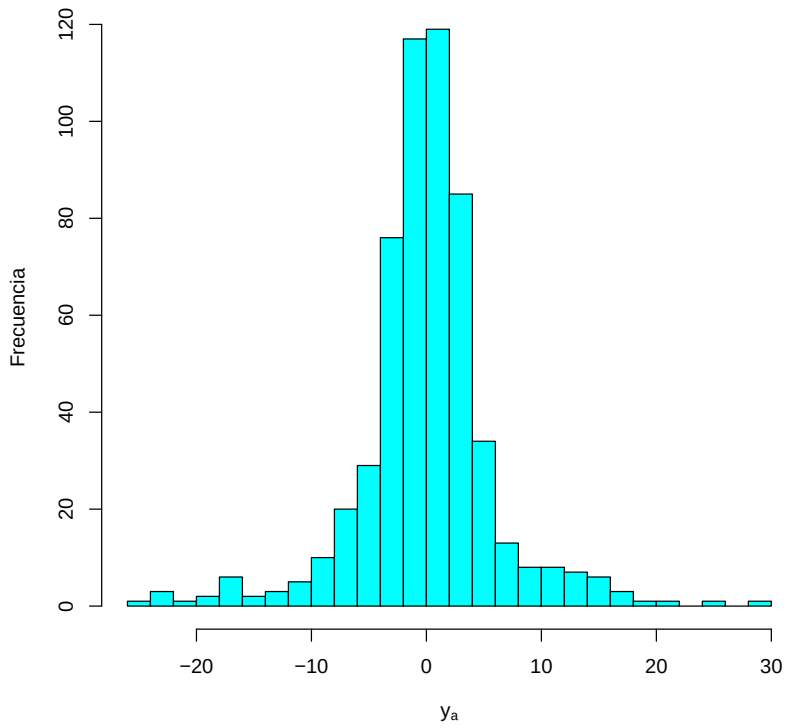

Figura 4.5: Histograma de $\boldsymbol{y}_a$ para Regresión Ridge con aumentación ortogonal de datos.

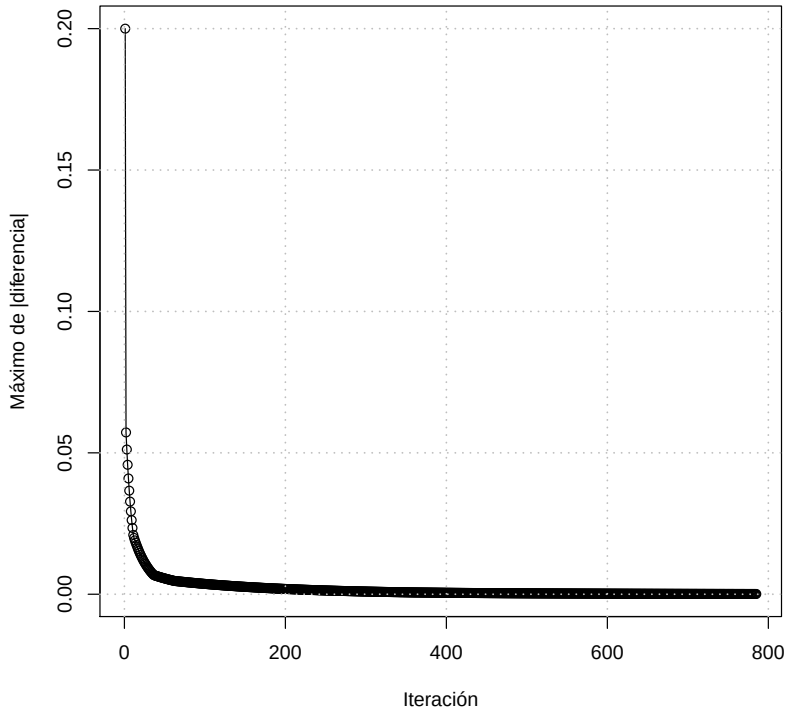

Figura 4.6: Máximo del valor absoluto de las diferencias de los parámetros estimados en iteraciones sucesivas con el algoritmo EM.

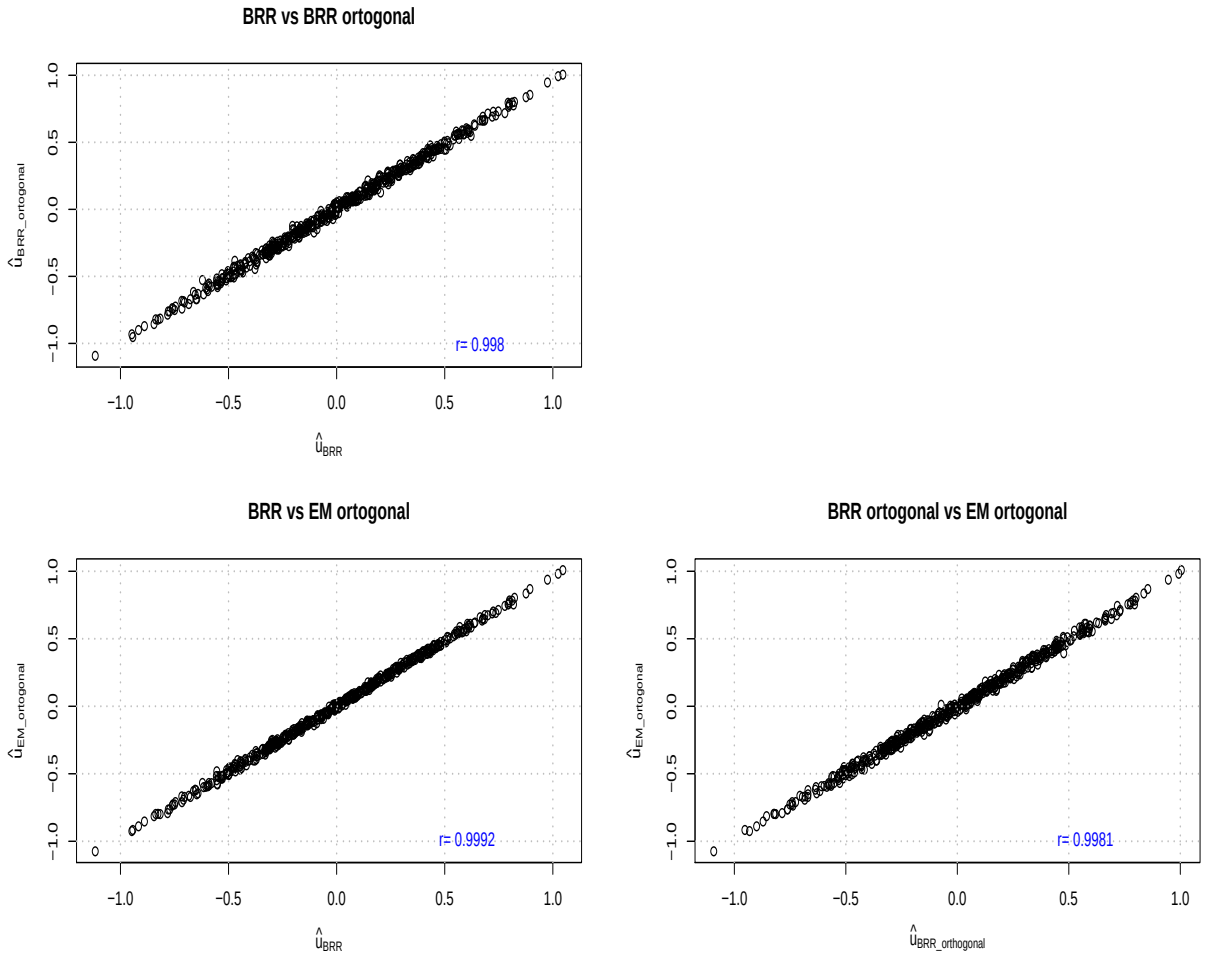

Figura 4.7: BLUPs obtenidos mediante Regresión Ridge ($\widehat{\boldsymbol{u}}_{BRR} = \boldsymbol{\Gamma}\boldsymbol{\Lambda}^{1/2}\widehat{\boldsymbol{u}}^{*}$) y el algoritmo EM ($\widehat{\boldsymbol{u}}_{EM}$).

CAPÍTULO 5. CONCLUSIONES

Los métodos desarrollados y presentados forman parte de una solución del problema de la dimensionalidad en el modelo lineal mixto y mejoraron la eficiencia de los recursos computacionales utilizando datos reales de alta dimensión aplicados para selección genómica. Los métodos son alternativas eficientes de otros métodos para el ajuste del modelo lineal mixto por lo que se pueden utilizar en todas las áreas que presenten datos con alta dimensión.

Se desarrolló un paquete de software en R que puede utilizarse para ajustar modelos mixtos con estructuras de covarianza definidas por el usuario para efectos aleatorios. El software fue desarrollado para aplicaciones de selección genómica, principalmente para aplicaciones en mejoramiento de plantas con conjuntos de datos de tamaño pequeños a moderados, pero dado que los modelos mixtos son ampliamente utilizados el paquete puede utilizarse en otras áreas de investigación. El software ajusta el modelo mixto utilizando rutinas computacionales conocidas y ampliamente probadas disponibles en el paquete `lme4`. El software proporciona una interfaz intuitiva y fácil de usar que permite a los usuarios adaptar a una amplia variedad de modelos mixtos.

Se desarrollaron dos algoritmos basados en la aumentación ortogonal de datos para los ajustes de los modelos de regresión ridge bayesiana utilizando el muestreador de Gibbs y lineal mixto utilizando el algoritmo Esperanza-Maximización. La aumentación ortogonal de datos es una alternativa práctica para acelerar computacionalmente los algoritmos conocidos, tales como el muestreador de Gibbs y Esperanza-Maximización.

CAPÍTULO 6. LITERATURA CITADA

Acosta-Pech, R., Crossa, J., de los Campos, G., Teyssèdre, S., Claustres, B., Pérez-Elizalde, S. y Pérez-Rodríguez, P. (2017). Genomic models with genotype × environment interaction for predicting hybrid performance: an application in maize hybrids. *Theoretical and Applied Genetics*, 130, 7, 1431–1440.

Akaike, H. (1973). Theory and an extension of the maximum likelihood principal. En *International symposium on information theory. Budapest, Hungary: Akademiai Kaiado*.

Barker, A. A. (1965). Monte carlo calculations of the radial distribution functions for a proton electron plasma. *Australian Journal of Physics*, 18, 2, 119–134.

Bates, D., Mächler, M., Bolker, B. y Walker, S. (2014). Fitting linear mixed-effects models using lme4. *arXiv preprint arXiv:1406.5823*.

Bellman, R. (1961). Adaptive control processes: a guided tour.

Bernardo, R. y Yu, J. (2007). Prospects for genomewide selection for quantitative traits in maize. *Crop Science*, 47, 3, 1082–1090.

Brent, R. P. (1973). *Algorithms for Minimization without Derivatives*. Prentice-Hall, Englewood Cliffs, New Jersey, primera edición.

Brooks, S. P. y Gelman, A. (1998). General methods for monitoring convergence of iterative simulations. *Journal of computational and graphical statistics*, 7, 4, 434–455.

Bühlmann, P. y Van De Geer, S. (2011). *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media.

Caballero, J. G., Pablo, E. J. y Martínez, C. C. (2003). Estimación por máxima verosimilitud restringida de componentes de varian-za y covarianza de múltiples características bajo los diseños I y II de Caro-lina del N. *Revista Fitotecnia Mexicana*, 26, 1, 53–66.

Casella, G. y George, E. I. (1992). Explaining the Gibbs sampler. *The American Statistician*, 46, 3, 167–174.

Chen, M.-H., Shao, Q.-M. y Ibrahim, J. G. (2012). *Monte Carlo methods in Bayesian computation*. Springer Science & Business Media.

Chib, S. y Edward, G. (1995). Understanding the Metropolis-Hastings algorithm. *The American Statistician*, 49, 347–335.

Covarrubias-Pazaran, G. (2016). Genome-assisted prediction of quantitative traits using the R package sommer. *PloS one*, 11, 6.

Cowles, M. K. y Carlin, B. P. (1996). Markov chain Monte Carlo convergence diagnostics: a comparative review. *Journal of the American Statistical Association*, 91, 434, 883–904.

Crossa, J., de Los Campos, G., Pérez, P., Gianola, D., Burgueño, J., Araus, J. L., Makumbi, D., Singh, R. P., Dreisigacker, S., Yan, J. *et al.* (2010). Prediction of genetic values of quantitative traits in plant breeding using pedigree and molecular markers. *Genetics*, 186, 2, 713–724.

Crossa, J., Pérez-Rodríguez, P., Cuevas, J., Montesinos-López, O., Jarquín, D., de los Campos, G., Burgueño, J., González-Camacho, J. M., Pérez-Elizalde, S., Beyene, Y. *et al.* (2017). Genomic selection in plant breeding: methods, models, and perspectives. *Trends in plant science*, 22, 11, 961–975.

Cuevas, J., Montesinos-López, O., Juliana, P., Guzmán, C., Pérez-Rodríguez, P., González-Bucio, J., Burgueño, J., Montesinos-López, A. y Crossa, J. (2019). Deep Kernel for Genomic and Near Infrared Predictions in Multi-environment Breeding Trials. *G3 Genes—Genomes—Genetics*, 9, 9, 2913–2924. ISSN 2160-1836.

de los Campos, G., Gianola, D., Rosa, G. J. M., Weigel, K. y Crossa, J. (2010). Semi-parametric Genomic-enabled Prediction of Genetic Values Using Reproducing Kernel Hilbert Spaces Methods. *Genetics Research*, 92, 295–308.

de los Campos, G., Naya, H., Gianola, D., Crossa, J., Legarra, A., Manfredi, E., Weigel, K. y Cotes, J. M. (2009). Predicting quantitative traits with regression models for dense molecular markers and pedigree. *Genetics*, 182, 1, 375–385.

DelSole, T. (2007). A Bayesian framework for multimodel regression. *Journal of climate*, 20, 12, 2810–2826.

Dempster, A. P., Laird, N. M. y Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39, 1, 1–22.

Endelman, J. B. (2011). Ridge regression and other kernels for genomic selection with R package rrBLUP. *Plant Genome*, 4, 250–255.

Fan, J. y Li, R. (2006). Statistical challenges with high dimensionality: Feature selection in knowledge discovery. *arXiv preprint math/0602133*.

Franke, B., Plante, J.-F., Roscher, R., Lee, E.-s. A., Smyth, C., Hatefi, A., Chen, F., Gil, E., Schwing, A., Selvitella, A. *et al.* (2016). Statistical inference, learning and models in big data. *International Statistical Review*, 84, 3, 371–389.

Gamerman, D. y Lopes, H. F. (2006). *Markov chain Monte Carlo: stochastic simulation for Bayesian inference*. Chapman and Hall/CRC, London.

Geisser, S. y Eddy, W. F. (1979). A predictive approach to model selection. *Journal of the American Statistical Association*, 74, 365, 153–160.

Gelfand, A. E. y Smith, A. F. (1990). Sampling based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85, 398–409.

Gelman, A., Carlin, J. B., Stern, H. S. y Rubin, D. B. (2004). *Bayesian Data Analysis*. Chapman and Hall/CRC, London.

Geman, S. y Geman, D. (1984). Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, 6, 721–741.

Geyer, C. J. (1992). Practical markov chain monte carlo. *Statistical science*, 473–483.

Ghosh, J. y Clyde, M. A. (2011). Rao-Blackwellization for Bayesian Variable Selection and Model Averaging in Linear and Binary Regression: A Novel Data Augmentation Approach. *Journal of the American Statistical Association*, 106, 495, 1041–1052.

Gianola, D., de los Campos, G., Hill, W. G., Manfredi, E. y Fernando, R. (2009). Additive Genetic Variability and the Bayesian Alphabet. *Genetics*, 183, 1, 347–363. ISSN 1943-2631.

Gianola, D., Fernando, R. L. y Stella, A. (2006). Genomic-assisted prediction of genetic value with semiparametric procedures. *Genetics*, 173, 3, 1761–1776.

Gilks, W. R., Richardson, S. y Spiegelhalter, D. (1995). *Markov chain Monte Carlo in practice*. CRC press.

Gilmour, A., Cullis, B., Welham, S., Gogel, B. y Thompson, R. (2004). An efficient computing strategy for prediction in mixed linear models. *Computational statistics & data analysis*, 44, 4, 571–586.

Giraud, C. (2015). *Introduction to high-dimensional statistics*. Chapman and Hall/CRC.

Golub, G. y Van Loan, C. (2013). *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press. ISBN 9781421407944.

Gueuning, T. (2017). Statistical inference and focused model selection for high-dimensional regression models.

Gumedze, F. y Dunne, T. (2011). Parameter estimation and inference in the linear mixed model. *Linear Algebra and its Applications*, 435, 8, 1920–1944. ISSN 0024-3795.

Hastie, T. y Tibshirani, R. (1987). Generalized additive models: some applications. *Journal of the American Statistical Association*, 82, 398, 371–386.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57, 97–109.

Haugh, M. (2015). The em algorithm. Accessed Oct. 21, 2021 [Online].

Hayes, B., Bowman, P., Chamberlain, A. y Goddard, M. (2009). Invited review: Genomic selection in dairy cattle: Progress and challenges. *Journal of Dairy Science*, 92, 2, 433–443.

Henderson, C. R. (1950). Estimation of genetic parameters. *Biometrics*, 6, 2, 186–187.

Henderson, C. R. (1953). Estimation of variance and covariance components. *Biometrics*, 9, 2, 226–252.

Hoerl, A. E. y Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12, 1, 55–67.

Hubin, A. y Storvik, G. (2016). Estimating the marginal likelihood with Integrated nested Laplace approximation (INLA). *arXiv preprint arXiv:1611.01450*.

Jarquín, D., Crossa, J., Lacaze, X., Du Cheyron, P., Daucourt, J., Lorgeou, J., Piraux, F., Guerreiro, L., Pérez, P., Calus, M. *et al.* (2014). A reaction norm model for genomic selection using high-dimensional genomic and environmental data. *Theoretical and applied genetics*, 127, 3, 595–607.

Kammann, E. y Wand, M. P. (2003). Geoadditive models. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 52, 1, 1–18.

Kernighan, B. W. y Ritchie, D. M. (1988). *The C Programming Language*. Pearson Education, segunda edición.

Lanczos, C. (1950). An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of Research of the National Bureau of Standards*, 45, 255–282. ISSN 160-1741.

Lee, Y. y Nelder, J. (1998). Generalized linear models for the analysis of quality-improvement experiments. *Canadian Journal of Statistics*, 26, 1, 95–105.

Li, S., Cai, T. T. y Li, H. (2021). Inference for high-dimensional linear mixed-effects models: A quasi-likelihood approach. *Journal of the American Statistical Association*, 1–12.

Lin, X. y Zhang, D. (1999). Inference in generalized additive mixed modelsby using smoothing splines. *Journal of the royal statistical society: Series b (statistical methodology)*, 61, 2, 381–400.

Lindley, D. V. y Smith, A. F. (1972). Bayes estimates for the linear model. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34, 1, 1–18.

Liu, J. S., Liang, F. y Wong, W. H. (2001). A theory for dynamic weighting in Monte Carlo computation. *Journal of the American Statistical Association*, 96, 454, 561–573.

Lopez-Cruz, M., Crossa, J., Bonnett, D., Dreisigacker, S., Poland, J., Jannink, J.-L., Singh, R. P., Autrique, E. y de los Campos, G. (2015). Increased prediction accuracy in wheat breeding trials using a marker x environment interaction genomic selection model. *G3: Genes, Genomes, Genetics*, 5, 4, 569–582.

Marquardt, D. W. y Snee, R. D. (1975). Ridge regression in practice. *The American Statistician*, 29, 1, 3–20.

Marshall, E. y Spiegelhalter, D. (2003). Approximate cross-validatory predictive checks in disease mapping models. *Statistics in medicine*, 22, 10, 1649–1660.

Martino, S. y Riebler, A. (2014). Integrated nested Laplace approximations (INLA). *Wiley StatsRef: Statistics Reference Online*, 1–19.

Martino, S. y Rue, H. (2009). Implementing approximate Bayesian inference using Integrated Nested Laplace Approximation: A manual for the inla program. *Department of Mathematical Sciences, NTNU, Norway*.

Martins, T. G., Simpson, D., Lindgren, F. y Rue, H. (2013). Bayesian computing with INLA: new features. *Computational Statistics & Data Analysis*, 67, 68–83.

McDonald, G. C. (2009). Ridge regression. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1, 1, 93–100.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. y Teller, E. (1953). Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21, 1087–1092.

Meuwissen, T. H. E., Hayes, B. J. y Goddard, M. E. (2001). Prediction of Total Genetic Value Using Genome-Wide Dense Marker Maps. *Genetics*, 157, 4, 1819–1829. ISSN 0016-6731.

Møller, J. (1993). Discussion on the meeting on the Gibbs sampler and other Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society, Series B (Statistical Methodology)*, 55, 84–85.

Montesinos-López, O. A., Montesinos-López, A., Crossa, J., Toledo, F. H., Pérez-Hernández, O., Eskridge, K. M. y Rutkoski, J. (2016). A genomic Bayesian multi-trait and multi-environment model. *G3: Genes, Genomes, Genetics*, 6, 9, 2725–2744.

Moon, T. K. (1996). The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13, 6, 47–60.

Morota, G. y Gianola, D. (2014). Kernel-based whole-genome prediction of complex traits: a review. *Frontiers in genetics*, 5, 363.

Ormerod, J. T. y Wand, M. P. (2010). Explaining variational approximations. *The American Statistician*, 64, 2, 140–153.

Patterson, H. D. y Thompson, R. (1971). Recovery of inter-block information when block sizes are unequal. *Biometrika*, 58, 3, 545–554.

Pearson, K. (1901). Principal components analysis. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 6, 2, 559.

Pérez, P. y de los Campos, G. (2014). Genome-wide regression and prediction with the BGLR statistical package. *Genetics*, 198, 2, 483–495.

Pérez, P., de los Campos, G., Crossa, J. y Gianola, D. (2010). Genomic-enabled prediction based on molecular markers and pedigree using the Bayesian linear regression package in R. *The plant genome*, 3, 2, 106–116.

Pérez-Elizalde, S., Cuevas, J., Pérez-Rodríguez, P. y Crossa, J. (2015). Selection of the Bandwidth Parameter in a Bayesian Kernel Regression Model for Genomic-Enabled Prediction. *Journal of Agricultural, Biological, and Environmental Statistics*, 20, 4, 512–532.

Peskin, P. H. (1973). Optimum Monte-Carlo sampling using Markov chain. *Biometrika*, 60, 607–612.

Pérez-Elizalde, S., Monroy-Castillo, B. E., Pérez-Rodríguez, P. y Crossa, J. (2022). HDBRR: a statistical package for high-dimensional Bayesian ridge regression without MCMC. *Journal of Statistical Computation and Simulation*, 1–27.

R Core Team (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Rao, C. R. (1971). Estimation of variance and covariance components?MINQUE theory. *Journal of multivariate analysis*, 1, 3, 257–275.

Robert, C. P. *et al.* (2007). *The Bayesian choice: from decision-theoretic foundations to computational implementation*, tomo 2. Springer.

Rue, H. y Martino, S. (2007). Approximate Bayesian inference for hierarchical Gaussian Markov random field models. *Journal of statistical planning and inference*, 137, 10, 3177–3192.

Rue, H., Martino, S. y Chopin, N. (2009). Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of the royal statistical society: Series b (statistical methodology)*, 71, 2, 319–392.

Savitzky, A. y Golay, M. J. E. (1964). Smoothing and Differentiation of Data by Simplified Least Squares Procedures. *Analytical Chemistry*, 36, 1627–1639.

Schwarz, G. (1978). Estimating the dimension of a model. *The annals of statistics*, 461–464.

Searle, S., Cassella, G. y McCullouch, C. (1992). Variance Components.–New-York: Jonh Wiley-Sons. *DOI: https://doi. org/10.1002/9780470316856*.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. Chapman & Hall.

Smith, R. L. (1984). Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions. *Operations Research*, 32, 6, 1296–1308.

Sorensen, D. y Gianola, D. (2007). *Likelihood, Bayesian, and MCMC methods in quantitative genetics*. Springer Science & Business Media.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P. y Van Der Linde, A. (2002). Bayesian measures of model complexity and fit. *Journal of the royal statistical society: Series b (statistical methodology)*, 64, 4, 583–639.

Stevens, A. y Ramirez-Lopez, L. (2022). *An introduction to the prospectr package*. R package version 0.2.4.

Tanner, M. A. y Wong, W. H. (1987). The calculation of posterior distributions by data augmentation. *Journal of the American Statistical Association*, 82, 528–549.

Taylor, S. J. (1994). Modeling stochastic volatility: A review and comparative study. *Mathematical finance*, 4, 2, 183–204.

Technow, F., Schrag, T. A., Schipprack, W., Bauer, E., Simianer, H. y Melchinger, A. E. (2014). Genome properties and prospects of genomic prediction of hybrid performance in a breeding program of maize. *Genetics*, 197, 4, 1343–1355.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58, 1, 267–288.

Tierney, L. (1994). Markov chains for exploring posterior distributions. *The Annals of Statistics*, 22, 1701–1762.

Tran, M.-N., Nguyen, T.-N. y Dao, V.-H. (2021). A practical tutorial on Variational Bayes. *arXiv preprint arXiv:2103.01327*.

Uyanik, G. K. y Güler, N. (2013). A Study on Multiple Linear Regression Analysis. *Procedia - Social and Behavioral Sciences*, 106, 234–240. ISSN 1877-0428. 4th International Conference on New Horizons in Education.

van Wieringen, W. N. (2015). Lecture notes on ridge regression. *arXiv preprint arXiv:1509.09169*.

VanRaden, P., Van Tassell, C., Wiggans, G., Sonstegard, T., Schnabel, R., Taylor, J. y Schenkel, F. (2009). Invited review: Reliability of genomic predictions for North American Holstein bulls. *Journal of dairy science*, 92, 1, 16–24.

Vazquez, A., Bates, D., Rosa, G., Gianola, D. y Weigel, K. (2010). an R package for fitting generalized linear mixed models in animal breeding. *Journal of animal science*, 88, 2, 497–504.

Watanabe, S. y Opper, M. (2010). Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. *Journal of machine learning research*, 11, 12.

Weiss, M. A. (2012). *Data Structures & Algorithm Analysis in C++*. Pearson Education, cuarta edición. ISBN 013284737X.

Wimmer, V., Albrecht, T., Auinger, H.-J. y Schön, C.-C. (2012). synbreed: a framework for the analysis of genomic prediction data using R. *Bioinformatics*, 28, 15, 2086–2087.

Wolfinger, R. D., Tobias, R. D. y Sall, J. (1991). Mixed models: a future direction. *SAS Institute Carey*, 1380–1388.

Xiong, S., Dai, B., Huling, J. y Qian, P. Z. G. (2016). Orthogonalizing EM: A Design-Based Least Squares Algorithm. *Technometrics*, 58, 3, 285–293.

Zhao, T., Fernando, R., Garrick, D. y Cheng, H. (2020). Fast parallelized sampling of Bayesian regression models for whole-genome prediction. *Genetics Selection Evolution*, 52.

Zhou, X. y Stephens, M. (2012). Genome-wide efficient mixed-model analysis for association studies. *Nature genetics*, 44, 7, 821–824.

Ziyatdinov, A., Vázquez-Santiago, M., Brunel, H., Martinez-Perez, A., Aschard, H. y Soria, J. M. (2018). lme4qtl: linear mixed models with flexible covariance structure for genetic studies of related individuals. *BMC bioinformatics*, 19, 1, 1–5.

ANEXOS

Anexo A: Código en R para ajustar el modelo BRR con aumentación ortogonal de datos

```
rm(list=ls())

library(mgcv) #For function slanczos

dyn.load("sample_orthogonal.so")

#Fits the model y = mu*1 + X*beta + error using orthogonal data augmentation
#and assuming beta ~ N(0, sigma^2_beta)
BRR_orthogonal<-function(y,X,fraction=0.5,dfb=5,dfe=5,nIter=10000,burnIn=5000,
         thin=10,saveFiles=FALSE)
{
   if(any(is.na(y))) stop("Missing values not allowed in y\n")
   if(any(is.na(X))) stop("Missing values not allowed in X\n")

   n=nrow(X)
   p=ncol(X)

   vy=var(y)

   x2=colSums(X*X)
   sx=colSums(X)
   sumMeanXSq=sum((apply(X,2L,mean))^2)
   MSx=sum(x2)/n-sumMeanXSq
   Sb=fraction*vy*(dfb+2)/MSx
   varB=0.10

   Se=(1-fraction)*vy*(dfe+2)
   varE=var(y)/2

```

```
30    Wo=cbind(1,X)
31    WotWo<-crossprod(Wo)
32
33    #lambda1=eigen(WotWo)$values[1]+0.05
34    lambda1=slanczos(A=WotWo,k=1,kl=-1)$values[1]+0.05
35    D=diag(rep(lambda1,ncol(WotWo)))
36    tmp=D-WotWo
37    L=chol(tmp)
38    q=ncol(L)
39
40    #Pre compute
41    Xty=as.vector(t(X)%*%y)
42    ybarra=mean(y)
43    sumy=sum(y)
44
45    #Initializing yNa, p+1 because we include the intercept
46    yNa<-rep(ybarra,p+1)
47
48    #Initializing beta
49    beta=rep(0,p)
50    Mbeta=matrix(NA,nrow=nIter,ncol=p)
51
52    #Initializing mu
53    mu=ybarra
54    Vmu=rep(NA,nIter)
55
56    #varE
57    VvarE=rep(NA,nIter)
58
59    #varB
60    VvarB=rep(NA,nIter)
61
62    b=c(mu,beta)
63
64    #Initializing RSS, residual sum of squares
65    l1=as.vector(L[,1])
66    RSS=sum(y^2)+sum(yNa^2) + (lambda1*mu^2) - 2*mu*(sumy+sum(l1*yNa))
67
68    MyNa=matrix(NA,nrow=nIter,ncol=p+1)
69
70    for(iter in 1:nIter)
71    {
72       cat("iter=",iter,"\n")
```

```
73
74      .Call("sample_orthogonal",L, b, yNa, Xty, q, varE, varB, lambda1,RSS,sumy)
75
76      MyNa[iter,]<-yNa
77
78      beta<-b[-1]
79      Mbeta[iter,]=beta
80      Vmu[iter]=b[1]
81
82      #Sample varE
83      escala=RSS+Se
84      varE=escala/rchisq(1,df=dfe+n+p)
85      VvarE[iter]=varE
86      cat("varE=",varE,"\n")
87
88      #Sample sigma2beta
89      escala=sum(beta^2) + Sb
90      varB=escala/rchisq(1,df=dfb+p)
91      VvarB[iter]=varB
92      cat("varB=",varB,"\n")
93  }
94
95  if(saveFiles)
96  {
97      write.table(Vmu,file="intercept.dat",row.names=FALSE,col.names=FALSE)
98      write.table(VvarB,file="varB.dat",row.names=FALSE,col.names=FALSE)
99      write.table(VvarE,file="varE.dat",row.names=FALSE,col.names=FALSE)
100 }
101
102 #Burnin, thin, etc
103 index=seq(from=burnIn+thin,to=nIter,by=thin)
104
105 Mbeta=Mbeta[index,]
106 beta=colMeans(Mbeta)
107 SD.beta=apply(Mbeta,2,sd)
108
109 Vmu=Vmu[index]
110 mu=mean(Vmu)
111 SD.mu=sd(Vmu)
112
113 VvarE=VvarE[index]
114 varE=mean(VvarE)
115 SD.varE=sd(VvarE)
```

```
116
117     VvarB=VvarB[index]
118     varB=mean(VvarB)
119     SD.varB=sd(VvarB)
120
121     MyNa=MyNa[index,]
122     yNa=colMeans(MyNa)
123     SD.yNa=apply(MyNa,2,sd)
124
125
126     #return the goodies
127     out=list(beta=beta, SD.beta=SD.beta,
128              mu=mu,SD.mu=SD.mu,
129              varE=varE,
130              SD.varE=SD.varE,
131              varB=varB,
132              SD.varB=SD.varB,
133              yNa=yNa,
134              SD.yNa=SD.yNa)
135 }
136
137 #Example
138 library(BGLR)
139
140 data(mice)
141 X=mice.X[,1:2000]
142 X=scale(X)
143 y=mice.pheno$Obesity.BMI
144 y=as.vector(scale(y))
145 y=y+3
146
147 fm0=BRR_orthogonal(y,X,nIter=50000,burnIn=25000,saveFiles=TRUE)
148 yHat=fm0$mu+as.vector(X%*%fm0$beta)
149 plot(y,yHat)
150
151 eta=list(list(X=X,model="BRR"))
152 fm1=BGLR(y=y,ETA=eta,nIter=50000,burnIn=25000,thin=10)
153 unlink("*.dat")
154 plot(y,fm1$yHat)
155
156 #Comparison
157 plot(fm1$yHat,yHat)
158 cor(fm1$yHat,yHat)
```

```
159
160 plot(fm1$ETA[[1]]$b,fm0$beta)
161 cor(fm1$ETA[[1]]$b,fm0$beta)
```

Anexo B: Código en C para ajustar el modelo BRR con aumentación ortogonal de datos

El siguiente código de programación es utilizado para hacer el ajuste de un modelo BRR con aumentación ortogonal de datos. Se presenta el código en lenguaje de programación `C` (Kernighan y Ritchie, 1988) que se utilizó para muestrear los parámetros $\mu|resto$, $\beta_j|resto$ e imputar los valores faltantes para la variable respuesta ($\boldsymbol{y}_a$).

```
/*
File:  sample_orthogonal.c

Compilation in Windows:

R CMD SHLIB sample_orthogonal.c -lRlapack -lRblas

Compilation in UNIX (macOS, Linux):

R CMD SHLIB sample_orthogonal.c
*/


#include <R.h>
#include <Rmath.h>
#include <Rinternals.h>
#include <Rdefines.h>
#include <Rconfig.h>
#include <R_ext/Lapack.h>

/*
L includes the intercept column and has ncol given in the
arguments, b=(mu,beta_1,...,beta_p)'
*/

SEXP sample_orthogonal(SEXP L, SEXP b, SEXP yNa, SEXP Xy, SEXP ncol,
                       SEXP varE, SEXP varB, SEXP lambda1,SEXP RSS, SEXP sumy)
{
  double *pL;
  double *pb;
  double *pXy;
  double *pRSS;
  double *pyNa;
  double yNaOld;
  double bOld;
```

```
36
37  double sigma2E;
38  double sigmaE;
39  double sigma2Beta;
40  double denominator;
41  double lambda;
42  double s;
43  double mk;
44  double sy;
45
46  int q;
47  int k;
48  int inc=1;
49
50  PROTECT(L=AS_NUMERIC(L));
51  pL=NUMERIC_POINTER(L);
52
53  PROTECT(Xy=AS_NUMERIC(Xy));
54  pXy=NUMERIC_POINTER(Xy);
55
56  PROTECT(b=AS_NUMERIC(b));
57  pb=NUMERIC_POINTER(b);
58
59  PROTECT(yNa=AS_NUMERIC(yNa));
60  pyNa=NUMERIC_POINTER(yNa);
61
62  PROTECT(RSS=AS_NUMERIC(RSS));
63  pRSS=NUMERIC_POINTER(RSS);
64
65  q=INTEGER_VALUE(ncol);
66
67  sigma2E=NUMERIC_VALUE(varE);
68  sigmaE=sqrt(sigma2E);
69  sigma2Beta=NUMERIC_VALUE(varB);
70  lambda=NUMERIC_VALUE(lambda1);
71  sy=NUMERIC_VALUE(sumy);
72
73  GetRNGstate();
74
75
76  double *g;
77  g=malloc(q*sizeof(double));
78
```

```
79  double *tmp;
80  tmp=malloc(q*sizeof(double));
81
82  memcpy(g,pb,q*sizeof(double));
83
84  /*
85  Compute g=L*b
86  */
87  F77_CALL(dtrmv)("U","N","N",&q,pL,&q,g,&inc);
88
89  /*
90  Sample yNa
91  */
92  for(k=0; k<q;k++)
93  {
94    yNaOld=pyNa[k];
95    pyNa[k]=g[k]+sigmaE*norm_rand();
96    pRSS[0]+=pyNa[k]*pyNa[k]-yNaOld*yNaOld-2*g[k]*(pyNa[k]-yNaOld);
97  }
98
99  memcpy(tmp,pyNa,q*sizeof(double));
100
101 /*
102 Compute tmp=L'*yNa
103 */
104 F77_CALL(dtrmv)("U","T","N",&q,pL,&q,tmp,&inc);
105
106 /*
107 Sample beta
108 */
109 denominator=lambda+sigma2E/sigma2Beta;
110 s=sqrt(sigma2E/denominator);
111
112 /*
113 Skip one, the intercept... substact -1 when indexing pXy because
114 the number of elements corresponds to the columns of the original matrix of
115 predictors without including the intercept
116 */
117 for(k=1; k<q;k++)
118 {
119   bOld=pb[k];
120   mk=pXy[k-1]+tmp[k];
121   pb[k]=mk/denominator+s*norm_rand();
```

```
122     pRSS[0]+=(pow(pb[k],2) - pow(bOld,2))*lambda-2*(pb[k]-bOld)*mk;
123   }
124
125   /*
126   Intercept
127   */
128   s=sqrt(sigma2E/lambda);
129   bOld=pb[0];
130   mk=sy+tmp[0];
131   pb[0]=mk/lambda+s*norm_rand();
132   pRSS[0]+=(pow(pb[0],2) - pow(bOld,2))*lambda  -2*(pb[0]-bOld)*mk;
133
134   free(g);
135   free(tmp);
136
137   PutRNGstate();
138
139   UNPROTECT(5);
140
141   return(R_NilValue);
142 }
```

Anexo C: Código en R para ajustar el modelo BRR con aumentación ortogonal de datos con EM

```
rm(list=ls())

library(mgcv) #For function slanczos

#Equation (5), Zhou and Stephens
dloglik=function(lambda,n,v_y,V,d)
{

   #Trace of Hinv*G, Optimized
   Tr_Hinv=sum(1/(lambda*d+1))
   Tr_Hinv_G=(n-Tr_Hinv)/lambda

   dbar=1/(lambda*d+1)
   tmp1=t(v_y*dbar)%*%V
   tmp2=solve(t(V)%*%diag(1/(lambda*d+1))%*%V)
   ty_P_y=sum(v_y^2/(lambda*d+1))-tmp1%*%tmp2%*%t(tmp1)
   ty_P_y=as.numeric(ty_P_y)

   #Check this, is suboptimal
   tmp3=sum(v_y^2*dbar^2)
   tmp4=t(v_y*dbar^2)%*%V%*%tmp2%*%t(tmp1)
   tmp5=as.vector(t(v_y*dbar)%*%V%*%tmp2%*%t(V))*dbar
   tmp6=sum(tmp5^2)

   ty_P_P_y = as.numeric(tmp3-2*tmp4+tmp6)

   ty_P_G_P_y=(ty_P_y-ty_P_P_y)/lambda

   dL1=-0.5*Tr_Hinv_G + 0.5*n * ty_P_G_P_y/(ty_P_y)
   dL1=as.numeric(dL1)

   return(dL1)
}

f_ty_P_y=function(lambda,n,v_y,V,d)
{
   dbar=1/(lambda*d+1)
   tmp1=t(v_y*dbar)%*%V
   tmp2=solve(t(V)%*%diag(1/(lambda*d+1))%*%V)
   ans=sum(v_y^2/(lambda*d+1))-tmp1%*%tmp2%*%t(tmp1)
```

```
41    ans=as.numeric(ans)
42    return(ans)
43 }
44
45 #Originally we want to fit
46 #y = mu*1 + X*beta + error, with X matrix of markers and beta ~ N(0, sigma^2_beta)
47 #Let u=X*beta ~ N(0,\sigma^2_beta G, where G is a genomic relationship matrix)
48 #The model is then equivalent to:
49 #y = mu*1 + u + e = mu*1 + Gamma*Lambda^1/2 u* + e, with u* ~ N(0, \sigma^2_\beta*I)
50 #                 = mu*1 + Zs * u* + e
51
52 #This function fits y=mu*1 + u + e
53 EM_orthogonal=function(y,G,maxiter_varpar=100,maxiter_em=10000, tol=1E-4)
54 {
55    eigen_G=eigen(G,symm=TRUE)
56
57    d=eigen_G$values
58    U=eigen_G$vectors
59
60    Zs=U%*%sqrt(diag(d))
61
62    v_y=as.vector(crossprod(U,y))
63    V=crossprod(U,matrix(rep(1,length(y))))
64
65    maxiter=100
66
67    sol_uniroot=uniroot(f=dloglik,interval=c(1e-5,1e5),
68                        n=length(y),v_y=v_y,V=V,d=d,maxiter=maxiter_varpar)
69
70    if(sol_uniroot$iter<maxiter_varpar)
71    {
72       lambda=sol_uniroot$root
73       dbar=1/(lambda*d+1)
74       cat("Brent's algorithm converged!\n")
75       ty_P_y=f_ty_P_y(lambda,v_y=v_y,V=V,d=d)
76       varE=ty_P_y/length(y)
77       varU=lambda*varE
78       cat("varE=",varE,"\n")
79       cat("varU=",varU,"\n")
80
81    }else{
82       stop("Brent's algorithm did not converge!\n")
83    }
```

```
84
85    ## Orthogonal transformation ##
86
87    Wo=cbind(1,Zs)
88
89    WotWo=crossprod(Wo)
90
91    #lambda1=eigen(WotWo)$values[1]+0.05
92    lambda1=slanczos(A=WotWo,k=1,kl=-1)$values[1]+0.05
93    D=diag(rep(lambda1,ncol(WotWo)))
94    tmp=D-WotWo
95    L=chol(tmp)
96
97    ## Orthogonal transformation ##
98    Wc=rbind(Wo,L)
99
100   XFixed=as.matrix(Wc[,1])
101   Zr=Wc[,-1]
102
103   nNa=nrow(Wc)-length(y)
104
105   #Bad initial values, is there a better way?
106   yNa=rep(mean(y),nNa)
107
108   yc=c(y,yNa)
109
110   beta=mean(y)
111
112   #See Sorensen and Gianola, Example 9.3 Maximum likelihood estimation in the mixed
       linear model
113   k=varE/varU
114
115   error=1
116   iter=0
117
118   betaOld=Inf
119   uStarOld=rep(Inf,ncol(Zr))
120   error_history=rep(NA,maxiter_em)
121
122   while(error>tol & iter<maxiter_em)
123   {
124      iter=iter+1
125
```

```
126       #update uStar
127       uStar=as.vector(1.0/(lambda1+k)*t(Zr)%*%(yc-XFixed%*%beta))
128
129       #update beta
130       ytilde=yc-Zr%*%uStar
131       beta=solve(crossprod(XFixed))%*%t(XFixed)%*%ytilde
132
133       #update yNa
134       yNa=as.vector(L%*%c(beta,uStar))
135       yc=c(y,yNa)
136
137       error=max(abs(c(betaOld,uStarOld)-c(beta,uStar)))
138       error_history[iter]=error
139       betaOld=beta
140       uStarOld=uStar
141
142       cat("iter=",iter,"Error=",error,"\n")
143    }
144
145
146    #Obtain u
147    u=as.vector(Zs%*%uStar)
148
149    #Obtain yHat
150    yHat=as.vector(beta)+u
151
152    #Solo para graficar...
153    error_history[1]=0.2 #Debe ser no definido
154    error_history=error_history[1:iter]
155
156    return(list(mu=as.vector(beta),yHat=yHat,u=u,varE=varE,varU=varU,
157                yNa=yNa,error_history=error_history))
158 }
159
160 #Example
161 library(BGLR)
162
163 data(mice)
164 X=mice.X
165 y=mice.pheno$Obesity.BMI
166 y=as.vector(scale(y))
167
168 X=scale(X,center=TRUE,scale=TRUE)
```

```
G=tcrossprod(X)/ncol(X)

fm0=EM_orthogonal(y,G,tol=1E-4)

eta=list(list(K=G,model="RKHS"))
fm1=BGLR(y=y,ETA=eta,nIter=10000,burnIn=5000)
unlink("*.dat")

plot(y,fm0$yHat)
plot(y,fm1$yHat)
plot(fm0$yHat,fm1$yHat)

plot(fm0$u,fm1$ETA[[1]]$u)
cor(fm0$u,fm1$ETA[[1]]$u)

#Alternative
eigen_G=eigen(G,symm=TRUE)
Zs=eigen_G$vectors%*%sqrt(diag(eigen_G$values))
eta=list(list(X=Zs,model="BRR"))
fm2=BGLR(y=y,ETA=eta,nIter=10000,burnIn=5000)
unlink("*.dat")

u=as.vector(Zs%*%fm2$ETA[[1]]$b)
plot(fm0$u,u)
cor(fm0$u,u)
```