



COLEGIO DE POSTGRADUADOS

INSTITUCIÓN DE ENSEÑANZA E INVESTIGACIÓN
EN CIENCIAS AGRÍCOLAS

CAMPUS MONTECILLO

SOCIOECONOMÍA, ESTADÍSTICA E INFORMÁTICA
ESTADÍSTICA

**Aplicación del Elastic Net LASSO y modelos
relacionados en Selección Genómica basados en
Marcadores Moleculares**

Marco Antonio López Cruz

T E S I S

PRESENTADA COMO REQUISITO PARCIAL PARA
OBTENER EL GRADO DE:

MAESTRO EN CIENCIAS

MONTECILLO, TEXCOCO, EDO. DE MÉXICO
2012

La presente tesis titulada: **Aplicación del Elastic Net LASSO y modelos relacionados en Selección Genómica basados en Marcadores Moleculares**, realizada por el alumno: **Marco Antonio López Cruz**, bajo la dirección del Consejo Particular indicado ha sido aprobada por el mismo y aceptada como requisito parcial para obtener el grado de:

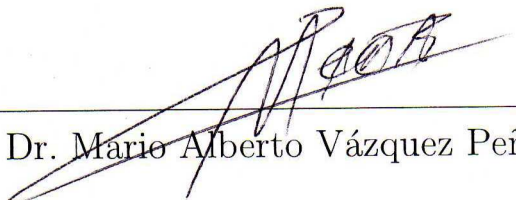
MAESTRO EN CIENCIAS

SOCIOECONOMÍA, ESTADÍSTICA E INFORMÁTICA ESTADÍSTICA

CONSEJO PARTICULAR

CONSEJERO Pérez Rdz.
Dr. Paulino Pérez Rodríguez

ASESOR 
Dr. Eduardo Gutiérrez González

ASESOR 
Dr. Mario Alberto Vázquez Peña

Aplicación del Elastic Net LASSO y modelos relacionados en Selección Genómica basados en Marcadores Moleculares

Marco Antonio López Cruz

Colegio de Postgraduados, 2012

El Elastic Net Bayesiano (BEN) es un método de regresión que utiliza una mezcla de las penalizaciones L_1 y L_2 (Kyung *et al.*, 2010, Li y Lin, 2010). Se ha demostrado que este modelo puede ser usado exitosamente cuando el tamaño de muestra es mucho menor que el número de predictores ($n \ll p$). En este trabajo se muestra cómo utilizar este modelo para incluir de forma conjunta Marcadores Moleculares (MM) y Pedigree, ampliamente utilizados en genética cuantitativa en la llamada selección asistida por MM. Por medio de validación cruzada, el poder predictivo del BEN se compara con el de otros modelos: LASSO Bayesiano y Regresión Ridge Bayesiana, usando datos reales de rendimiento de cultivares de trigo y cebada, y tiempos de floración de maíz. Los resultados muestran que el BEN tiene un poder predictivo igual o superior que el resto de los modelos mencionados.

Palabras clave: Validación cruzada, Regresión penalizada, predicción de valores genéticos.

Application of Bayesian Elastic Net and Related Methods in Genomic Selection based on Molecular Markers

Marco Antonio López Cruz

Colegio de Postgraduados, 2012

The Bayesian Elastic Net (BEN) is a regression method that uses a mixture of L_1 and L_2 penalties (Kyung *et al.*, 2010, Li y Lin, 2010). It has been shown that this model can be used successfully when the sample size is much smaller than the number of predictors ($n \ll p$). This paper shows how to use this model to include jointly Molecular Markers (MM) and Pedigree, widely used in quantitative genetics in the called MM-assisted selection. Through cross-validation, the predictive power of BEN is compared with other models: Bayesian LASSO and Bayesian Ridge Regression, using real data of yield of cultivars wheat and barley and flowering time of maize. The results show that BEN has a predictive power equal to or greater than the rest of the models.

Key words: Cross-validation, penalized regression, prediction of genetic values.

AGRADECIMIENTOS

Al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico brindado, muy necesario para mi manutención durante la realización de mis estudios.

Al Colegio de Postgraduados, por haberme brindado la oportunidad de seguir mi formación académica en sus aulas.

A los integrantes de mi Consejo Particular:

Al Dr. Paulino Pérez Rodríguez, porque desde un inicio confió en mi persona y por compartir y transmitirme de manera desinteresada sus conocimientos adquiridos.

Al Dr. Eduardo Gutiérrez Gonzalez por su orientación, apoyo y colaboración desinteresada en el presente trabajo.

Al Dr. Mario Alberto Vazquez Peña, por dedicar parte de su tiempo en la revisión de este trabajo de tesis.

A quienes fueron mis profesores y pusieron en mis manos herramientas valiosas que utilizaré en mi quehacer profesional.

A mis compañeros de clase y además amigos, quienes siempre brindaron apoyo.

A todas aquellas personas que no mencioné, pero de alguna manera influyeron en mi formación, a todos gracias.

DEDICATORIA

A mis primeros maestros y además... padres: Jose Antonio y Amparo.

A mis mejores compañeros en esta escuela de la vida y además ... hermanos: Luis Antonio, Vlady Jazmin y Juan Manuel.

A mi amiga, compañera y novia: Mayra Velen.

Índice

1. Introducción	1
2. Objetivos	2
2.1. Objetivos Generales	2
2.2. Objetivos particulares	2
3. Marco Teórico	3
3.1. Problemas en regresión cuando $p \gg n$	3
3.2. Métodos de Regresión Penalizada	4
3.2.1. Regresión Ridge	4
3.2.2. Regresión LASSO	5
3.2.3. Elastic Net	5
3.2.4. Regresión Ridge Bayesiana	6
3.2.5. LASSO Bayesiano	6
3.2.6. Elastic Net Bayesiano	7
4. BEN mixto	9
4.1. BEN más pedigree	9

Índice

4.2. Simulación	10
4.2.1. Distribuciones a priori y distribución a posteriori	10
4.2.2. Distribuciones condicionales	12
4.2.3. Muestreo de Gibbs	14
4.2.4. Selección de los parámetros de penalidad λ_1 y λ_2	15
4.2.5. Validación cruzada	18
5. Aplicación del BEN mixto en Selección Genómica	19
5.1. Datos de cebada (<i>Hordeum vulgare</i>)	20
5.2. Datos de trigo (<i>Triticum aestivum</i>)	20
5.3. Datos de maíz (<i>Zea mays</i>)	20
6. Resultados y Discusión	21
7. Conclusiones	39
Referencias	39
Anexos	42
Anexo A: Prueba del lema 1	42
Anexo B: Rutinas en R-2.12.2 para implementar el Modelo Elastic Net con o sin pedigree	44
Anexo C: Rutinas en R-2.12.2 para implementar el algoritmo EM Monte Carlo de Levine y Casella	53
Anexo D: Rutinas en lenguaje C para implementar el muestreo de los parámetros β y u	56

Índice de tablas

6.1. Correlaciones obtenidas con cada uno de los 3 modelos usando validación cruzada con 10 folds para los datos de cebada	22
6.2. Correlaciones obtenidas con cada uno de los 6 modelos usando validación cruzada con 10 folds para los datos de trigo	22
6.3. Correlaciones obtenidas con cada uno de los 3 modelos usando validación cruzada con 10 folds para los datos de maíz	22
6.4. Algunos parámetros obtenidos con cada uno de los modelos completos para los tres conjuntos de datos.	23

Índice de figuras

3.1. Formas de la a priori de β para distintos valores de λ_1 y λ_2	8
6.1. Comparación de los estimadores de los efectos usando el modelo M-BEN contra los obtenidos usando M-BL y M-BRR para los datos de cebada. Errores en cada iteración EM Monte Carlo con $\delta_1 = 0.001$ y $\delta_2 = 0.01$ para el modelo M-BEN para los datos de cebada.	24
6.2. Comparación de los efectos y heredabilidades de los MM obtenidos con cada uno de los modelos para los datos de cebada.	25
6.3. Comparación de los estimadores de los efectos usando los modelos M-BEN y PM-BEN contra los obtenidos usando M-BL y M-BRR con y sin pedigree para los datos de trigo en el ambiente 1.	26
6.4. Comparación de los estimadores de los efectos usando los modelos M-BEN y PM-BEN contra los obtenidos usando M-BL y M-BRR con y sin pedigree para los datos de trigo en el ambiente 2.	27
6.5. Comparación de los estimadores de los efectos usando los modelos M-BEN y PM-BEN contra los obtenidos usando M-BL y M-BRR con y sin pedigree para los datos de trigo en el ambiente 3.	28
6.6. Comparación de los estimadores de los efectos usando los modelos M-BEN y PM-BEN contra los obtenidos usando M-BL y M-BRR con y sin pedigree para los datos de trigo en el ambiente 4.	29
6.7. Errores en cada iteración EM Monte Carlo con $\delta_1 = 0.001$ y $\delta_2 = 0.01$ para el modelo M-BEN y PM-BEN usando los datos de trigo para cada uno de los 4 ambientes.	30

6.8. Comparación de los efectos y heredabilidades de los MM obtenidos con cada uno de los modelos sin pedigree para los datos de trigo en el ambiente 1.	31
6.9. Comparación de los efectos y heredabilidades de los MM obtenidos con cada uno de los modelos sin pedigree para los datos de trigo en el ambiente 2.	32
6.10. Comparación de los estimadores de los efectos usando los modelos M-BEN contra los obtenidos usando M-BL y M-BRR usando los datos de maíz para los 3 fenotipos de la población 1.	33
6.11. Comparación de los estimadores de los efectos usando los modelos M-BEN contra los obtenidos usando M-BL y M-BRR usando los datos de maíz para los 3 fenotipos de la población 2.	34
6.12. Errores en cada iteración EM Monte Carlo con $\delta_1 = 0.001$ y $\delta_2 = 0.01$ para el modelo M-BEN usando los datos de maíz para los 3 fenotipos de las poblaciones 1 y 2.	35
6.13. Comparación de los efectos y heredabilidades de los MM obtenidos con cada uno de los modelos para el fenotipo DTA de los datos de maíz en la población 1.	36
6.14. Comparación de los efectos y heredabilidades de los MM obtenidos con cada uno de los modelos para el fenotipo DTS de los datos de maíz en la población 1.	37
6.15. Comparación de los efectos y heredabilidades de los MM obtenidos con cada uno de los modelos para el fenotipo ASI de los datos de maíz en la población 1.	38

Capítulo 1

Introducción

Los métodos de regresión penalizada son utilizados cuando se presenta el problema de multicolinealidad, y más aún, en el caso de que el número de predictores es mucho mayor que el tamaño de la muestra ($p \gg n$). A este tipo de métodos pertenecen la regresión Ridge (Hoerl y Kennard, 1970) que usa la suma de cuadrados de los parámetros β (o norma L_2) como penalización, y la regresión LASSO (Tibshirani, 1996) cuya penalización es la suma de los valores absolutos de los parámetros β (o norma L_1), además de estos, existe el llamado Elastic Net (Zou y Hastie, 2005) cuya penalización es una mezcla entre las normas L_1 y L_2 . Comúnmente los métodos de regresión penalizada pueden estudiarse desde el punto de vista Bayesiano asignando una distribución a priori a los coeficientes de los predictores: la regresión Ridge Bayesiana (BRR, con una a priori Normal), el LASSO Bayesiano (BL, con una a priori Laplace o Doble Exponencial) y el Elastic Net Bayesiano (BEN, con una a priori de una mezcla entre las distribuciones Normal y Laplace).

En el presente trabajo se aplican el BEN, el BL y la BRR en la llamada Selección Genómica (SG) perteneciente a la Genética Cuantitativa. Los modelos incluirán Marcadores Moleculares (MM) que harán el papel de las variables predictoras, en este contexto se les llamará M-BEN, M-BL y M-BRR. En el Capítulo 4 se presenta una idea sobre como extender el M-BEN agregándole información contenida en un pedigree (PM-BEN). En el mismo capítulo, se detalla cómo hacer obtener valores de la distribución a posteriori de los parámetros del M-BEN/PM-BEN mediante simulación proponiéndoles a priori adecuadas. La simulación se hace por medio del muestreador de Gibbs (Casella y George, 1992) con ayuda de una modificación ligera del algoritmo EM Monte Carlo, programadas en R apoyándose en alguna parte del lenguaje C. En el Capítulo 6 se muestran los resultados obtenidos con Validaciones Cruzadas (método usado para evaluar el poder de predicción y comparar modelos), también se hace una comparación de los estimadores de β para el M-BEN/PM-BEN con cada uno de los modelos. Los datos que se usaron fueron obtenidos de experimentos sobre el rendimiento de cultivares de trigo y cebada, y tiempos de floración de maíz.

Capítulo 2

Objetivos

2.1. Objetivos Generales

- Desarrollar una metodología que permita el uso de forma conjunta de Marcadores Moleculares y pedigree con el modelo BEN.
- Demostrar con datos reales que el modelo BEN es un método eficiente de predicción y selección de variables cuando se tiene el problema $p \gg n$.

2.2. Objetivos particulares

- Verificar que el BEN se desempeña igual o mejor que la regresión Ridge y LASSO.
- Comprobar que la introducción de un pedigree en el BEN, lleva a una mejor predicción.

Capítulo 3

Marco Teórico

3.1. Problemas en regresión cuando $p \gg n$

En estadística, una regresión es un modelo paramétrico de la forma $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, donde $\mathbf{y} = (y_1, \dots, y_n)^T$ es un vector de variables respuesta, \mathbf{X} es la matriz de predictores de $n \times p$, donde n es el tamaño de muestra y p el número de predictores, $\boldsymbol{\beta} = (\beta_1, \dots, \beta_p)^T$ es un vector de parámetros desconocidos que representan la contribución de cada covariable a la variabilidad de la variable respuesta y $\boldsymbol{\varepsilon} = (\varepsilon_1, \dots, \varepsilon_n)^T$ es un vector de términos de error aleatorio de los que se supone que $\varepsilon_i \sim NIID(0, \sigma^2)$, $i = 1, \dots, n$. Matricialmente, se tiene:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

El método de mínimos cuadrados ordinarios (MCO) es una forma común de estimar coeficientes de regresión $\boldsymbol{\beta}$, sin embargo, tiene dos deficiencias: 1) Aunque un estimador de MCO proporciona una estimación insesgada de los coeficientes de regresión, a menudo posee varianza grande, el cual causará que los valores predichos no sean tan precisos. 2) No es posible obtener el estimador de MCO en situaciones cuando el número de variables explicatorias es más grande que el número de observaciones $p \gg n$. Este segundo punto se debe a que en los estimadores de MCO aparece el término $(\mathbf{X}^T \mathbf{X})^{-1}$, el cual no es posible calcular exactamente debido a que el producto $\mathbf{X}^T \mathbf{X}$ es singular, es decir, su determinante es cero y no se puede invertir.

Además de esto, cuando ocurre el problema $p \gg n$, es más probable que exista colinea-

3.2. Métodos de Regresión Penalizada

alidad o multicolinealidad entre las covariables, es decir, que alguna de ellas sea función de un conjunto del resto de ellas, lo que causará que los coeficientes de regresión posean grandes errores estándar y por lo tanto, que no puedan ser estimados con gran precisión.

3.2. Métodos de Regresión Penalizada

Los métodos de regresión penalizada consisten en agregar términos a la suma de cuadrados, esto con la finalidad de contraer a los estimadores de los coeficientes para reducir sus varianzas y tener una buena precisión en la predicción de valores (Kyung *et al.*, 2010). A este tipo de métodos pertenecen la regresión Ridge (Hoerl y Kennard, 1970), el LASSO (Tibshirani, 1996) y el llamado Elastic Net (Zou y Hastie, 2005). Con estos dos últimos, es posible realizar predicción y selección de variables simultáneamente, ya que el primero solamente efectúa la predicción.

3.2.1. Regresión Ridge

La regresión Ridge, RR (Hoerl y Kennard, 1970) es el método más antiguo de estimación penalizada. En RR, los estimadores de los coeficientes de regresión se obtienen mediante la ecuación $\hat{\boldsymbol{\beta}}_{RR} = (\mathbf{X}^T \mathbf{X} + \lambda_{RR} \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$, donde $\lambda_{RR} \geq 0$, o lo que es lo mismo, minimizando la suma de cuadrados residuales, $RSS(\mathbf{y}, \boldsymbol{\beta}) = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$, sujeto a las siguientes restricciones: $SS(\boldsymbol{\beta}) = \sum_{j=1}^p \beta_j^2 \leq t$; o equivalentemente:

$$\hat{\boldsymbol{\beta}}_{RR} = \min_{\boldsymbol{\beta}} \left\{ (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) + \lambda_{RR} \sum_{j=1}^p \beta_j^2 \right\}, \quad (3.1)$$

donde $\lambda_{RR} = \lambda(t) \geq 0$ es una constante no negativa que controla la compensación entre la bondad de ajuste del modelo, medida por la RSS, y la complejidad del modelo, medida por la suma de cuadrados de los coeficientes de regresión, o la norma L_2 de $\boldsymbol{\beta}$. La penalidad cuadrática L_2 induce a una contracción hacia cero de los coeficientes de regresión; esto introduce sesgo pero reduce la varianza de los estimadores (Zou y Hastie, 2005).

Zou y Hastie (2005) mencionan que la regresión Ridge no proporciona un modelo parsimonioso, ya que siempre mantiene todas las variables en el modelo, es decir, la RR no realiza selección de variables.

3.2. Métodos de Regresión Penalizada

3.2.2. Regresión LASSO

El LASSO (Least Absolute Shrinkage and Selection Operator) de [Tibshirani \(1996\)](#) es otro método de estimación penalizada comúnmente usado. A diferencia de la regresión Ridge, LASSO contiene la suma de los valores absolutos de los coeficientes de regresión, o norma L_1 , esto es:

$$\hat{\beta}_L = \min_{\beta} \left\{ (\mathbf{y} - \mathbf{X}\beta)^T (\mathbf{y} - \mathbf{X}\beta) + \lambda_L \sum_{j=1}^p |\beta_j| \right\}. \quad (3.2)$$

La solución a este problema involucra reducir exactamente a cero algunos coeficientes, por lo tanto, LASSO realiza selección de variable y contracción simultáneamente.

Para resolver el problema del LASSO en (3.2), [Efrom, Hastie, Johnstone y Tibshirani \(2004\)](#) proponen el algoritmo LARS (Least Angle Regression), un procedimiento que contiene una fórmula matemática simple para acelerar los cálculos. LARS es un algoritmo de selección de modelos, sin embargo, [Efrom *et al.* \(2004\)](#) mostraron que con una ligera modificación, el LARS implementa el LASSO.

3.2.3. Elastic Net

[Zou y Hastie \(2005\)](#), señalan que LASSO presenta limitaciones en los siguientes escenarios:

1. En el caso de que $p > n$, LASSO selecciona a lo más n variables debido a que es un problema de optimización convexa.
2. Si existe algún grupo de covariables entre las cuales las correlaciones por pares son muy altas, entonces LASSO tiende a seleccionar solamente una variable del grupo sin importar cual de ellas es seleccionada.
3. En situaciones usuales $n > p$, LASSO se desempeña insatisfactoriamente cuando los predictores están correlacionados y en esta situación, es superado por la regresión Ridge ([Tibshirani, 1996](#)).

Para mejorar el desempeño de LASSO, [Zou y Hastie \(2005\)](#) proponen un método de regresión penalizada llamado Elastic Net (EN), cuya función de penalidad usa una combinación de las normas L_1 y L_2 . En el EN, los problemas de optimización se convierten en:

$$\hat{\beta}_{EN} = \min_{\beta} \left\{ (\mathbf{y} - \mathbf{X}\beta)^T (\mathbf{y} - \mathbf{X}\beta) + \lambda_1 \sum_{j=1}^p |\beta_j| + \lambda_2 \sum_{j=1}^p \beta_j^2 \right\}, \quad (3.3)$$

3.2. Métodos de Regresión Penalizada

donde λ_1 y λ_2 son constantes no negativas, las cuales controlan el peso asignado a las penalidades L_1 y L_2 respectivamente. Este problema de optimización es equivalente a:

$$\min_{\boldsymbol{\beta}} \left\{ (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \right\} \text{ sujeto a : } (1 - \alpha) \sum_{j=1}^p |\beta_j| + \alpha \sum_{j=1}^p \beta_j^2 \leq t,$$

donde t es una constante positiva arbitraria y $\alpha = \lambda_2 / (\lambda_1 + \lambda_2)$.

Para resolver el problema del Elastic Net de una manera eficiente, [Zou y Hastie \(2005\)](#) proponen el algoritmo LARS-EN basado en el algoritmo LARS de [Efron *et al.* \(2004\)](#).

3.2.4. Regresión Ridge Bayesiana

La regresión Ridge Bayesiana (BRR) se obtiene al observar la ecuación (3.1), desde el punto de vista Bayesiano corresponde a la moda a posteriori cuando los parámetros $\boldsymbol{\beta}$ tienen distribución independientemente Normal, esto es:

$$p(\boldsymbol{\beta} | \sigma_{\beta}^2) = \prod_{j=1}^p \frac{1}{\sqrt{2\pi\sigma_{\beta}^2}} \exp \left\{ -\frac{\beta_j^2}{2\sigma_{\beta}^2} \right\},$$

donde σ_{β}^2 es la varianza común a priori de los efectos. Con esta a priori, la media y moda a posteriori de $\boldsymbol{\beta}$ es igual a la obtenida en (3.1) cuando $\lambda_{RR} = \sigma_{\beta}^2 / \sigma_{\epsilon}^2$ ([Pérez *et al.*, 2010](#)).

3.2.5. LASSO Bayesiano

Observando en (3.2) la forma del término de penalidad en el modelo LASSO, [Tibshirani \(1996\)](#) sugiere que los estimadores de $\boldsymbol{\beta}$ se pueden interpretar como la moda a posteriori cuando los parámetros se distribuyen independiente e idénticamente Laplace. Así, [Park y Casella \(2008\)](#) proponen un enfoque Bayesiano para el LASSO (LASSO Bayesiano, BL) usando una a priori condicional Laplace para $\boldsymbol{\beta}$ de la forma:

$$p(\boldsymbol{\beta} | \sigma_{\epsilon}^2) = \prod_{j=1}^p \left(\frac{\lambda}{2\sigma_{\epsilon}} \right) \exp \left\{ -\frac{\lambda|\beta_j|}{\sigma_{\epsilon}} \right\} \quad (3.4)$$

y una a priori marginal no informativa $p(\sigma_{\epsilon}^2) \propto 1/\sigma_{\epsilon}^2$ sobre σ_{ϵ}^2 . Sin embargo, dado que no es posible resolver el problema original de manera directa usando un muestreador de

3.2. Métodos de Regresión Penalizada

Gibbs, [Park y Casella \(2008\)](#) proponen el siguiente modelo jerárquico:

$$\begin{aligned}\boldsymbol{\beta} \mid \sigma_\varepsilon^2, \tau_1^2, \dots, \tau_p^2 &\sim N_p(\mathbf{0}_p, \sigma_\varepsilon^2 D_\tau) \\ \sigma_\varepsilon^2, \tau_1^2, \dots, \tau_p^2 &\sim p(\sigma_\varepsilon^2) d\sigma_\varepsilon^2 \prod_{j=1}^p \frac{\lambda^2}{2} \exp\{-\lambda^2 \tau_j^2 / 2\} d\tau_j^2 \\ \sigma_\varepsilon^2, \tau_1^2, \dots, \tau_p^2 &> 0,\end{aligned}$$

donde $D_\tau = \text{diag}(\tau_1, \dots, \tau_p)$. Usando esta representación se tiene un modelo jerárquico a partir del cual se obtiene el muestreador de Gibbs.

Para resolver computacionalmente los métodos Bayesianos BRR y BL, [Pérez, de los Campos, Crossa y Gianola \(2010\)](#) elaboraron un paquete en R ([R Development Core Team, 2011](#)) llamado BLR (Bayesian Linear Regression), el cual implementa un muestreo de Gibbs con las a priori respectivas.

3.2.6. Elastic Net Bayesiano

[Li y Lin \(2010\)](#) mencionan que el problema del EN es equivalente a encontrar la moda de la marginal a posteriori de $\boldsymbol{\beta} \mid \mathbf{y}$ cuando la a priori de $\boldsymbol{\beta}$ está dada por:

$$p(\boldsymbol{\beta} \mid \sigma_\varepsilon^2) \propto \exp\left\{-\frac{1}{2\sigma_\varepsilon^2} \left(\lambda_1 \sum_{j=1}^p |\beta_j| + \lambda_2 \sum_{j=1}^p \beta_j^2\right)\right\} \quad (3.5)$$

y $p(\sigma_\varepsilon^2) \propto 1/\sigma_\varepsilon^2$ sobre σ_ε^2 . De esta manera se puede migrar a una versión Bayesiana del EN (BEN).

De la a priori en (3.5) se puede observar que se tienen casos especiales de a priori para $\boldsymbol{\beta}$. Cuando $\lambda_1 = 0$ y $\lambda_2 > 0$, se tiene la distribución Normal y si $\lambda_1 > 0$ y $\lambda_2 = 0$, se obtiene la distribución Doble Exponencial o Laplace. En la [Figura 3.1](#) se presentan distintas formas de la distribución de $\boldsymbol{\beta}$ para diferentes valores de λ_1 y λ_2 cuando $p = \sigma_\varepsilon^2 = 1$.

3.2. Métodos de Regresión Penalizada

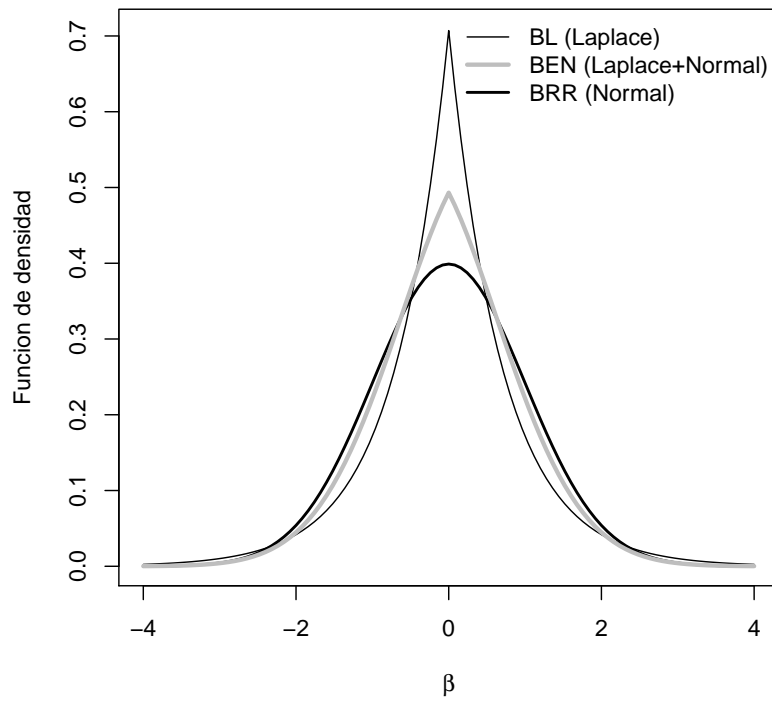


Figura 3.1: Formas de la a priori de β para distintos valores de λ_1 y λ_2

Capítulo 4

BEN mixto

En los programas de mejoramiento genético se cuenta con información relacionada con el parentesco de los individuos o líneas utilizadas como base en las cruzas para obtener nuevos individuos que se prueban en campo. Esta información puede incorporarse en los modelos vistos en la sección previa usando el enfoque del modelo mixto.

En este capítulo se muestra como extender el modelo BEN para incluir la información del pedigree u otra matriz Genómica (ver [VanRaden, 2008](#)). Asimismo, se muestra como resolver el problema mediante métodos Monte Carlo con Cadenas de Markov (MCMC).

4.1. BEN más pedigree

Siguiendo las ideas en [de los Campos *et al.* \(2009\)](#), el modelo BEN se puede extender para que incluya MM y un efecto genético infinitesimal que permita asociar a los individuos en un pedigree (ver Capítulo 5). El modelo lineal es:

$$y_i = \mathbf{x}_i\boldsymbol{\beta} + u_i + \varepsilon_i, \quad i = 1, \dots, n,$$

donde y_i es la variable respuesta para el i -ésimo fenotipo, $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{ip})$ es un conjunto de MM (covariables) para el i -ésimo fenotipo y u_i es el efecto genético infinitesimal para el i -ésimo fenotipo, determinado por un pedigree. Se supone que $\mathbf{u} = (u_1, \dots, u_n)^T \sim N_n(\mathbf{0}, \sigma_u^2 \mathbf{A})$, donde \mathbf{A} es una matriz cuadrada de relaciones aditivas entre los n fenotipos llamada pedigree, y $\varepsilon_i \sim NIID(0, \sigma_\varepsilon^2)$ es el usual término de residuales.

4.2. Simulación

Para obtener valores de la distribución de los parámetros y con ello hacer estimaciones puntuales, intervalos de credibilidad, etc, se implementará el muestreador de Gibbs (Cassella y George, 1992) apoyándose del algoritmo EM Monte Carlo (Levine y Fan, 2004) para calcular los pesos λ_1 y λ_2 . Antes de esto, será necesario calcular las distribuciones condicionales completas para cada uno de los parámetros, y para esto, será de gran utilidad asignarle distribuciones a priori apropiadas a los parámetros para poder encontrar fácilmente la distribución a posteriori conjunta.

4.2.1. Distribuciones a priori y distribución a posteriori

Se propone una distribución a priori a β , pero condicionada sobre σ_ε^2 , tal como:

$$p(\beta | \sigma_\varepsilon^2) \propto \exp \left\{ -\frac{1}{2\sigma_\varepsilon^2} \left(\lambda_1 \sum_{j=1}^p |\beta_j| + \lambda_2 \sum_{j=1}^p \beta_j^2 \right) \right\}.$$

A σ_ε^2 se le asigna una a priori no informativa de la forma $p(\sigma_\varepsilon^2) \propto 1/\sigma_\varepsilon^2$.

Hasta este punto, la presencia del término $|\beta_j|$ llevará a distribuciones condicionales no conocidas (Li y Lin, 2010); sin embargo, mediante el Lema 1 mostrado en el Anexo A, se tiene que la distribución a priori para $\beta | \sigma_\varepsilon^2$ se puede escribir como una mezcla de distribuciones Normales, una sobre β con media cero y varianza $\sigma_\varepsilon^2(t-1)/(\lambda_2 t)$ y una distribución Gama Truncada para t con parámetro de forma $1/2$, parámetro de escala $8\lambda_2\sigma_\varepsilon^2/\lambda_1^2$ y soporte $(1, \infty)$ (Li y Lin, 2010), es decir:

$$p(\beta | \sigma_\varepsilon^2) \propto \prod_{j=1}^p \int_1^\infty \sqrt{\frac{t}{t-1}} \exp \left\{ -\frac{\beta_j^2}{2} \left(\frac{\lambda_2}{\sigma_\varepsilon^2} \frac{t}{t-1} \right) \right\} t^{-\frac{1}{2}} \exp \left\{ -\frac{1}{2\sigma_\varepsilon^2} \frac{\lambda_1^2}{4\lambda_2} t \right\} dt.$$

La expresión anterior implica que:

$$\beta | \tau, \sigma_\varepsilon^2 \sim \prod_{j=1}^p N \left(0, \frac{\sigma_\varepsilon^2(\tau_j - 1)}{\lambda_2 \tau_j} \right)$$

y

$$\tau | \sigma_\varepsilon^2 \sim \prod_{j=1}^p GT \left(\frac{1}{2}, \frac{8\lambda_2\sigma_\varepsilon^2}{\lambda_1^2}, (1, \infty) \right),$$

donde $GT(\psi, \delta, \theta)$ es la distribución Gama Truncada con parámetro de forma ψ , de escala δ y con soporte en θ .

4.2. Simulación

Al efecto \mathbf{u} se le asigna una distribución Normal Multivariada con media $\mathbf{0}_n$ y varianza $\sigma_u^2 \mathbf{A}$, es decir $p(\mathbf{u} | \sigma_u^2) = N_n(\mathbf{u} | \mathbf{0}, \sigma_u^2 \mathbf{A})$. Al parámetro σ_u^2 se le propone una distribución Ji-cuadrada invertida escalada con v grados de libertad y parámetro de escala s , a saber, $p(\sigma_u^2 | s, v) = \chi^{-2}(\sigma_u^2 | s, v)$. La función de densidad de probabilidades (fdp) de esta distribución es:

$$f(x | s, v) = \frac{\left(\frac{sv}{2}\right)^{\frac{v}{2}}}{\Gamma\left(\frac{v}{2}\right) x^{1+\frac{v}{2}}} \exp\left\{-\frac{sv}{2x}\right\}, \quad x > 0, s > 0, v > 0.$$

Con estas a priori asignadas, note que se tienen dos jerarquías en el modelo:

$$p(\boldsymbol{\beta} | \boldsymbol{\tau}, \sigma_\varepsilon^2) \rightarrow p(\boldsymbol{\tau} | \sigma_\varepsilon^2) \rightarrow p(\sigma_\varepsilon^2)$$

y

$$p(\mathbf{u} | \sigma_u^2) \rightarrow p(\sigma_u^2 | s, v).$$

Con esta información, se procederá a encontrar la distribución a posteriori de los parámetros $\{\boldsymbol{\beta}, \boldsymbol{\tau}, \mathbf{u}, \sigma_\varepsilon^2, \sigma_u^2\}$, comenzando por calcular la función de verosimilitud, la cual es:

$$p(\mathbf{y} | \boldsymbol{\beta}, \sigma_\varepsilon^2, \mathbf{u}) = \prod_{i=1}^n N(y_i | \mathbf{x}_i \boldsymbol{\beta} + u_i, \sigma_\varepsilon^2).$$

La distribución a priori conjunta se puede escribir como:

$$\begin{aligned} p(\boldsymbol{\beta}, \boldsymbol{\tau}, \mathbf{u}, \sigma_\varepsilon^2, \sigma_u^2) &= p(\boldsymbol{\beta} | \boldsymbol{\tau}, \sigma_\varepsilon^2) p(\boldsymbol{\tau} | \sigma_\varepsilon^2) p(\sigma_\varepsilon^2) p(\mathbf{u} | \sigma_u^2) p(\sigma_u^2 | s, v) \\ &\propto \prod_{j=1}^p \left[N\left(\beta_j | 0, \frac{\sigma_\varepsilon^2 (\tau_j - 1)}{\lambda_2 \tau_j}\right) \cdot GT\left(\tau_j | \frac{1}{2}, \frac{8\lambda_2 \sigma_\varepsilon^2}{\lambda_1^2}, (1, \infty)\right) \right] \frac{1}{\sigma_\varepsilon^2} \\ &\quad \times N_n(\mathbf{u} | \mathbf{0}, \sigma_u^2 \mathbf{A}) \cdot \chi^{-2}(\sigma_u^2 | s, v). \end{aligned}$$

Ahora, usando la verosimilitud y la a priori conjunta, es posible aplicar el Teorema de Bayes para obtener la distribución a posteriori de los parámetros, la cual está dada por:

$$\begin{aligned} p(\boldsymbol{\beta}, \boldsymbol{\tau}, \mathbf{u}, \sigma_\varepsilon^2, \sigma_u^2 | \mathbf{y}) &\propto p(\boldsymbol{\beta} | \boldsymbol{\tau}, \sigma_\varepsilon^2) p(\boldsymbol{\tau} | \sigma_\varepsilon^2) p(\sigma_\varepsilon^2) p(\mathbf{u} | \sigma_u^2) p(\sigma_u^2 | s, v) p(\mathbf{y} | \boldsymbol{\beta}, \sigma_\varepsilon^2, \mathbf{u}) \\ &\propto \prod_{j=1}^p \left[N\left(\beta_j | 0, \frac{\sigma_\varepsilon^2 (\tau_j - 1)}{\lambda_2 \tau_j}\right) \cdot GT\left(\tau_j | \frac{1}{2}, \frac{8\lambda_2 \sigma_\varepsilon^2}{\lambda_1^2}, (1, \infty)\right) \right] \frac{1}{\sigma_\varepsilon^2} \\ &\quad \times N_n(\mathbf{u} | \mathbf{0}, \sigma_u^2 \mathbf{A}) \cdot \chi^{-2}(\sigma_u^2 | s, v) \prod_{i=1}^n N(y_i | \mathbf{x}_i \boldsymbol{\beta} + u_i, \sigma_\varepsilon^2). \end{aligned}$$

4.2. Simulación

4.2.2. Distribuciones condicionales

La selección de las distribuciones a prioris permiten encontrar fácilmente las distribuciones condicionales completas necesarias para implementar el muestreador de Gibbs, las cuales resultan ser de la siguiente manera:

1. $p(\boldsymbol{\beta} \mid \text{resto})$

$$\begin{aligned} p(\boldsymbol{\beta} \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\tau}, \mathbf{u}, \sigma_\varepsilon^2, \sigma_u^2) &\propto \prod_{i=1}^n N(y_i \mid \mathbf{x}_i \boldsymbol{\beta} + u_i, \sigma_\varepsilon^2) \cdot \prod_{j=1}^p N\left(\beta_j \mid 0, \frac{\sigma_\varepsilon^2 (\tau_j - 1)}{\lambda_2 \tau_j}\right) \\ &\propto \exp \left\{ -\frac{1}{2\sigma_\varepsilon^2} \left[\text{SCC} + \sum_{j=1}^p \frac{\lambda_2 \tau_j}{\tau_j - 1} \right] \right\}, \end{aligned}$$

donde $\text{SCC} = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{u})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{u})$. La distribución condicional completa para $\boldsymbol{\beta}$ es Normal Multivariada con media $\mathbf{B}^{-1} \mathbf{X}^T (\mathbf{y} - \mathbf{u})$ y matriz de covarianzas $\sigma_\varepsilon^2 \mathbf{B}^{-1}$, donde $\mathbf{B} = \mathbf{X}^T \mathbf{X} + \lambda_2 \text{diag}(\frac{\tau_1}{\tau_1 - 1}, \dots, \frac{\tau_p}{\tau_p - 1})$. No es necesario obtener \mathbf{B}^{-1} para muestrear de $\boldsymbol{\beta}$, ya que se pueden aplicar los resultados demostrados en [Sorensen y Gianola \(2002\)](#) y muestrear de $p(\beta_j \mid \text{resto})$, $j = 1, \dots, p$, es decir, cada β_j tiene distribución condicional completa Normal con $E(\beta_j \mid \text{resto}) = b_{jj}^{-1} (r_j - \sum_{k \neq j} b_{jk} \beta_k)$ y varianza $\text{Var}(\beta_j \mid \text{resto}) = \sigma_\varepsilon^2 b_{jj}^{-1}$, donde b_{jj} es el j -ésimo elemento diagonal de la matriz \mathbf{B} y r_j es el j -ésimo elemento de $\mathbf{r} = \mathbf{X}^T (\mathbf{y} - \mathbf{u})$.

2. $p(\boldsymbol{\tau} - \mathbf{1}_p \mid \text{resto})$

Como lo sugieren [Li y Lin \(2010\)](#), la distribución condicional completa pero para $\boldsymbol{\tau} - \mathbf{1}_p$ es:

$$\boldsymbol{\tau} - \mathbf{1}_p \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}, \mathbf{u}, \sigma_\varepsilon^2, \sigma_u^2 \sim \prod_{j=1}^p IGG\left(\frac{1}{2}, \frac{\lambda_1}{4\lambda_2 \sigma_\varepsilon^2}, \frac{\lambda_2 \beta_j^2}{\sigma_\varepsilon^2}\right),$$

donde $IGG(\lambda, \psi, \chi)$ denota la distribución inversa Gaussiana Generalizada, cuya fdp es:

$$f(x \mid \lambda, \psi, \chi) = \frac{(\psi/\chi)^{\lambda/2}}{2K_\lambda(\sqrt{\psi\chi})} x^{\lambda-1} \exp\left\{-\frac{1}{2}(\chi x^{-1} + \psi x)\right\}, \quad x > 0,$$

donde $K_\lambda(\cdot)$ es la función Bessel modificada del tercer tipo con orden λ . Para muestrear de esta distribución, se puede usar la función `rgig()` con el paquete ‘‘HiperbolicDis’’ en R. Sin embargo, [Li y Lin \(2010\)](#) proponen un muestreo más eficiente, esto es, muestrear independientemente para cada $j = 1, \dots, p$, de:

$$\frac{1}{\tau_j - 1} \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}, \mathbf{u}, \sigma_\varepsilon^2, \sigma_u^2 \sim IG\left(\mu = \frac{\sqrt{\lambda_1}}{2\lambda_2 |\beta_j|}, \lambda = \frac{\lambda_1}{4\lambda_2 \sigma_\varepsilon^2}\right), \quad j = 1, \dots, p,$$

4.2. Simulación

donde $IG(\mu, \lambda)$ es la distribución inversa Gaussiana cuya fdp es:

$$f(x | \mu, \lambda) = \sqrt{\frac{\lambda}{2\pi x^3}} \exp\left\{-\frac{\lambda(x - \mu)^2}{2\mu^2 x}\right\}, \quad x > 0, \mu > 0, \lambda > 0.$$

3. $p(\sigma_\varepsilon^2 | \text{resto})$

$$\begin{aligned} \sigma_\varepsilon^2 | \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\tau}, \mathbf{u}, \sigma_u^2 &\sim \left(\frac{1}{\sigma_\varepsilon^2}\right)^{\frac{n}{2}+p+1} \left[\Gamma_U\left(\frac{1}{2}, \frac{\lambda_1^2}{8\lambda_2\sigma_\varepsilon^2}\right)\right]^{-p} \\ &\times \exp\left\{-\frac{1}{2\sigma_\varepsilon^2} \left[\text{SCC} + \lambda_2 \sum_{j=1}^p \frac{\tau_j}{\tau_j - 1} \beta_j^2 + \frac{\lambda_1^2}{4\lambda_2} \sum_{j=1}^p \tau_j\right]\right\}, \end{aligned}$$

donde $\Gamma_U(\alpha, x) = \int_x^\infty t^{\alpha-1} e^{-t} dt$ es la función gama incompleta superior. El muestreo de esta distribución se hará por medio del algoritmo de aceptación-rechazo. Sea $f(\sigma_\varepsilon^2) \propto 1/\sigma_\varepsilon^2$, entonces por definición de la función gama incompleta:

$$f(\sigma_\varepsilon^2) \leq \Gamma\left(\frac{1}{2}\right)^{-p} \left(\frac{1}{\sigma_\varepsilon^2}\right)^{a+1} \exp\left\{-\frac{1}{\sigma_\varepsilon^2} b\right\} = \frac{\Gamma(a) \Gamma\left(\frac{1}{2}\right)^{-p}}{b^a} h(\sigma_\varepsilon^2),$$

donde $h(\cdot)$ es la fdp de la Gama-Inversa(a, b) y

$$a = \frac{n}{2} + p, \quad b = \frac{1}{2} \left[(\mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{u})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{u}) + \lambda_2 \sum_{j=1}^p \frac{\tau_j}{\tau_j - 1} \beta_j^2 + \frac{\lambda_1^2}{4\lambda_2} \sum_{j=1}^p \tau_j \right].$$

Para generar σ_ε^2 de $f(\sigma_\varepsilon^2)$, se genera un candidato Z de h y una u de $u \sim U(0, 1)$ y aceptar Z si $\ln u \leq p \ln \Gamma\left(\frac{1}{2}\right) - p \ln \Gamma_u\left(\frac{1}{2}, \frac{\lambda_1^2}{8Z\lambda_2}\right)$.

4. $p(\mathbf{u} | \text{resto})$

$$\begin{aligned} p(\mathbf{u} | \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\tau}, \sigma_\varepsilon^2, \sigma_u^2) &\propto N_n(\mathbf{u} | \mathbf{0}, \sigma_u^2 \mathbf{A}) \cdot \prod_{i=1}^n N(y_i | \mathbf{x}_i \boldsymbol{\beta} + u_i, \sigma_\varepsilon^2) \\ &\propto N_n(\mathbf{u} | \mathbf{0}, \sigma_u^2 \mathbf{A}) \cdot \prod_{i=1}^n N(y_i^* | u_i, \sigma_\varepsilon^2), \end{aligned}$$

donde $y_i^* = y_i - \mathbf{x}_i \boldsymbol{\beta}$. De esta ecuación se sigue que la densidad a posteriori de \mathbf{u} es Normal Multivariada con matriz de covarianzas $\sigma_\varepsilon^2 \mathbf{C}^{-1}$ y media $\mathbf{C}^{-1} \mathbf{y}^*$, donde $\mathbf{C} = \mathbf{I} + (\sigma_\varepsilon^2 / \sigma_u^2) \mathbf{A}^{-1}$. No será necesario invertir la matriz \mathbf{C} , ya que usando resultados mostrados en [Sorensen y Gianola \(2002\)](#), se sigue que cada elemento de \mathbf{u} tiene distribución condicional completa Normal con media $E(u_i | \text{resto}) = c_{ii}^{-1}(r_i - \sum_{k \neq i} c_{ik} u_k)$ y varianza $\text{Var}(u_i | \text{resto}) = \sigma_\varepsilon^2 c_{ii}^{-1}$, donde c_{ii} es el i -ésimo elemento diagonal de la matriz \mathbf{C} y r_i es el i -ésimo elemento de $\mathbf{r} = \mathbf{y}^*$.

4.2. Simulación

5. $p(\sigma_u^2 \mid \text{resto})$

$$p(\sigma_u^2 \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\tau}, \mathbf{u}, \sigma_u^2) \propto N_n(\mathbf{u} \mid \mathbf{0}, \sigma_u^2 \mathbf{A}) \cdot \chi^{-2}(\sigma_u^2 \mid s, v).$$

De esta ecuación se deduce que la distribución condicional completa de σ_u^2 es χ^{-2} con parámetro de escala $s + \mathbf{u}^T \mathbf{A}^{-1} \mathbf{u}$ y grados de libertad $v + n$, donde n es el orden de la matriz cuadrada \mathbf{A} que es igual al número de observaciones.

4.2.3. Muestreo de Gibbs

El muestreo de Gibbs (Casella y George, 1992) es una técnica que permite muestrear de una distribución conjunta utilizando sus distribuciones condicionales.

Se desea generar valores de las variables aleatorias $\boldsymbol{\beta}$. Se posee la distribución conjunta $p(\boldsymbol{\beta}, \boldsymbol{\tau}, \mathbf{u}, \sigma_\varepsilon^2, \sigma_u^2 \mid \mathbf{y})$, pero no es tan fácil obtener la distribución marginal $f(\boldsymbol{\beta})$. Las formas de las a priori asignadas a los parámetros, ayudaron a obtener con mayor facilidad las distribuciones condicionales completas de ellos, a saber:

$$\begin{aligned} p(\beta_j \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\tau}, \mathbf{u}, \sigma_\varepsilon^2, \sigma_u^2), \quad j = 1, \dots, p \\ p(1/(\tau_j - 1) \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}, \mathbf{u}, \sigma_\varepsilon^2, \sigma_u^2), \quad j = 1, \dots, p \\ p(u_i \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\tau}, \sigma_\varepsilon^2, \sigma_u^2), \quad i = 1, \dots, n \\ p(\sigma_\varepsilon^2 \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\tau}, \mathbf{u}, \sigma_u^2) \\ p(\sigma_u^2 \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}, \boldsymbol{\tau}, \mathbf{u}, \sigma_\varepsilon^2) \end{aligned}$$

Con esta información, el muestreo de Gibbs procede de la siguiente manera:

1. Asignar valores iniciales a los parámetros $\boldsymbol{\tau}$, \mathbf{u} , σ_ε^2 , σ_u^2 , es decir, $\boldsymbol{\tau}^{(0)}$, $\mathbf{u}^{(0)}$, $\sigma_\varepsilon^{2(0)}$, $\sigma_u^{2(0)}$.
2. Hacer $k = 1$.
3. Muestrear independientemente para cada $j = 1, \dots, p$, de la condicional

$$p(\beta_j \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\tau}^{(0)}, \mathbf{u}^{(0)}, \sigma_\varepsilon^{2(0)}, \sigma_u^{2(0)})$$

para obtener los valores $\boldsymbol{\beta}^{(k)}$.

4. Muestrear independientemente para cada $j = 1, \dots, p$, de la condicional

$$p(1/(\tau_j - 1) \mid \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}^{(k)}, \mathbf{u}^{(0)}, \sigma_\varepsilon^{2(0)}, \sigma_u^{2(0)})$$

para así tener la muestra $\boldsymbol{\tau}^{(k)} = (\tau_1^{(k)}, \tau_2^{(k)}, \dots, \tau_p^{(k)})$.

4.2. Simulación

5. Muestrear independientemente para cada $i = 1, \dots, n$, de la condicional

$$p(u_i | \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}^{(k)}, \boldsymbol{\tau}^{(k)}, \sigma_\varepsilon^{2(0)}, \sigma_u^{2(0)})$$

y obtener los valores $\mathbf{u}^{(k)}$.

6. Muestrear de la condicional $p(\sigma_\varepsilon^2 | \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}^{(k)}, \boldsymbol{\tau}^{(k)}, \mathbf{u}^{(k)}, \sigma_u^{2(0)})$ y obtener el valor de $\sigma_\varepsilon^{2(k)}$.
7. Muestrear de la condicional $p(\sigma_u^2 | \mathbf{y}, \mathbf{X}, \boldsymbol{\beta}^{(k)}, \boldsymbol{\tau}^{(k)}, \mathbf{u}^{(k)}, \sigma_\varepsilon^{2(k)})$ y obtener el valor de $\sigma_u^{2(k)}$.
8. Hacer $\boldsymbol{\tau}^{(0)} = \boldsymbol{\tau}^{(k)}$, $\mathbf{u}^{(0)} = \mathbf{u}^{(k)}$, $\sigma_\varepsilon^{2(0)} = \sigma_\varepsilon^{2(k)}$, $\sigma_u^{2(0)} = \sigma_u^{2(k)}$, $k = k + 1$ y repetir los pasos 3 – 7 hasta tener un número m deseado de muestras.

Las primeras q muestras se llamarán muestras de *calentamiento*, así que se pueden desechar y solamente tomar en cuenta las últimas $m - q$ muestras.

4.2.4. Selección de los parámetros de penalidad λ_1 y λ_2

Los parámetros λ_1 y λ_2 serán elegidos usando el algoritmo EM Monte Carlo como lo describe [Levine y Fan \(2004\)](#). La función log-verosimilitud corresponde a la mostrada en [Li y Lin \(2010\)](#), es decir:

$$\ln f(\mathbf{y}, \mathbf{w} | \boldsymbol{\Psi}) = p \ln \lambda_1 - p \ln \Gamma_U \left(\frac{1}{2}, \frac{\lambda_1^2}{8\lambda_2\sigma_\varepsilon^2} \right) - \frac{\lambda_2}{2\sigma_\varepsilon^2} \sum_{j=1}^p \frac{\tau_j}{\tau_j - 1} \beta_j^2 - \frac{\lambda_1^2}{8\lambda_2\sigma_\varepsilon^2} \sum_{j=1}^p \tau_j,$$

donde $\boldsymbol{\Psi} = (\lambda_1, \lambda_2)$ y $\mathbf{w} = (\beta_1, \dots, \beta_p, \tau_1, \dots, \tau_p, \sigma_\varepsilon^2)^T$, así que la función log-verosimilitud estimada de las simulaciones Monte Carlo está dada por:

$$Q_m(\boldsymbol{\Psi} | \hat{\boldsymbol{\Psi}}^{(r)}) = \frac{1}{m} \sum_{t=1}^m \ln f(\mathbf{y}, \mathbf{w}_t^{(r)} | \boldsymbol{\Psi}),$$

donde $\mathbf{w}_1^{(r)}, \dots, \mathbf{w}_m^{(r)}$ es una muestra de la distribución condicional de \mathbf{w} dados los valores observados \mathbf{y} y $\boldsymbol{\Psi}^{(r)}$. El sub índice m denota la muestra MC. Se adopta una versión ligeramente modificada del algoritmo EM MCMC automatizado descrito en [Levine y Fan \(2004\)](#).

Después del proceso de calentamiento, se procede como sigue:

1. Generar $\mathbf{w}_1, \dots, \mathbf{w}_m \sim g(\mathbf{w} | \mathbf{y}, \hat{\boldsymbol{\Psi}}^{(0)})$ usando el muestreador de Gibbs descrito anteriormente.

4.2. Simulación

2. Paso-E: Estimar $Q_m(\Psi | \hat{\Psi}^{(r)}) = \frac{1}{m} \sum_{t=1}^m \ln f(\mathbf{y}, \mathbf{w}_t^{(r)} | \Psi)$.
3. Paso-M: Maximizar $Q_m(\Psi | \hat{\Psi}^{(r)})$ para obtener $\hat{\Psi}^{(r+1)}$.
4. Estimar el error MC:
 - a. Obtener la sub muestra al tiempo $t_k = \sum_{i=1}^k x_i$ donde $x_k - 1 \sim Poisson(\theta_k)$, $k = 1, \dots, N_m$ y $N_m = \sup\{n : t_k \leq m\}$.
 - b. Paso-E: Estimar $Q_{N_m}(\Psi | \hat{\Psi}^{(r)}) = \frac{1}{N_m} \sum_{t=1}^{N_m} \ln f(\mathbf{y}, \mathbf{w}_{t_k} | \Psi)$.
 - c. Paso-M: Maximizar $Q_{N_m}(\Psi | \hat{\Psi}^{(r)})$ para obtener $\hat{\Psi}_{EM;t_k}^{(r+1)}$.
 - d. Calcular $\hat{\Sigma}$ usando la ecuación mostrada en [Levine y Fan \(2004\)](#).

$$\hat{\Sigma} \approx \left[Q_{N_m}^{(2)}(\hat{\Psi}^{(r+1)} | \hat{\Psi}^{(r)}) \right]^{-1} \left\{ \frac{1}{m} \sum_{t=1}^m \left[\frac{\partial}{\partial \Psi} \ln f(\mathbf{y}, \mathbf{w}_t^{(r+1)} | \Psi) - \hat{\mu}_m \right] \times \left[\frac{\partial}{\partial \Psi} \ln f(\mathbf{y}, \mathbf{w}_t^{(r+1)} | \Psi) - \hat{\mu}_m \right]^T \right\} \left[Q_{N_m}^{(2)}(\hat{\Psi}^{(r+1)} | \hat{\Psi}^{(r)}) \right]^{-1} \Big|_{\Psi = \hat{\Psi}^{(r+1)}}$$

donde $\hat{\mu}_m = \frac{1}{m} \sum_{t=1}^m \frac{\partial}{\partial \Psi} \ln f(\mathbf{y}, \mathbf{w}_t | \Psi)$.

- e. Obtener el elipsoide de confiabilidad alrededor de Ψ .

$$N_m \left(\hat{\Psi}_{EM;t_k}^{(r+1)} - \Psi \right)^T \hat{\Sigma}^{-1} \left(\hat{\Psi}_{EM;t_k}^{(r+1)} - \Psi \right) \leq \chi_{2,1-\alpha}^2.$$

5. Si $\hat{\Psi}_{EM;t_k}^{(r)}$ se encuentra en la región de confianza del paso previo, entonces ajustar $m = \frac{4}{3}m$ y obtener $\mathbf{w}_1, \dots, \mathbf{w}_m \sim g(\mathbf{w} | \mathbf{y}, \hat{\Psi}^{(r+1)})$ usando el muestreador de Gibbs.
6. Repetir los pasos 2 – 5 hasta convergencia, es decir, detener el algoritmo cuando

$$\text{dif} = \max_i \left(\frac{|\Psi_i^{(r+1)} - \Psi_i^{(r)}|}{\Psi_i^{(r)} + \delta_1} \right) < \delta_2, \quad (4.1)$$

para tres iteraciones consecutivas. Donde i indexa los elementos de Ψ . Los valores δ_1 y δ_2 se pueden ajustar, por ejemplo, a $\delta_1 = 0.001$, $\delta_2 = 0.01$.

Los valores α y θ_k se pueden fijar como lo sugerido por [Levine y Fan \(2004\)](#), es decir, $\alpha = 0.5$ y $\theta_k = \theta k^b$ con $\theta = 1$ y $b = 0.5$.

Para calcular $\hat{\Sigma}$ en el paso 4d, es necesario obtener el vector gradiente y la matriz Hessiana para $\ln f(\mathbf{y}, \mathbf{w} | \Psi)$.

Vector Gradiente

4.2. Simulación

$$\frac{\partial}{\partial \lambda_1} \ln f(\mathbf{y}, \mathbf{w} \mid \Psi) = \frac{p}{\lambda_1} + \frac{p \cdot \exp\left\{-\frac{\lambda_1^2}{8\lambda_2\sigma_\varepsilon^2}\right\}}{\sqrt{2\pi\lambda_2}\sigma_\varepsilon \operatorname{Erfc}\left(\frac{\lambda_1}{\sqrt{8\lambda_2}\sigma_\varepsilon}\right)} - \frac{\lambda_1}{4\lambda_2\sigma_\varepsilon^2} \sum_{j=1}^p \tau_j$$

$$\begin{aligned} \frac{\partial}{\partial \lambda_2} \ln f(\mathbf{y}, \mathbf{w} \mid \Psi) &= -\frac{p\lambda_2 \exp\left\{-\frac{\lambda_1^2}{8\lambda_2\sigma_\varepsilon^2}\right\}}{\sqrt{8\pi\lambda_2^3}\sigma_\varepsilon \operatorname{Erfc}\left(\frac{\lambda_1}{\sqrt{8\lambda_2}\sigma_\varepsilon}\right)} + \frac{\lambda_1^2}{8\lambda_2^2\sigma_\varepsilon^2} \sum_{j=1}^p \tau_j \\ &\quad - \frac{1}{2\sigma_\varepsilon^2} \sum_{j=1}^p \frac{\beta_j^2 \tau_j}{\tau_j - 1} \end{aligned}$$

Matriz Hessiana

$$\begin{aligned} \frac{\partial^2}{\partial \lambda_1 \partial \lambda_1} \ln f(\mathbf{y}, \mathbf{w} \mid \Psi) &= \frac{p \cdot \exp\left\{-\frac{\lambda_1^2}{4\lambda_2\sigma_\varepsilon^2}\right\}}{2\pi\lambda_2\sigma_\varepsilon^2 \operatorname{Erfc}\left(\frac{\lambda_1}{\sqrt{8\lambda_2}\sigma_\varepsilon}\right)^2} - \frac{p\lambda_1 \exp\left\{-\frac{\lambda_1^2}{8\lambda_2\sigma_\varepsilon^2}\right\}}{\sqrt{32\pi\lambda_2^3}\sigma_\varepsilon^3 \operatorname{Erfc}\left(\frac{\lambda_1}{\sqrt{8\lambda_2}\sigma_\varepsilon}\right)} \\ &\quad - \frac{p}{\lambda_1^2} - \frac{1}{4\lambda_2\sigma_\varepsilon^2} \sum_{j=1}^p \tau_j \end{aligned}$$

$$\begin{aligned} \frac{\partial^2}{\partial \lambda_2 \partial \lambda_2} \ln f(\mathbf{y}, \mathbf{w} \mid \Psi) &= \frac{p \cdot \exp\left\{-\frac{\lambda_1^2}{4\lambda_2\sigma_\varepsilon^2}\right\} \lambda_1^2}{8\pi\lambda_2^3\sigma_\varepsilon^2 \operatorname{Erfc}\left(\frac{\lambda_1}{\sqrt{8\lambda_2}\sigma_\varepsilon}\right)^2} - \frac{p \cdot \exp\left\{-\frac{\lambda_1^2}{8\lambda_2\sigma_\varepsilon^2}\right\} \lambda_1^3}{\sqrt{512\pi\lambda_2^7}\sigma_\varepsilon^3 \operatorname{Erfc}\left(\frac{\lambda_1}{\sqrt{8\lambda_2}\sigma_\varepsilon}\right)} \\ &\quad + \frac{3p \cdot \exp\left\{-\frac{\lambda_1^2}{8\lambda_2\sigma_\varepsilon^2}\right\} \lambda_1}{\sqrt{32\pi\lambda_2^5}\sigma_\varepsilon \operatorname{Erfc}\left(\frac{\lambda_1}{\sqrt{8\lambda_2}\sigma_\varepsilon}\right)} - \frac{\lambda_1^2}{4\lambda_2^3\sigma_\varepsilon^2} \sum_{j=1}^p \tau_j \end{aligned}$$

$$\begin{aligned} \frac{\partial^2}{\partial \lambda_1 \partial \lambda_2} \ln f(\mathbf{y}, \mathbf{w} \mid \Psi) &= -\frac{p\lambda_1 \exp\left\{-\frac{\lambda_1^2}{4\lambda_2\sigma_\varepsilon^2}\right\}}{4\pi\lambda_2^2\sigma_\varepsilon^2 \operatorname{Erfc}\left(\frac{\lambda_1}{\sqrt{8\lambda_2}\sigma_\varepsilon}\right)^2} + \frac{p\lambda_1^2 \exp\left\{-\frac{\lambda_1^2}{8\lambda_2\sigma_\varepsilon^2}\right\}}{\sqrt{128\pi\lambda_2^5}\sigma_\varepsilon^3 \operatorname{Erfc}\left(\frac{\lambda_1}{\sqrt{8\lambda_2}\sigma_\varepsilon}\right)} \\ &\quad - \frac{p \cdot \exp\left\{-\frac{\lambda_1^2}{8\lambda_2\sigma_\varepsilon^2}\right\}}{\sqrt{8\pi\lambda_2^3}\sigma_\varepsilon \operatorname{Erfc}\left(\frac{\lambda_1}{\sqrt{8\lambda_2}\sigma_\varepsilon}\right)} + \frac{\lambda_1}{4\lambda_2^2\sigma_\varepsilon^2} \sum_{j=1}^p \tau_j, \end{aligned}$$

4.2. Simulación

donde $\text{Erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2} dt$ es la función de error complementaria. Note que $\text{Erfc}(x) = \Phi(-\sqrt{2}x)$, siendo $\Phi(\cdot)$ la función de distribución de una variable Normal Estándar.

4.2.5. Validación cruzada

La VC es un popular método que se usa para estimar el error de predicción y comparar diferentes modelos. Consiste en dividir los datos aleatoriamente en conjuntos disjuntos, comúnmente llamados *folds* (en este caso, 10). Así, los datos quedarán divididos en 10 conjuntos de tamaño aproximado $k = \lfloor n/10 \rfloor$. Por ejemplo, si n es múltiplo de 10:

$$\mathbf{y}^* = \underbrace{(y_{[1]}, \dots, y_{[k]})}_{\text{conjunto 1}}, \underbrace{(y_{[k+1]}, \dots, y_{[2k]})}_{\text{conjunto 2}}, \dots, \underbrace{(y_{[9k+1]}, \dots, y_{[10k]})}_{\text{conjunto 10}}^T,$$

donde $y_{[i]}$ es cualquier elemento de \mathbf{y} que fue ordenado para que ocupe la posición $[i]$. La matriz \mathbf{X} también se ordena de tal manera que sus n hileras correspondan a los n elementos de \mathbf{y}^* , resultando \mathbf{X}^* .

El método procede de la siguiente manera:

1. Se ajusta el modelo $\mathbf{y}_{-1} = \mathbf{X}_{-1}\boldsymbol{\beta}$ y se obtienen los valores $\hat{\boldsymbol{\beta}}_{-1}$, donde \mathbf{y}_{-1} es el vector \mathbf{y}^* de donde se suprimieron las k observaciones correspondientes al conjunto 1 y \mathbf{X}_{-1} es la matriz \mathbf{X}^* de la cual se eliminaron las hileras correspondientes al conjunto 1.
2. Con el vector de coeficientes estimados $\hat{\boldsymbol{\beta}}_{-1}$, se predicen los k valores eliminados:

$$\hat{\mathbf{y}}_1 = \mathbf{X}_1 \hat{\boldsymbol{\beta}}_{-1} + \sigma_\varepsilon z,$$

donde $z \sim N(0, 1)$, $\hat{\mathbf{y}}_1 = (\hat{y}_{[1]}, \dots, \hat{y}_{[k]})^T$ y \mathbf{X}_1 es la matriz que solo contiene las k hileras correspondientes al conjunto 1.

3. Se repiten los pasos 1 y 2 para cada uno de los conjuntos restantes.
4. Cuando ya se tienen todas las observaciones predichas de todos los 10 conjuntos, es decir, el vector $\hat{\mathbf{y}}^* = (\hat{y}_{[1]}, \dots, \hat{y}_{[k]}, \hat{y}_{[k+1]}, \dots, \hat{y}_{[2k]}, \dots, \hat{y}_{[9k+1]}, \dots, \hat{y}_{[10k]})^T$, se calcula su correlación con el vector de observaciones \mathbf{y}^* .

La correlación del paso 4 es de interés, ya que entre más grande sea, mejor será el modelo utilizado para predecir valores futuros.

Usualmente n no es múltiplo de 10, en cuyo caso algunos de los grupos tendrán más observaciones que el resto, pero la idea general es la misma.

Capítulo 5

Aplicación del BEN mixto en Selección Genómica

La predicción de valores genéticos es de gran importancia en genética cuantitativa. Una buena predicción de valores genéticos de genotipos cuyos fenotipos aun no se han observado es necesaria para lograr una mejora genética y reducir costos fenotípicos. Por muchos años, tales predicciones, se han realizado usando datos fenotípicos y de sus familiares, estos últimos representados por un pedigree. Actualmente se disponen de mapas genéticos de humanos, plantas y animales, los cuales comprenden una gran cantidad de segmentos de cromosomas o Marcadores Moleculares (MM) ([Meuwissen *et al.*, 2001](#)).

La información sobre los MM junta con la información fenotípica, se pueden usar para incrementar el progreso genético aumentando la precisión de la selección y reduciendo el intervalo de generación ([Fernando y Grossman, 1989](#)).

Mediante el uso de técnicas de Ingeniería Genética, es posible realizar experimentos a nivel molecular en busca de variedades mejoradas que puedan aportar un mayor beneficio y un menor costo para el agricultor. Para esto es necesario identificar aquellas regiones del genoma fuertemente asociadas con el fenotipo de interés. Esto se logra con una buena selección de MM y el uso de mapas genéticos. Comúnmente, el número de MM (p) es muy grande comparado con el número de registros genotípicos (n), así que para realizar una buena predicción y selección de variables, se tiene que recurrir a buenos métodos de regresión. Por esta razón se implementará el modelo BEN a este tipo de datos, agregando la información contenida en un pedigree.

5.1. Datos de cebada (*Hordeum vulgare*)

5.1. Datos de cebada (*Hordeum vulgare*)

Estos datos de cebada provienen de un proyecto de mapeo del genoma de la cebada de Norteamérica. Contiene $n = 145$ líneas di-haploides; cada una fue cultivada en 25 diferentes ambientes. El fenotipo analizado fue el *peso promedio del grano*. Estos datos contienen información genotípica con $p = 127$ MM binarios codificados como 0 para un genotipo y 1 para el otro. Más información sobre este proyecto está disponible en <http://www.intl-pag.org/3/abstracts/37pg3.html>. Estos datos fueron analizados anteriormente por Yi y Xu (2008).

5.2. Datos de trigo (*Triticum aestivum*)

Esta base de datos contiene un conjunto de $n = 599$ líneas de trigo del Programa Global de Trigo del CIMMYT, este programa comprendió muchos experimentos por todo el mundo a través de una gran variedad de ambientes. Las líneas de trigo fueron genotipeadas para $p = 1447$ MM usando DArT (Diversity Array Technology) de Triticarte (Canberra, Australia; <http://www.triticarte.com.au>), estos MM DArT fueron tomados en dos valores denotados por su presencia y su ausencia (1 y 0). El fenotipo evaluado fue el *rendimiento del grano* en cuatro mega-ambientes ($A_1 - A_4$). Se dispone también de información del pedigree, conformado con datos de generaciones anteriores usando el sistema ICIS (Internacional Crop Information System) descrito en http://cropwiki.irri.org/icis/index.php/TDM_GMS_Browse. Estos datos se encuentran disponibles libremente en internet y están incluidos en la biblioteca de funciones BLR (<http://cran.r-project.org/web/packages/BLR/index.html>).

5.3. Datos de maíz (*Zea mays*)

Estos datos provienen de un estudio de asociación del genoma del maíz usando una técnica llamada mapeo de asociación anidado tal como se describe en <http://maizecoop.cropsci.uiuc.edu/nam-rils.php>. Los fenotipos evaluados son *días a la antesis* (floración masculina, DTA), *días a la seda* (floración femenina, DTS) y el calculado *intervalo antesis-seda* (ASI), (Buckler *et al.*, 2009, Tiang *et al.*, 2011). El estudio se realizó para 5 poblaciones de maíz (aunque en el presente trabajo solamente se analizaron las poblaciones 1 y 2 con $n_1 = 194$ y $n_2 = 196$) y los fenotipos se genotipearon para $p = 1106$ MM. Los datos se encuentran disponibles en la url http://www.panzea.org/db/gateway?file_id=Buckler_etal_2009_Science_flowering_time_data.

Capítulo 6

Resultados y Discusión

En este apartado se presentan los resultados de la investigación. Para el algoritmo del muestreador de Gibbs se generaron 35,000 muestras (descartando las primeras 5,000 que fueron desechadas), una vez que λ_1 y λ_2 habían sido estimadas usando el algoritmo desarrollado por [Levine y Fan \(2004\)](#). Para verificar la convergencia se graficó el valor de cada uno de los componentes de varianza vs el número de iteración.

Usando los datos de cebada, se implementó el BEN con los parámetros iniciales $\lambda_1 = 1$ y $\lambda_2 = 25$. En la Tabla [6.1](#) se muestran las correlaciones para la prueba de VC para los modelos M-BL, M-BRR y M-BEN. En la Figura [6.1](#) se presenta una comparación de los estimadores β usando el M-BEN contra los estimadores usando el modelo M-BL y M-BRR. Esta figura muestra que los estimadores del BEN son más parecidos con los del BRR, esto se debe a que en el análisis se optimizó $\lambda_1 = 1 < \lambda_2 = 11.54$ (ver Tabla [6.4](#)), siendo λ_2 el parámetro correspondiente a la BRR. En la misma figura se muestra como varía el error definido en la ecuación [\(4.1\)](#). En esta figura se puede ver que en las iteraciones 3, 4 y 5, este error fue menor que $\delta_2 = 0.01$, por lo que el algoritmo de estimación termina con 5 iteraciones EM Monte Carlo.

Para los datos de trigo, los parámetros iniciales para implementar el BEN fueron $\lambda_1 = 0.8$ y $\lambda_2 = 450$, y los hiperparámetros $s = 1$ y $v = 4$ para la varianza a priori de \mathbf{u} . La Tabla [6.2](#) contiene las correlaciones de la VC para cada uno de los modelos M-BL, M-BRR, M-BEN y los que usan el pedigree (PM-BL, PM-BRR y PM-BEN). En las Figuras [6.3-6.6](#) se presentan las comparaciones de los estimadores usando el modelo M-BEN contra los estimadores usado M-BL, M-BRR, PM-BL y PM-BRR, esto para cada uno de los 4 ambientes. En la Figura [6.7](#) se muestra como varía el error de la ecuación [\(4.1\)](#) para el modelo M-BEN y PM-BEN para los 4 ambientes.

Finalmente, para los datos de maíz, los parámetros iniciales utilizados para implementar el BEN fueron $\lambda_1 = 1$ y $\lambda_2 = 25$. Con las VC se obtuvieron las correlaciones mostradas en la Tabla [6.3](#), la cual comprende únicamente las poblaciones 1 y 2 con sus 3 fenotipos:

6. Resultados y Discusión

DTA, DTS y ASI, haciendo uso de los modelos M-BL, M-BRR y M-BEN. En esta tabla, el valor “ND” significa que no fue posible la estimación. En las Figuras 6.10 y 6.11 se ilustran las comparaciones de los estimadores usando los distintos modelos para las poblaciones 1 y 2. También, en la Figura 6.12 se muestra como varía el error de la ecuación (4.1) en cada iteración EM Monte Carlo para ambas poblaciones.

En las Figuras 6.2, 6.8, 6.9, 6.13, 6.14 y 6.15 se ilustra el porcentaje de varianza fenotípica explicada por cada MM (es decir, heredabilidad), $h_j^2 = S_j^2 \beta_j^2 / S_y^2$, $j = 1, \dots, p$, donde S_y^2 es la varianza fenotípica y S_j^2 es la varianza muestral de $\mathbf{x}_j = (x_{1j}, \dots, x_{nj})^T$ (Yi y Xu, 2008). Estas figuras muestran que la heredabilidad proporciona una visión más clara que los efectos y puede ser usada como criterio para seleccionar aquellos MM fuertemente asociados con alguna característica de interés de los individuos, por ejemplo, aquellos MM con h^2 grande, sirven para identificar regiones del genoma asociadas con el rendimiento en cebada, en trigo o en el tiempo de floración en maíz, según sea el caso.

Tabla 6.1: Correlaciones obtenidas con cada uno de los 3 modelos usando validación cruzada con 10 folds para los datos de cebada

M-BL	M-BRR	M-BEN
0.8291	0.8042	0.8135

Tabla 6.2: Correlaciones obtenidas con cada uno de los 6 modelos usando validación cruzada con 10 folds para los datos de trigo

	M-BL	M-BRR	M-BEN	PM-BL	PM-BRR	PM-BEN
A1	0.5000	0.5026	0.4973	0.5219	0.5217	0.5117
A2	0.4633	0.4656	0.4678	0.4722	0.4776	0.4771
A3	0.3742	0.3752	0.3790	0.4355	0.4335	0.4463
A4	0.4587	0.4633	0.4624	0.4903	0.4928	0.4928

Tabla 6.3: Correlaciones obtenidas con cada uno de los 3 modelos usando validación cruzada con 10 folds para los datos de maíz

	M-BL	M-BRR	M-BEN
Población 1			
DTA	0.6843	0.6792	0.6754
DTS	0.7215	0.7195	0.7095
ASI	0.4328	0.4351	ND
Población 2			
DTA	0.4292	0.4351	ND
DTS	0.5073	0.5024	0.5149
ASI	0.5364	0.5349	0.5429

6. Resultados y Discusión

Tabla 6.4: Algunos parámetros obtenidos con cada uno de los modelos completos para los tres conjuntos de datos.

Datos	Fen-Amb-Pob	Model	Parámetros			
			σ_ε^2	σ_u^2	λ_1	λ_2
Ce		M-BEN	0.307	—	1	11.54
ba		M-BRR	0.177	—	—	—
da		M-BL	0.173	—	4.094	—
T r i g o	Amb1	M/PM-BEN	0.628/0.525	—/0.119	0.99/0.99	247.31/262.84
		M/PM-BRR	0.542/0.421	—/0.138	—/—	—/—
		M/PM-BL	0.546/0.416	—/0.14	20.15/19.17	—/—
	Amb2	M/PM-BEN	0.635/0.547	—/0.106	0.99/0.99	274.77/307.27
		M/PM-BRR	0.56/0.482	—/0.114	—/—	—/—
		M/PM-BL	0.571/0.494	—/0.115	21.89/24.23	—/—
	Amb3	M/PM-BEN	0.692/0.508	—/NA	0.99/0.99	335.96/348.1
		M/PM-BRR	0.645/0.453	—/0.21	—/—	—/—
		M/PM-BL	0.661/0.465	—/0.227	26.60/31.34	—/—
	Amb4	M/PM-BEN	0.655/0.528	—/0.126	0.99/0.99	301.81/310.55
		M/PM-BRR	0.587/0.453	—/0.166	—/—	—/—
		M/PM-BL	0.601/0.465	—/0.177	23.88/27.55	—/—
M a í z	Pob1 DTA	M-BEN	0.439	—	1.00	432.29
		M-BRR	0.319	—	—	—
		M-BL	0.329	—	29.19	—
	Pob1 DTS	M-BEN	0.405	—	1.01	372.74
		M-BRR	0.2367	—	—	—
		M-BL	NA	—	NA	—
	Pob1 ASI	M-BEN	0.647	—	1.01	741.38
		M-BRR	0.635	—	—	—
		M-BL	0.642	—	63.27	—
	Pob2 DTA	M-BEN	0.659	—	1.01	742.25
		M-BRR	0.634	—	—	—
		M-BL	0.632	—	56.1	—
	Pob2 DTS	M-BEN	0.569	—	1.01	555.46
		M-BRR	0.514	—	—	—
		M-BL	0.538	—	43.50	—
Pob2 ASI	M-BEN	0.522	—	1.01	553.21	
	M-BRR	0.478	—	—	—	
	M-BL	0.475	—	41.36	—	

6. Resultados y Discusión

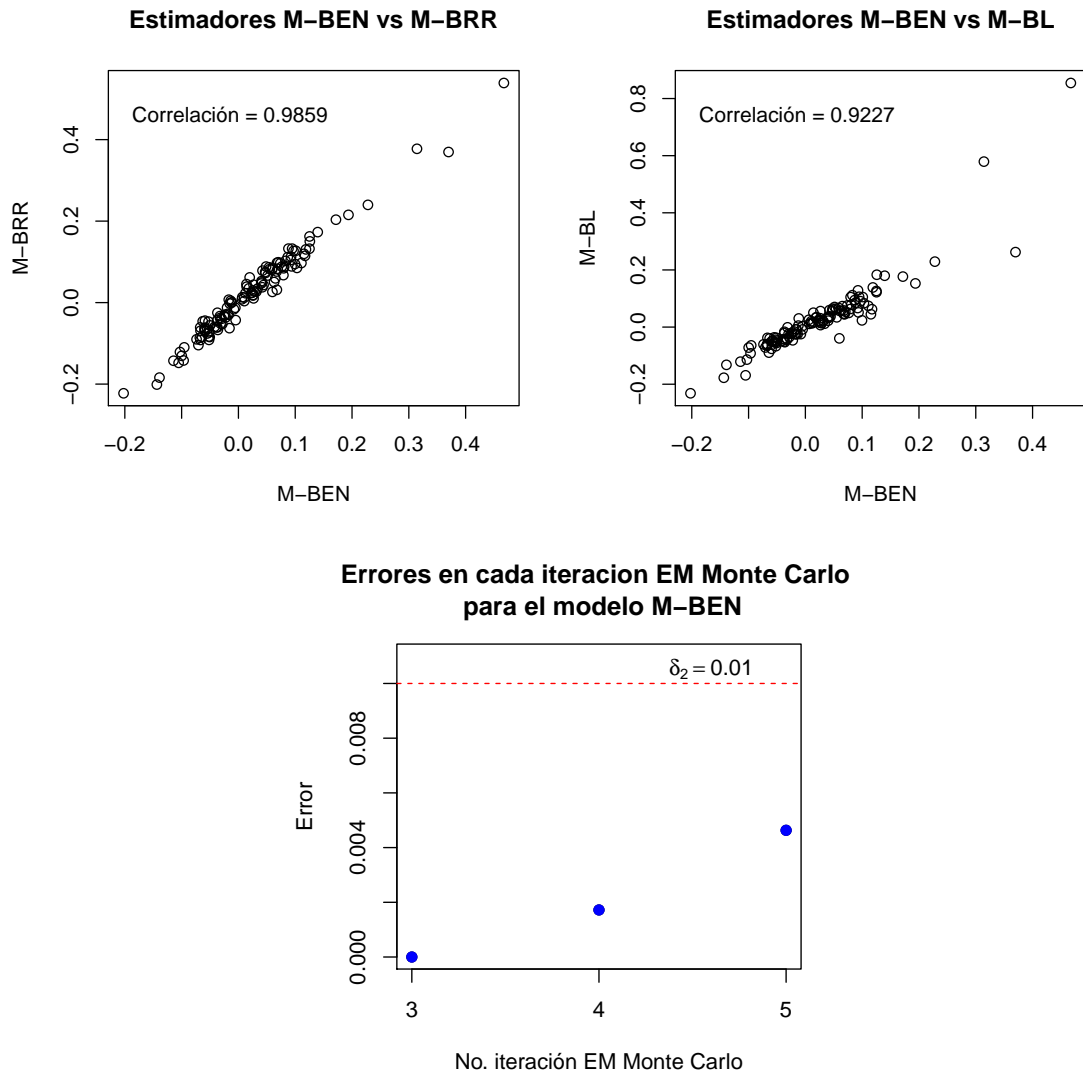


Figura 6.1: Comparación de los estimadores de los efectos usando el modelo M-BEN contra los obtenidos usando M-BL y M-BRR para los datos de cebada. Errores en cada iteración EM Monte Carlo con $\delta_1 = 0.001$ y $\delta_2 = 0.01$ para el modelo M-BEN para los datos de cebada.

6. Resultados y Discusión

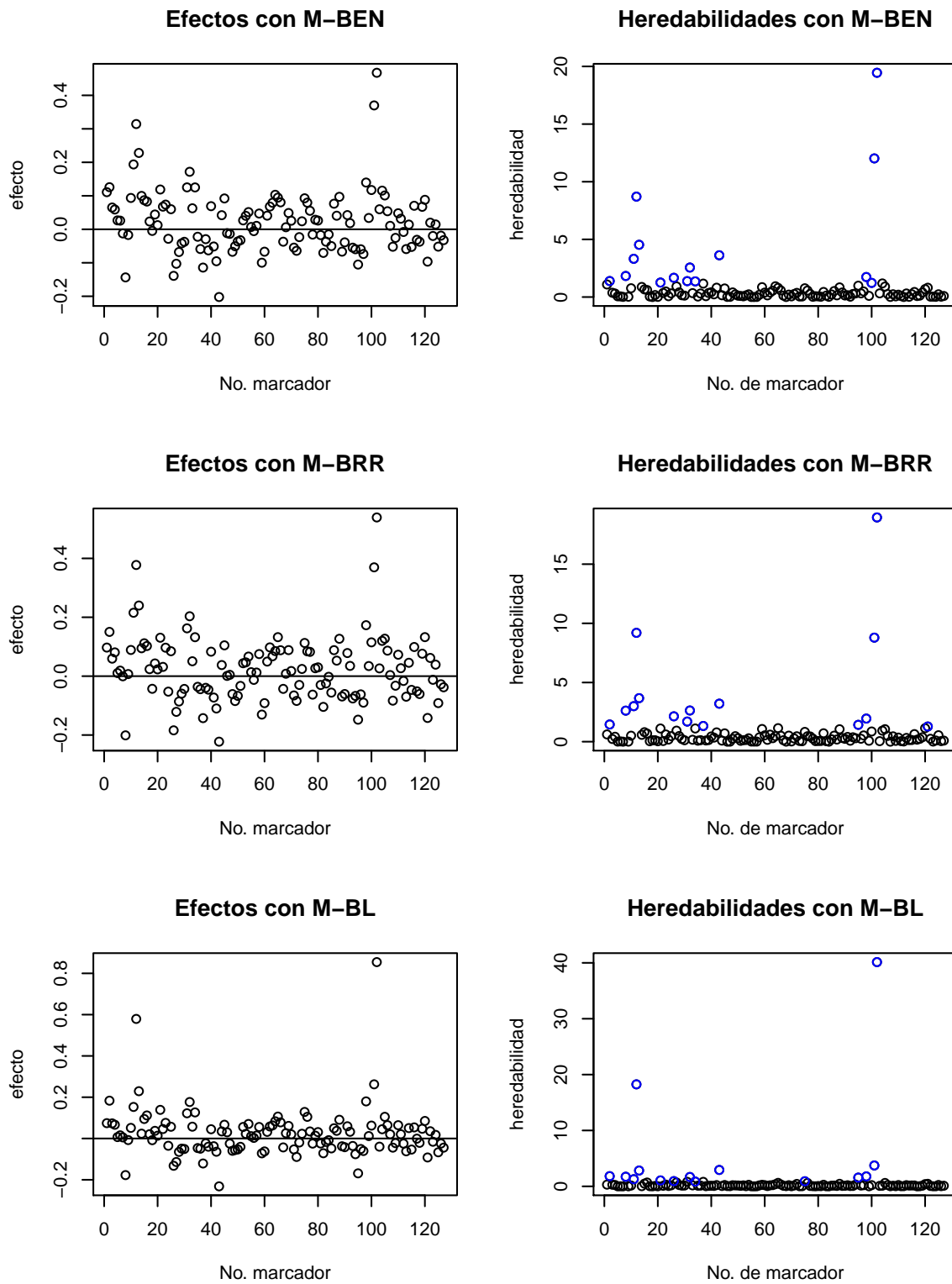


Figura 6.2: Comparación de los efectos y heredabilidades de los MM obtenidos con cada uno de los modelos para los datos de cebada.

6. Resultados y Discusión

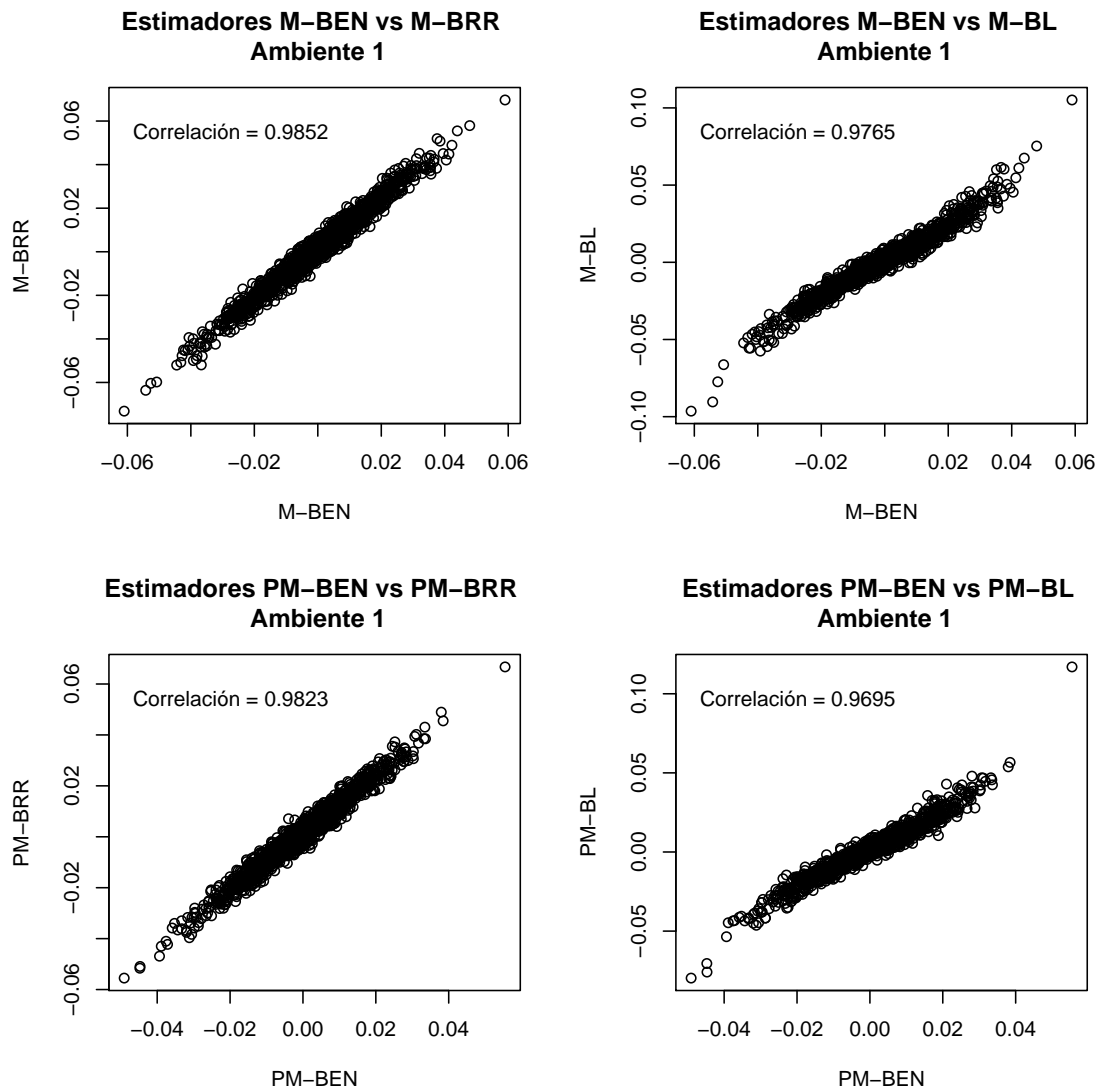


Figura 6.3: Comparación de los estimadores de los efectos usando los modelos M-BEN y PM-BEN contra los obtenidos usando M-BL y M-BRR con y sin pedigree para los datos de trigo en el ambiente 1.

6. Resultados y Discusión

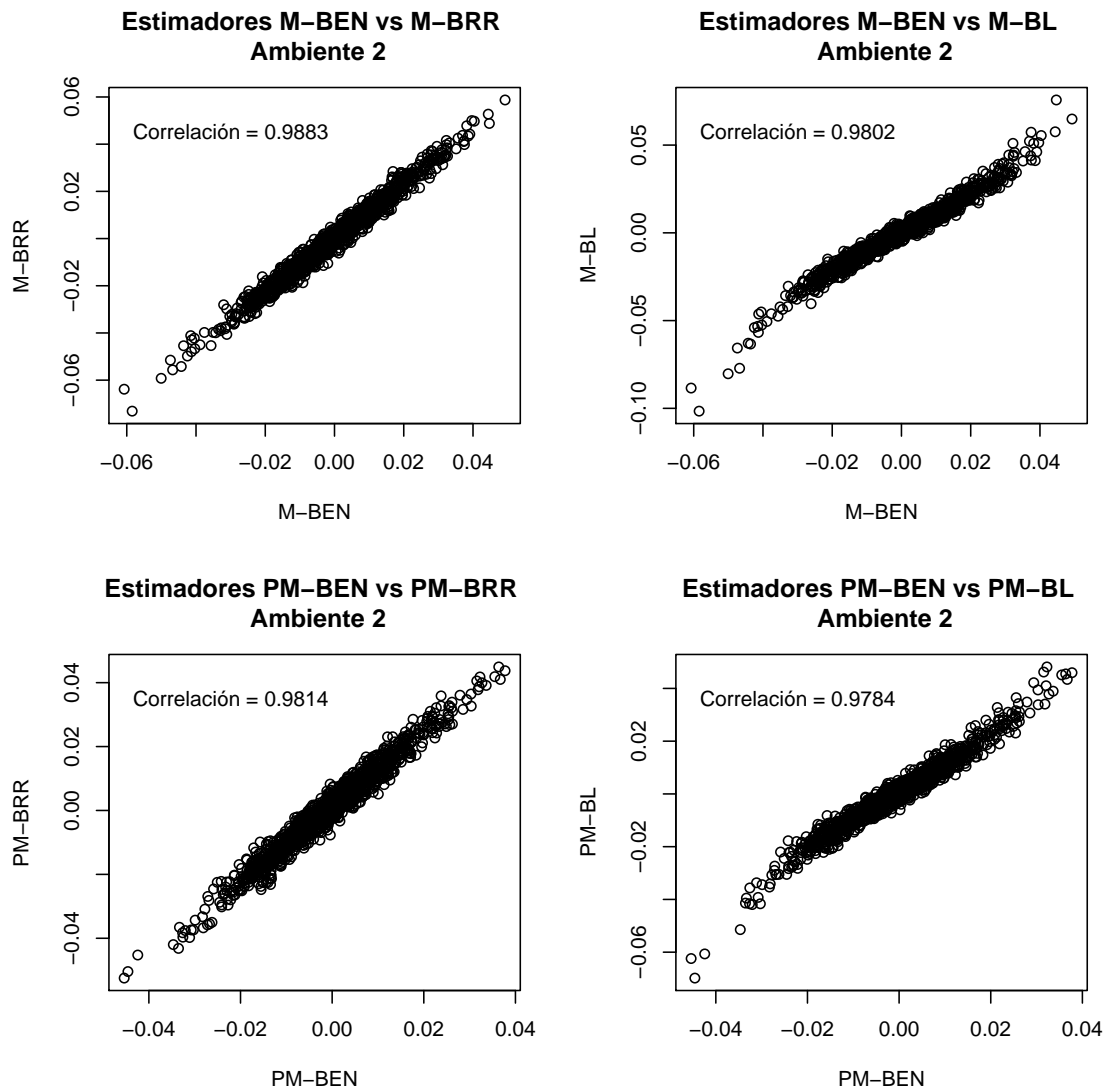


Figura 6.4: Comparación de los estimadores de los efectos usando los modelos M-BEN y PM-BEN contra los obtenidos usando M-BL y M-BRR con y sin pedigree para los datos de trigo en el ambiente 2.

6. Resultados y Discusión

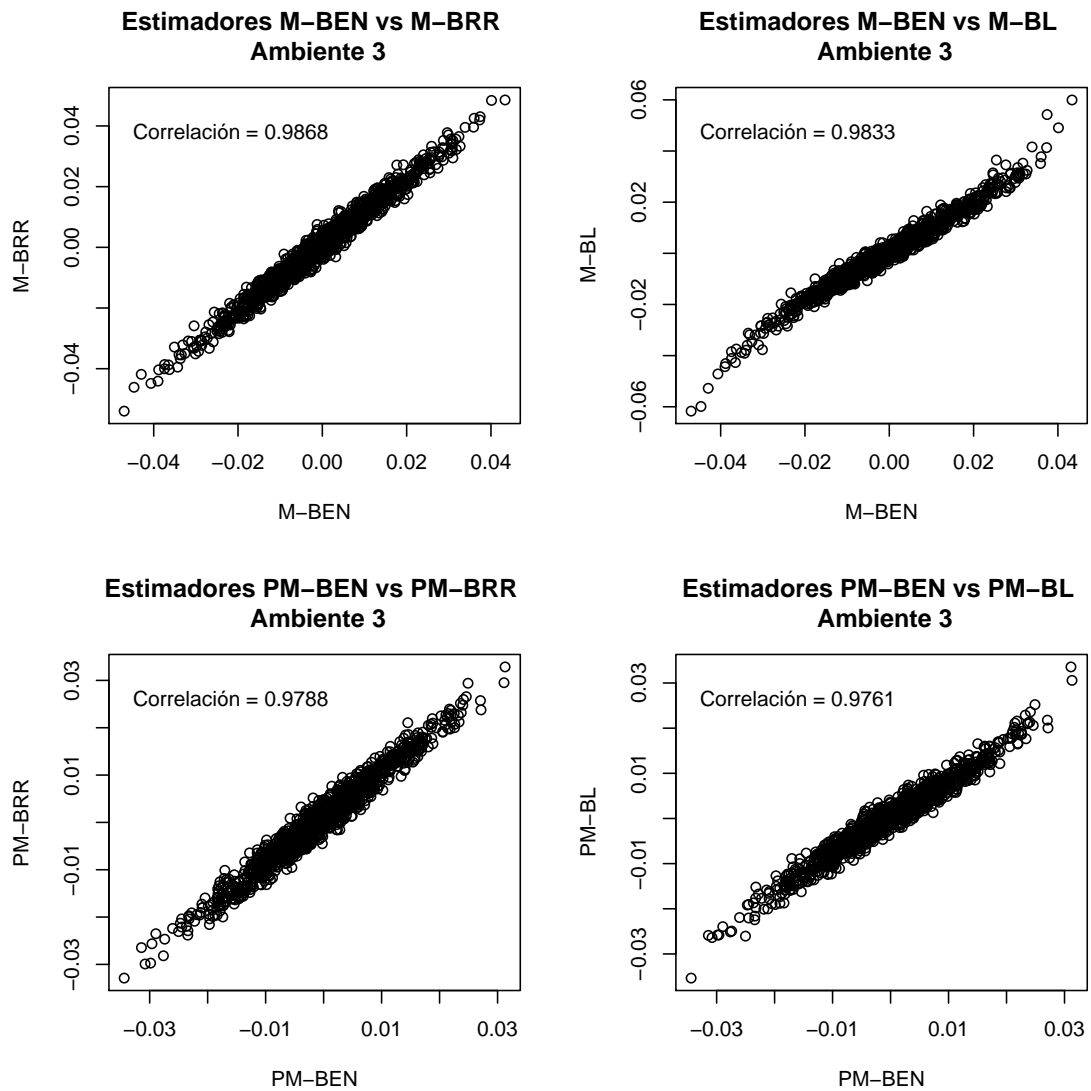


Figura 6.5: Comparación de los estimadores de los efectos usando los modelos M-BEN y PM-BEN contra los obtenidos usando M-BL y M-BRR con y sin pedigree para los datos de trigo en el ambiente 3.

6. Resultados y Discusión

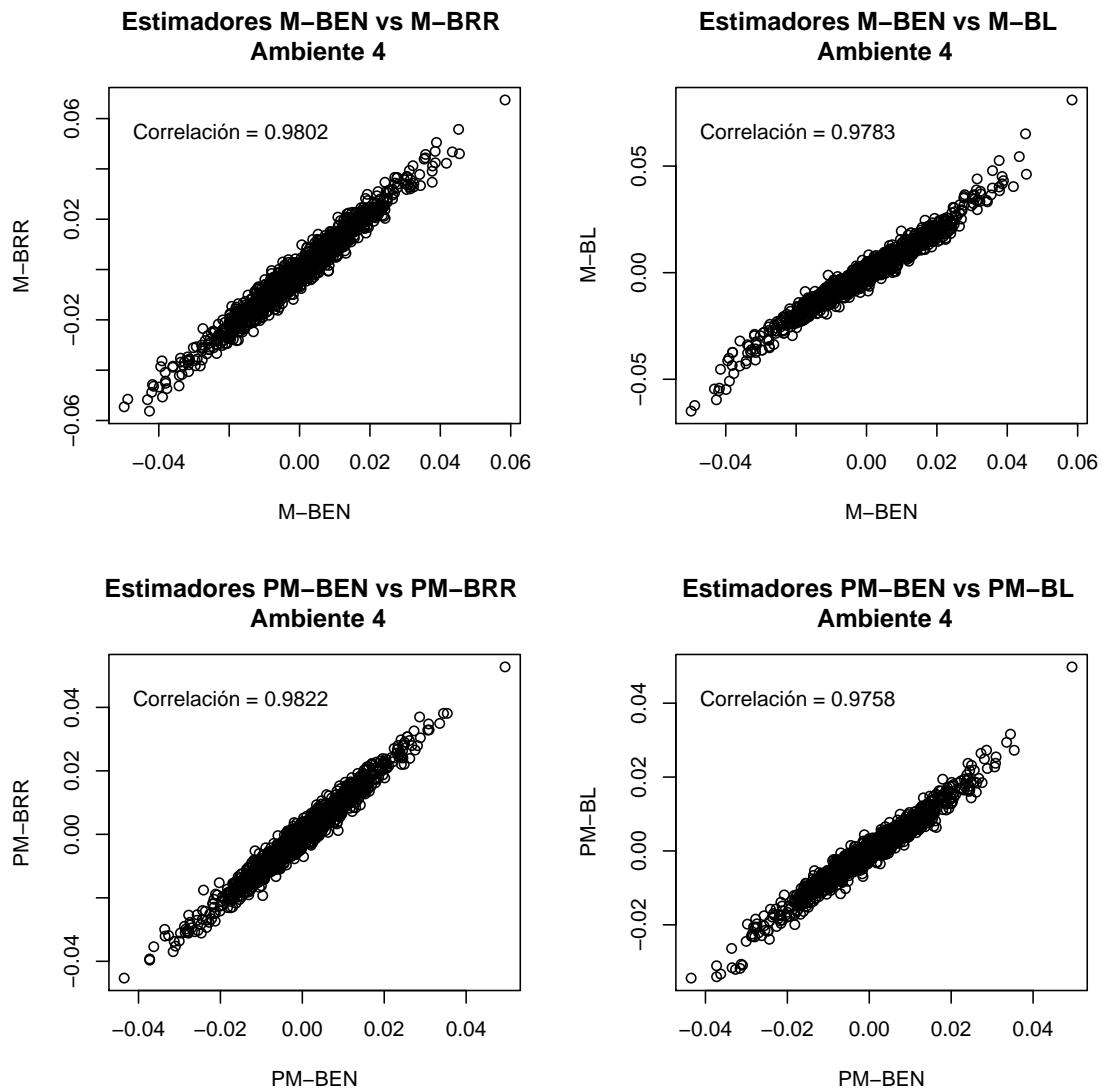


Figura 6.6: Comparación de los estimadores de los efectos usando los modelos M-BEN y PM-BEN contra los obtenidos usando M-BL y M-BRR con y sin pedigree para los datos de trigo en el ambiente 4.

6. Resultados y Discusión

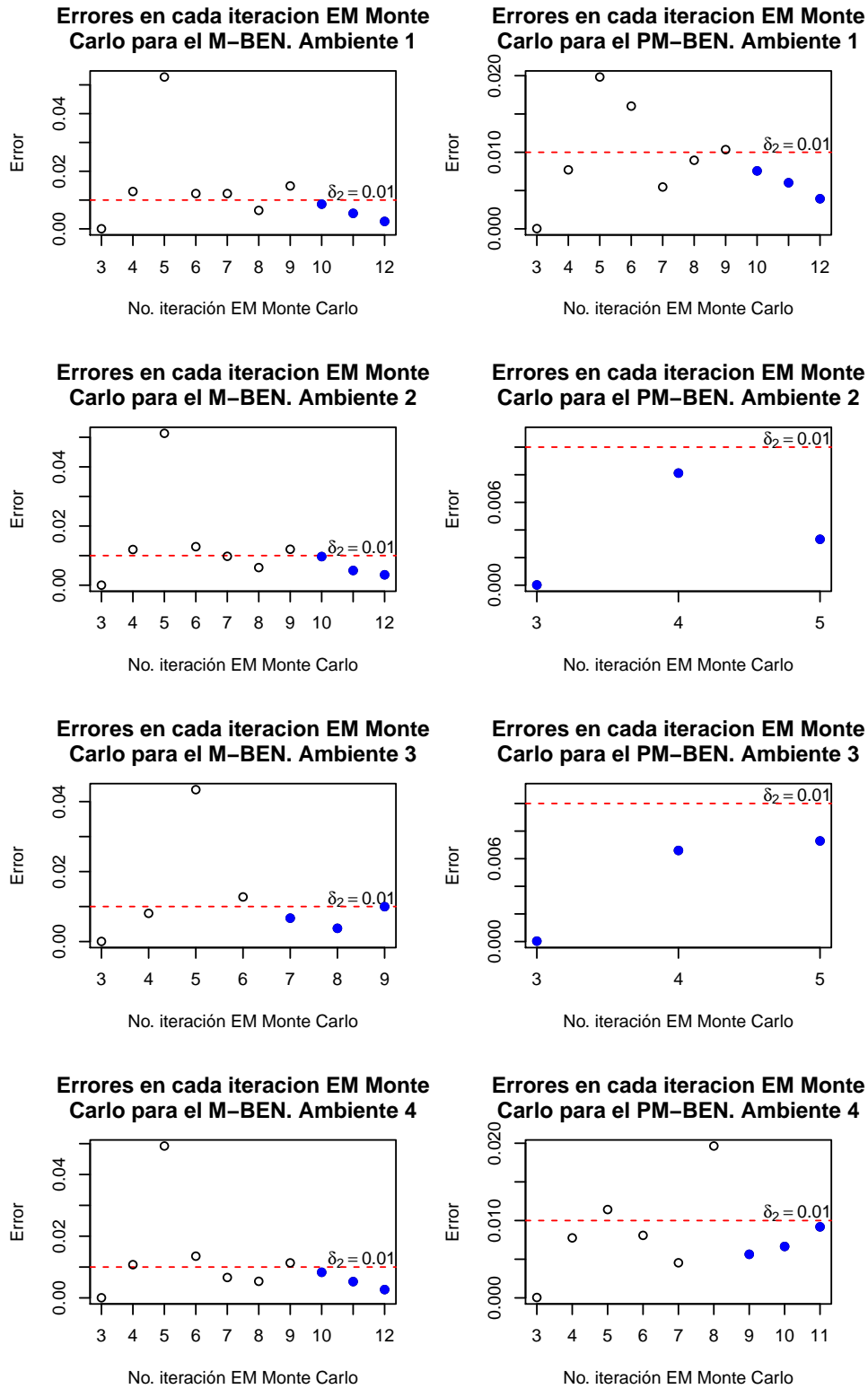


Figura 6.7: Errores en cada iteración EM Monte Carlo con $\delta_1 = 0.001$ y $\delta_2 = 0.01$ para el modelo M-BEN y PM-BEN usando los datos de trigo para cada uno de los 4 ambientes.

6. Resultados y Discusión

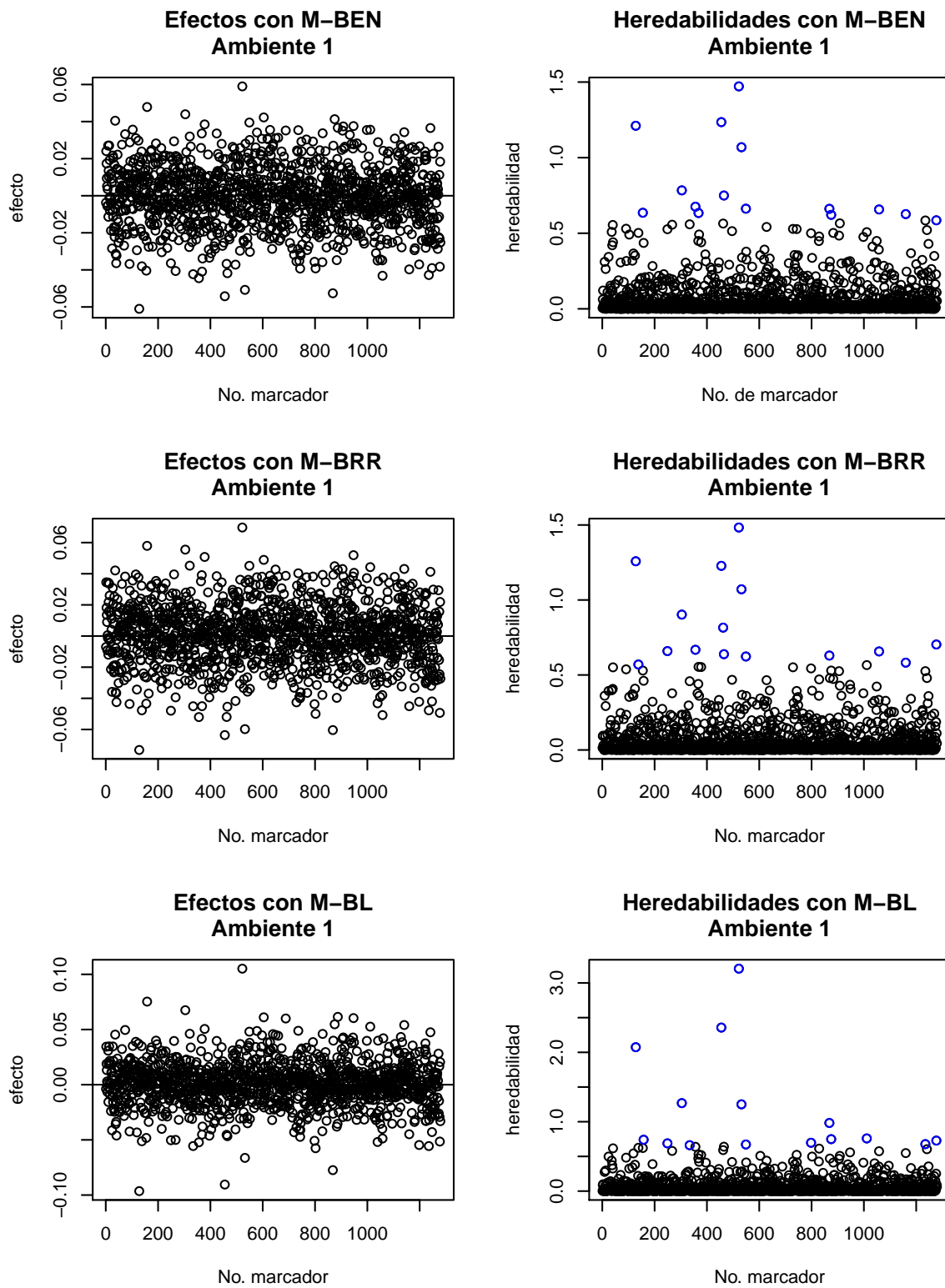


Figura 6.8: Comparación de los efectos y heredabilidades de los MM obtenidos con cada uno de los modelos sin pedigree para los datos de trigo en el ambiente 1.

6. Resultados y Discusión

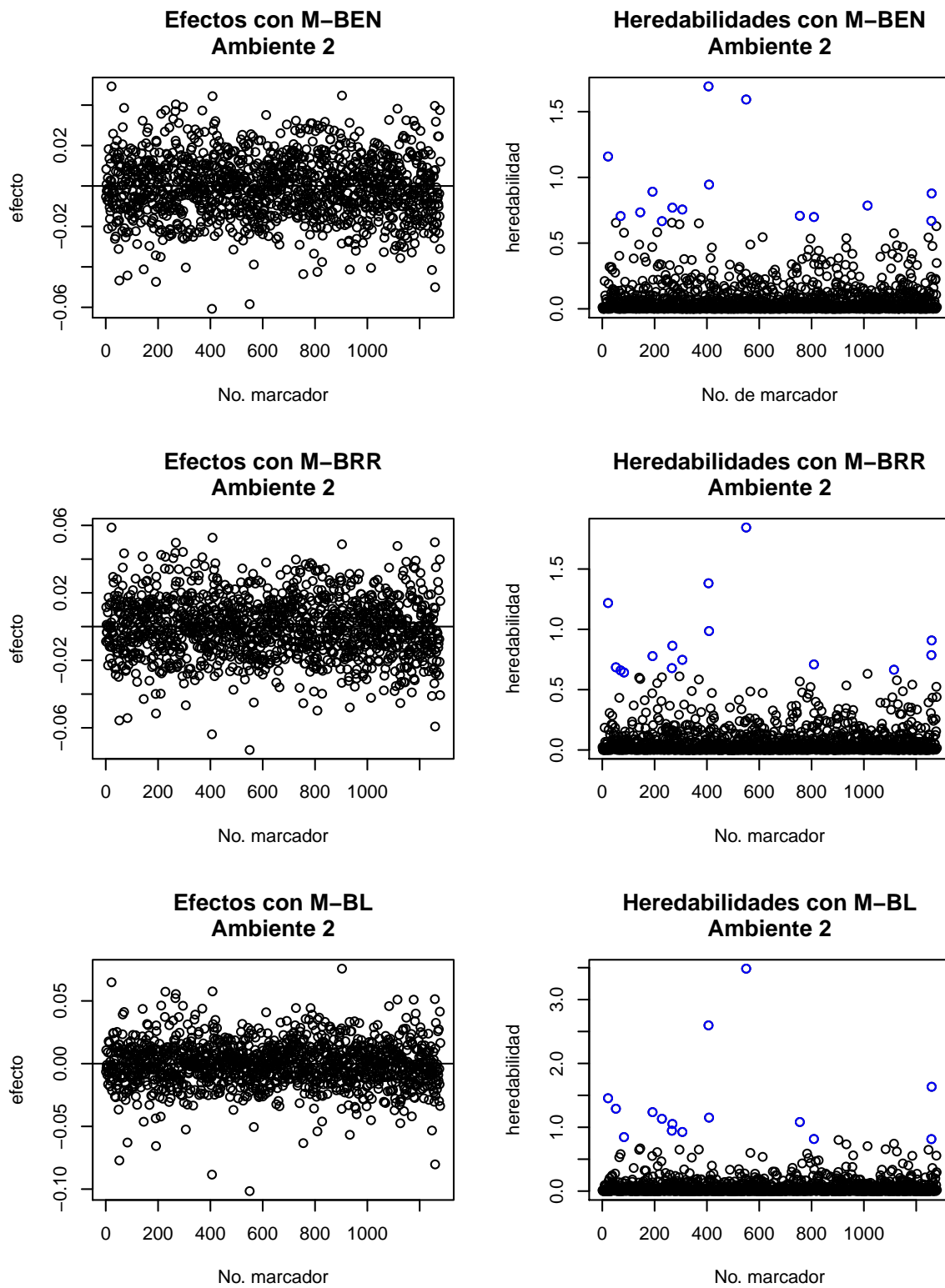


Figura 6.9: Comparación de los efectos y heredabilidades de los MM obtenidos con cada uno de los modelos sin pedigree para los datos de trigo en el ambiente 2.

6. Resultados y Discusión

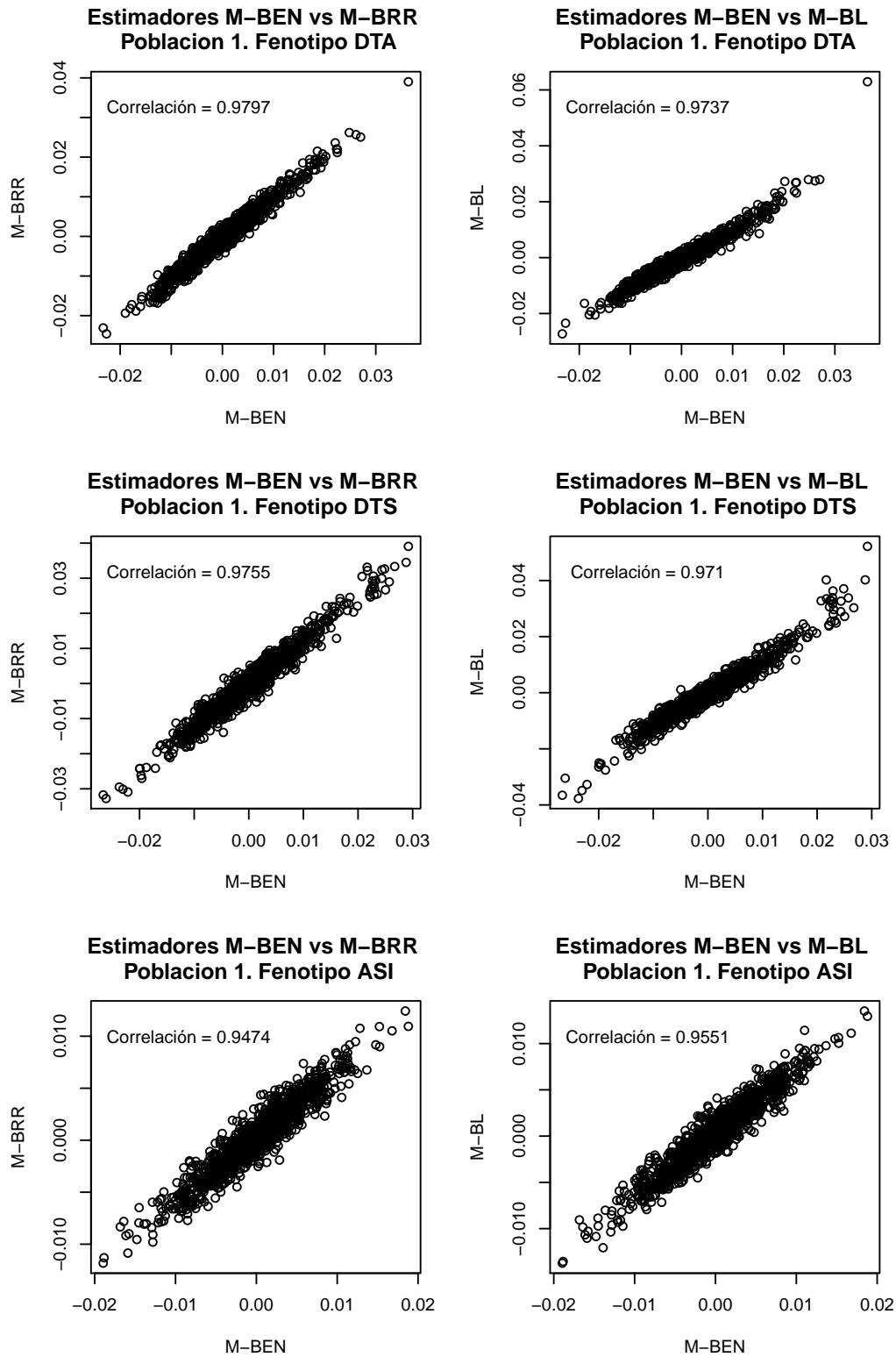


Figura 6.10: Comparación de los estimadores de los efectos usando los modelos M-BEN contra los obtenidos usando M-BL y M-BRR usando los datos de maíz para los 3 fenotipos de la población 1.

6. Resultados y Discusión

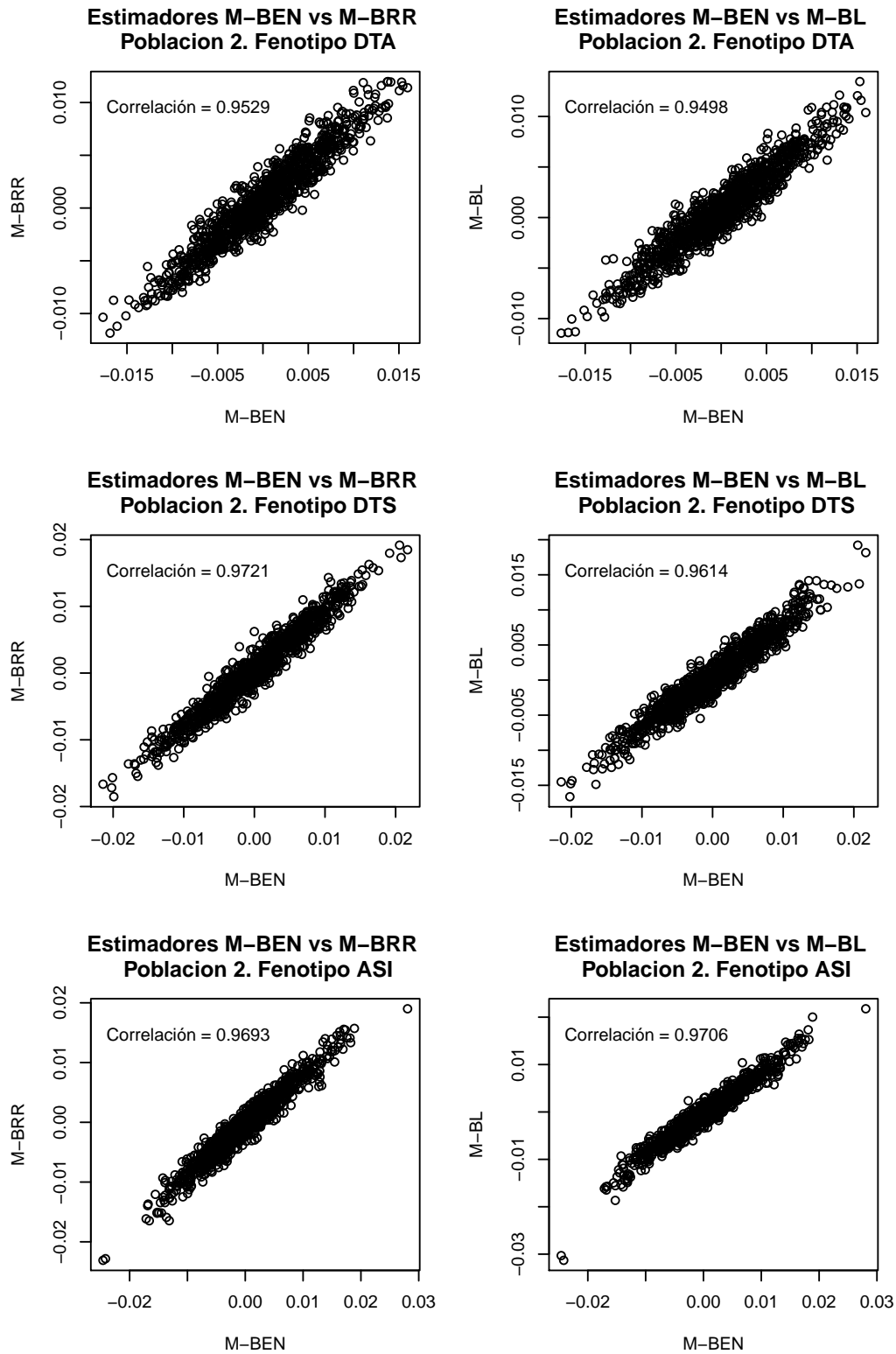


Figura 6.11: Comparación de los estimadores de los efectos usando los modelos M-BEN contra los obtenidos usando M-BL y M-BRR usando los datos de maíz para los 3 fenotipos de la población 2.

6. Resultados y Discusión

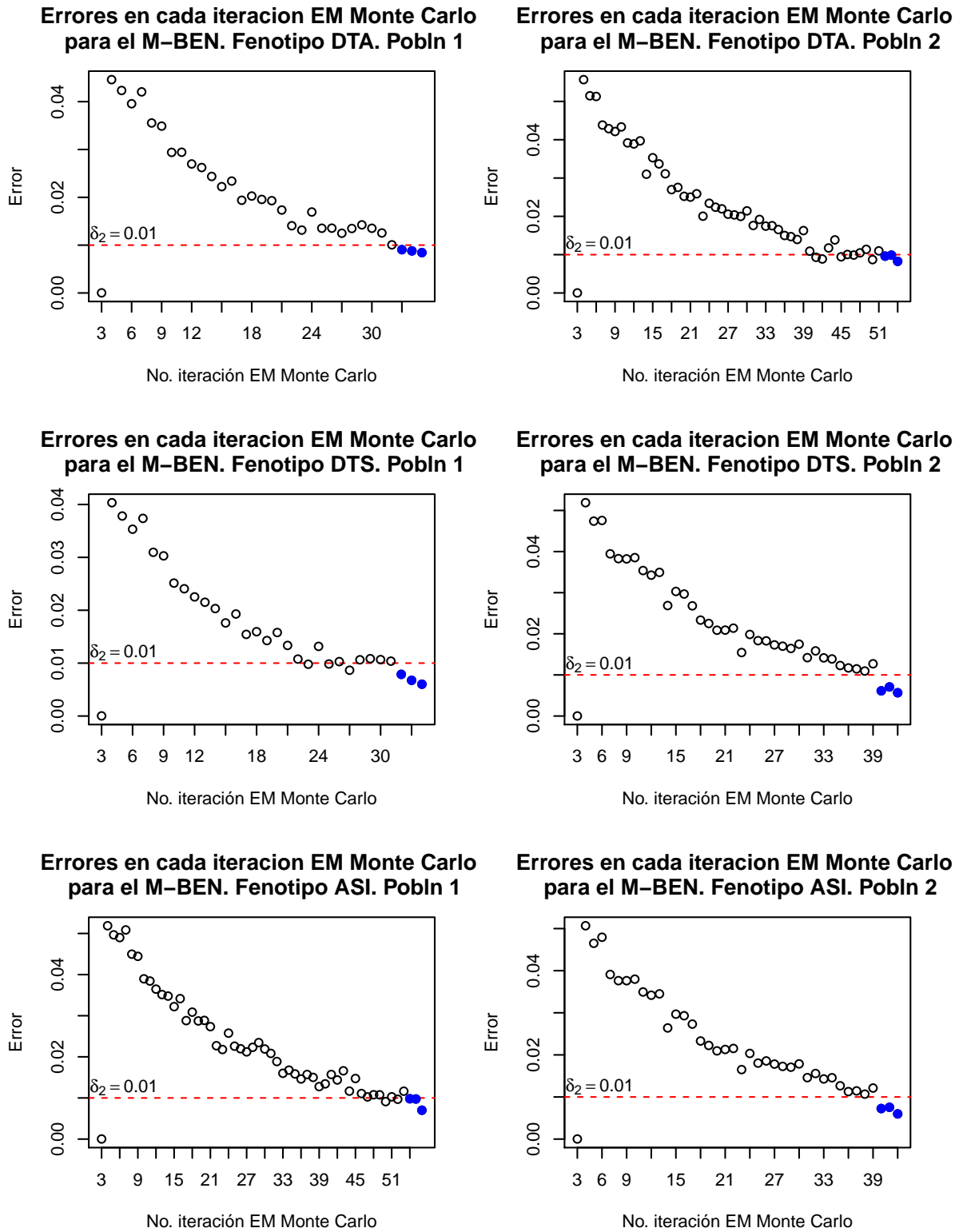


Figura 6.12: Errores en cada iteración EM Monte Carlo con $\delta_1 = 0.001$ y $\delta_2 = 0.01$ para el modelo M-BEN usando los datos de maíz para los 3 fenotipos de las poblaciones 1 y 2.

6. Resultados y Discusión

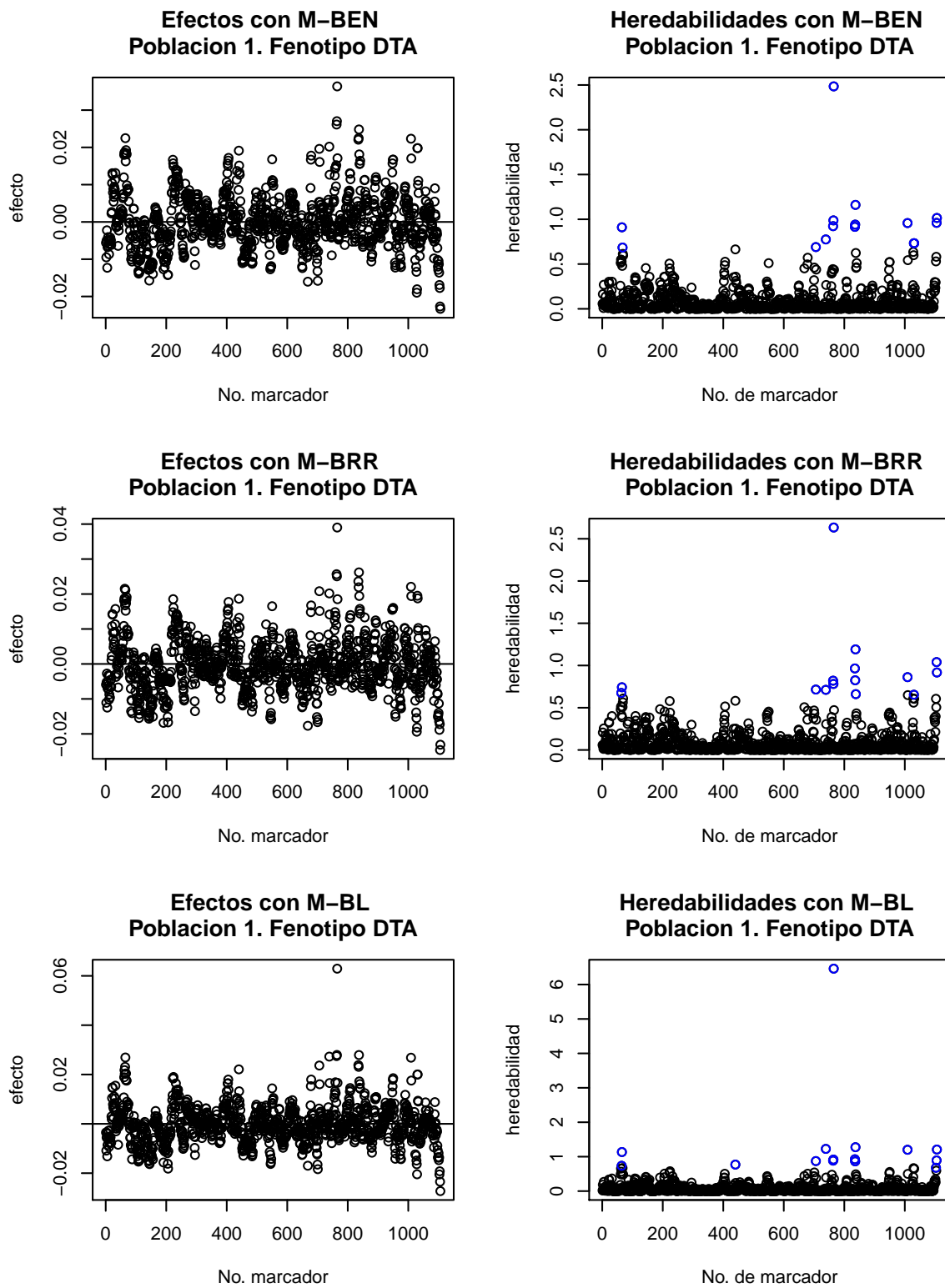


Figura 6.13: Comparación de los efectos y heredabilidades de los MM obtenidos con cada uno de los modelos para el fenotipo DTA de los datos de maíz en la población 1.

6. Resultados y Discusión

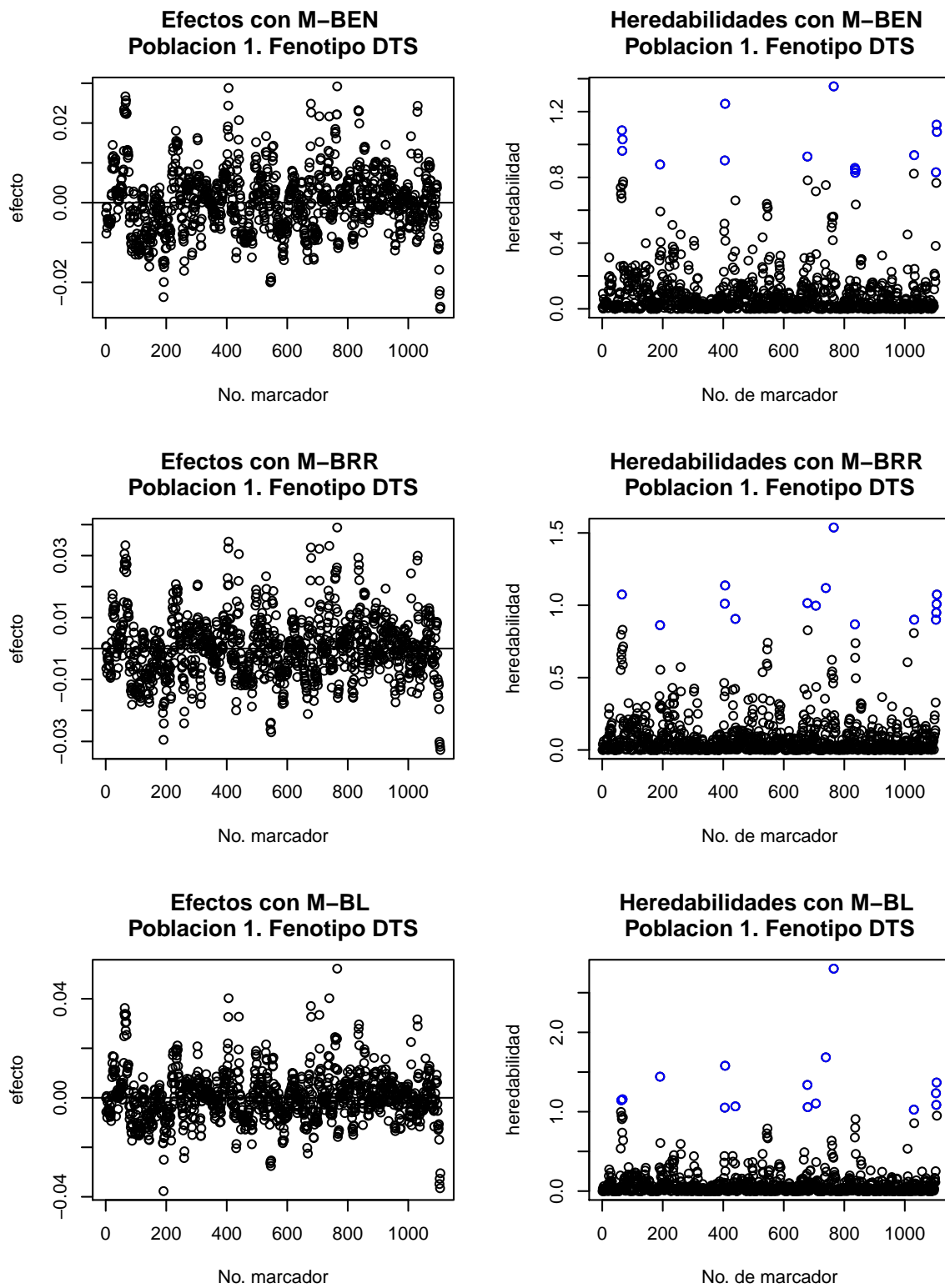


Figura 6.14: Comparación de los efectos y heredabilidades de los MM obtenidos con cada uno de los modelos para el fenotipo DTS de los datos de maíz en la población 1.

6. Resultados y Discusión

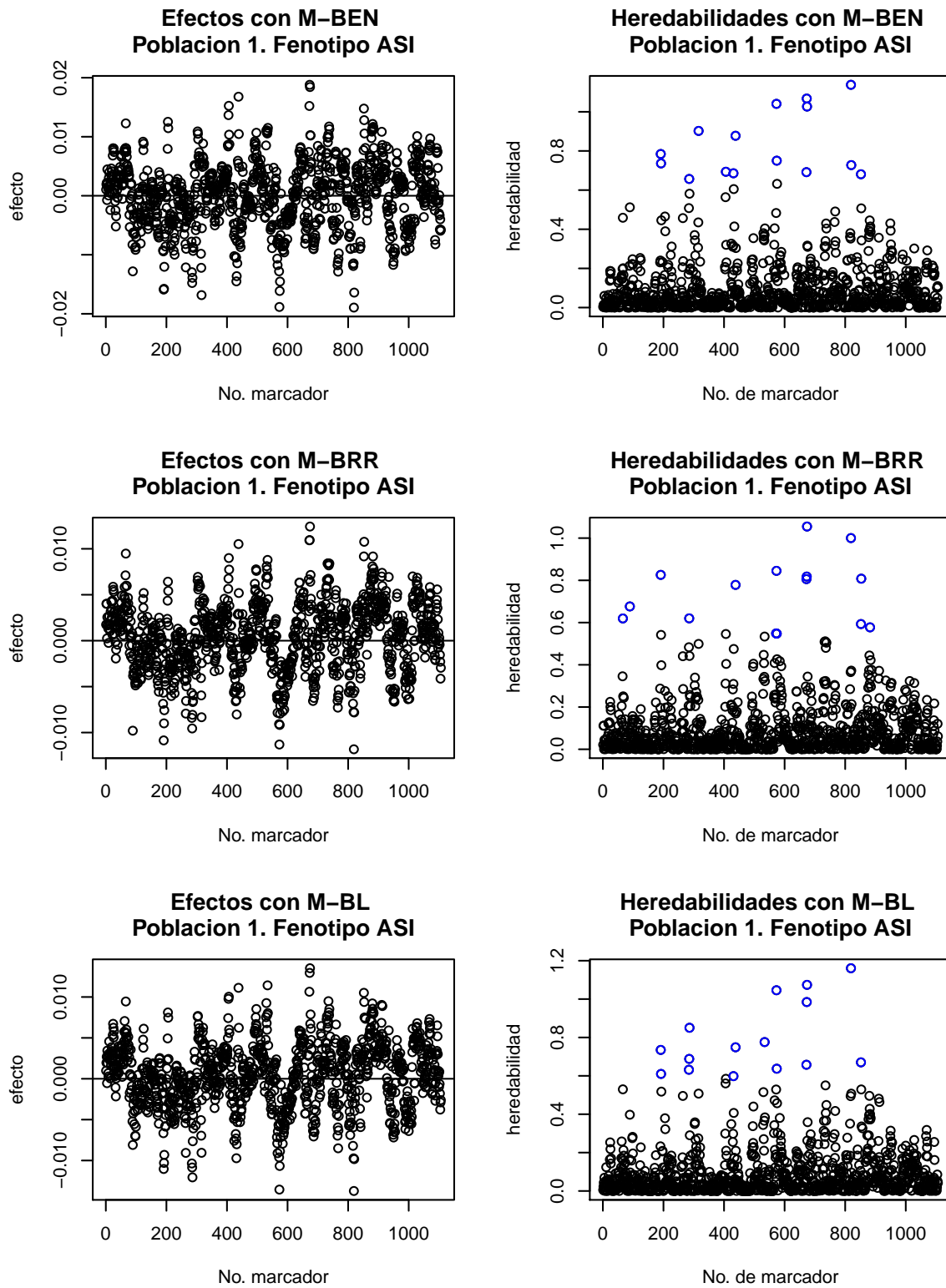


Figura 6.15: Comparación de los efectos y heredabilidades de los MM obtenidos con cada uno de los modelos para el fenotipo ASI de los datos de maíz en la población 1.

Capítulo 7

Conclusiones

Los resultados de este trabajo muestran que el modelo BEN es un método eficiente de estimación penalizada cuando $p \gg n$, tan bueno o mejor en predicción que los modelos BRR y BL. Este método puede ser muy útil en SG prediciendo valores fenotípicos futuros y seleccionando variables (MM) que están fuertemente ligados a algún fenotipo de interés.

Los análisis muestran que los estimadores de β obtenidos con el modelo BEN son más semejantes con los obtenidos con el BRR, lo que se refleja en un buen desempeño en predicción (rasgo característico de la BRR), no obstante, el BEN también contiene características del LASSO Bayesiano, por lo que posee la capacidad de selección de variables (característica del LASSO Bayesiano).

Como se aprecia en las correlaciones de las VC de la Tabla 6.2, la inclusión de un pedigree trae consigo una mejor predicción.

Dependiendo de los datos y para asegurar la convergencia del método, se deben utilizar valores iniciales apropiados para λ_1 y λ_2 para no tener problemas al momento de utilizar el algoritmo EM Monte Carlo.

Referencias

- Andrews, D. F. y Mallows, C. L. (1974). Scale Mixtures of Normal Distributions. *Journal of the Royal Statistical Society*, 36, 99–102.
- Buckler, E. S., Holland, J. M., Bradbury, P. J., Acharya, C. B., Brown, P. J., Brown, C., Ersoz, E., Flint-Garcia, S., Garcia, A., Glaubitz, J. C., Goodman, M. M., Harjes, C., Guill, K., Kroon, D. E., Larson, S., Lepak, N. K., Li, H., Mitchell, S. E., Pressoir, G., Peiffer, J. A., Oropeza, M., Rocheford, T. R., Romay, M. C., Romero, S., Salvo, S., Sanchez, H., da Silva, H. S., Sun, Q., Tian, F., Upadyayula, N., Ware, D., Yates, H., Yu, J., Zhang, Z., Kresovich, S. y McMullen, M. D. (2009). The Genetic Architecture of Maize Flowering Time. *Science*, 325, 714–718.
- Casella, G. y George, E. I. (1992). Explaining the Gibbs Sampler. *American Statistical Association*, 46, 167–174.
- 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 Society of America*, 182, 375–385.
- Efron, B., Hastie, T., Johnstone, I. y Tibshirani, R. (2004). Least Angle Regression. *The Annals of Statistics*, 32, 407–499.
- Fernando, R. L. y Grossman, M. (1989). Marker assisted selection using best linear unbiased prediction. *Genetics Selection Evolution*, 21, 467–477.
- Hoerl, A. E. y Kennard, R. W. (1970). Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12, 55–67.
- Kyung, M., Gill, J., Ghosh, M. y Casella, G. (2010). Penalized Regression, Standard Errors and Bayesian Lasso. *Journal of the International Society for Bayesian Analysis*, 5, 369–412.
- Levine, R. A. y Casella, G. (2001). Implementations of the Monte Carlo EM Algorithm. *American Statistical Association*, 10, 422–439.
- Levine, R. A. y Fan, J. J. (2004). An Automated (Markov Chain) Monte Carlo EM Algorithm. *Journal of Statistical Computation & Simulation*, 74, 349–360.
- Li, Q. y Lin, N. (2010). The Bayesian Elastic Net. *International Society for Bayesian Analysis*, 5, 151–170.

Referencias

- Meuwissen, T. H. E., Hayes, B. J. y Goddard, M. E. (2001). Prediction of Total Genetic Value Using Genome-Wide Dense Marker Maps. *Genetics Society of America*, 157, 1819–1829.
- Park, T. y Casella, G. (2008). The Bayesian Lasso. *Journal of the American Statistical Association*, 103, 681–686.
- 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. *Plant Genome*, 3(2), 106–116.
- R Development Core Team (2011). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.
- Sorensen, D. y Gianola, D. (2002). *Likelihood, Bayesian and MCMC Methods in Quantitative Genetics*. Springer, New York, primera edición.
- Tiang, F., Bradbury, P. J., Brown, P. J., Hung, H., Sun, Q., Flint-Garcia, S., Rocheford, T. R., McMullen, M. D., Holland, J. B. y Buckler, E. S. (2011). Genome-wide association study of leaf architecture in the maize nested association mapping population. *Nature Genetics*, 43, 159–162.
- Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal Of the Royal Statistical Society*, 58, 267–288.
- VanRaden, P. M. (2008). Efficient Methods to Compute Genomic Predictions. *American Dairy Science Association*, 91, 4414–4423.
- Yi, N. y Xu, S. (2008). Bayesian LASSO for Quantitative Trait loci Mapping. *Genetics Society of America*, 179, 1045–1055.
- Zou, H. y Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal Of the Royal Statistical Society*, 67, 301–320.

Anexos

Anexo A: Prueba del lema 1

Primeramente, note que:

$$\exp \left\{ -\frac{1}{2\sigma_\varepsilon^2} \left(\lambda_1 \sum_{j=1}^p |\beta_j| + \lambda_2 \sum_{j=1}^p \beta_j^2 \right) \right\} = \prod_{j=1}^p \exp \left\{ -\frac{1}{2\sigma_\varepsilon^2} (\lambda_1 |\beta_j| + \lambda_2 \beta_j^2) \right\}.$$

Posteriormente, como lo sugieren [Andrews y Mallows \(1974\)](#), para $a > 0$, se tiene que:

$$\frac{a}{2} \exp \{-a|z|\} = \int_0^\infty \frac{1}{\sqrt{2\pi s}} \exp \left\{ -\frac{1}{2} \frac{z^2}{s} \right\} \frac{a^2}{2} \exp \left\{ -\frac{1}{2} a^2 s \right\} ds.$$

Sea $a = \frac{\lambda_1}{2\sigma_\varepsilon^2}$ y $z = \beta_j$, entonces se tiene:

$$\exp \left\{ -\frac{\lambda_1}{2\sigma_\varepsilon^2} |\beta_j| \right\} \propto \int_0^\infty \frac{1}{\sqrt{s}} \exp \left\{ -\frac{1}{2} \frac{\beta_j^2}{s} \right\} \exp \left\{ -\frac{1}{2} \left(\frac{\lambda_1}{2\sigma_\varepsilon^2} \right)^2 s \right\} ds,$$

donde \propto denota que ambos lados difieren por un multiplicando que involucra a σ_ε^2 y posiblemente

algunas otras constantes. Luego se sigue que:

$$\begin{aligned}
 \exp \left\{ -\frac{1}{2\sigma_\varepsilon^2} (\lambda_1 |\beta_j| + \lambda_2 \beta_j^2) \right\} &\propto \int_0^\infty \frac{1}{\sqrt{s}} \exp \left\{ \frac{\beta_j^2}{2} \left(\frac{1}{s} + \frac{\lambda_2}{\sigma_\varepsilon^2} \right) \right\} \exp \left\{ -\frac{1}{2} \left(\frac{\lambda_1}{2\sigma_\varepsilon^2} \right)^2 s \right\} ds \\
 &= \int_0^\infty \sqrt{\frac{1}{s} + \frac{\lambda_2}{\sigma_\varepsilon^2}} \exp \left\{ \frac{\beta_j^2}{2} \left(\frac{1}{s} + \frac{\lambda_2}{\sigma_\varepsilon^2} \right) \right\} \frac{1}{\sqrt{1 + \frac{\lambda_2}{\sigma_\varepsilon^2} s}} \\
 &\quad \times \exp \left\{ -\frac{1}{2} \left(\frac{\lambda_1}{2\sigma_\varepsilon^2} \right)^2 s \right\} ds \\
 &\propto \int_1^\infty \sqrt{\frac{t}{t-1}} \exp \left\{ \frac{\beta_j^2}{2} \left(\frac{\lambda_2}{\sigma_\varepsilon^2} \frac{t}{t-1} \right) \right\} t^{-\frac{1}{2}} \exp \left\{ -\frac{1}{2\sigma_\varepsilon^2} \frac{\lambda_1^2}{4\lambda_2} t \right\} dt,
 \end{aligned}$$

donde en este último paso se realizó el cambio de variable $t = 1 + \frac{\lambda_2}{\sigma_\varepsilon^2} s$.

Anexo B: Rutinas en R-2.12.2 para implementar el Modelo Elastic Net con o sin pedigree

```
#####
#           Elastic Net Bayesiano Lasso           #
#                                                                 #
#####

# Carga las librerías MASS y SuppDist
library(MASS)
library(SuppDists)

# h function que es una Gama-Inversa(a,b)
h=function(x,a,b)
{
  (1/x)^(a+1)*exp(-b/x)*b^a/(gamma(a))
}

psi=function(t) 1/sqrt(t)*exp(-t)
Erf =function(x) 2 * pnorm(x * sqrt(2)) - 1
Erfc=function(x) 1-Erf(x)
Power=function(base, exponente) base^exponente
E=exp(1)

# Derivada parcial c/r lambda1 y lambda1 en la matriz Hessiana
d_lambda1_lambda1=function(theta,p,beta,sigma2,tau)
{
  lambda1=theta[1]
  lambda2=theta[2]
  sigma=sqrt(sigma2)
  c1=p*(-4*sqrt(lambda2)*sigma + Power(E,Power(lambda1,2))/
    (8.*lambda2*Power(sigma,2)))*
    sqrt(2*pi)*lambda1*Erfc(lambda1/(2.*sqrt(2)*sqrt(lambda2)*sigma))
  c2=8.*Power(E,Power(lambda1,2))/(4.*lambda2*Power(sigma,2))*pi*
    Power(lambda2,1.5)*Power(sigma,3)*
    Power(Erfc(lambda1/(2.*sqrt(2)*sqrt(lambda2)*sigma)),2)
  return(-(p/Power(lambda1,2))-c1/c2-sum(tau)/(4.*lambda2*Power(sigma,2)))
}

# Derivada parcial c/r lambda2 y lambda2 en la matriz Hessiana
d_lambda2_lambda2=function(theta,p,beta,sigma2,tau)
{
  lambda1=theta[1]
  lambda2=theta[2]
  sigma=sqrt(sigma2)
  c1=4*p*lambda1*sqrt(lambda2)*sigma-Power(E,Power(lambda1,2))

```

```

        /(8.*lambda2*Power(sigma,2))*p*sqrt(2*pi)*
        (Power(lambda1,2)-12*lambda2*Power(sigma,2))*
        Erfc(lambda1/(2.*sqrt(2)*sqrt(lambda2)*sigma))
c2=Power(E,Power(lambda1,2)/(4.*lambda2*Power(sigma,2)))*pi*
        Power(Erfc(lambda1/(2.*sqrt(2)*sqrt(lambda2)*sigma)),2)
c3=32*Power(lambda2,3.5)*Power(sigma,3)
return(lambda1*(c1/c2-8*lambda1*sqrt(lambda2)*sigma*sum(tau))/c3)
}

# Derivada parcial c/r lambda1 and lambda2 en la matriz Hessiana
d_lambda1_lambda2=function(theta,p,beta,sigma2,tau)
{
    lambda1=theta[1]
    lambda2=theta[2]
    sigma=sqrt(sigma2)
    c1=-4*p*lambda1*sqrt(lambda2)*sigma+Power(E,Power(lambda1,2)/
        (8.*lambda2*Power(sigma,2)))*p*
        sqrt(2*pi)*(Power(lambda1,2)-4*lambda2*
        Power(sigma,2))*Erfc(lambda1/(2.*sqrt(2)*sqrt(lambda2)*sigma))
    c2=Power(E,Power(lambda1,2)/(4.*lambda2*Power(sigma,2)))*pi*
        Power(Erfc(lambda1/(2.*sqrt(2)*sqrt(lambda2)*sigma)),2)
    return((c1/c2+4*lambda1*sqrt(lambda2)*sigma*sum(tau))/
        (16*Power(lambda2,2.5)*Power(sigma,3)))
}

myQ2=function(theta,beta,sigma2,tau)
{
    p=ncol(beta)
    m=nrow(beta)
    S=matrix(0,nrow=2,ncol=2)
    for(i in 1:m)
    {
        S[1,1]=S[1,1]+d_lambda1_lambda1(theta,p,beta[i,],sigma2[i],tau[i,])
        S[1,2]=S[1,2]+d_lambda1_lambda2(theta,p,beta[i,],sigma2[i],tau[i,])
        S[2,2]=S[2,2]+d_lambda2_lambda2(theta,p,beta[i,],sigma2[i],tau[i,])
    }
    S[1,1]=S[1,1]/m
    S[1,2]=S[1,2]/m
    S[2,1]=S[1,2]
    S[2,2]=S[2,2]/m
    S
}

# Submuestreo en el paso 4a del Algoritmo EM-Monte Carlo
subsampling=function(nu=1,d=0.5,m)
{
    tk=0
    xk=numeric()

```



```

    k=1
    while(tk<=m)
    {
        xk=c(xk,1+rpois(1,lambda=nu*(k^d)))
        k=k+1
        tk=sum(xk)
    }
    if(tk>m){xk=xk[-length(xk)]}
    tk=cumsum(xk)
    return(tk)
}

logL=function(theta,p,beta,sigma2,tau)
{
    lambda1=theta[1]
    lambda2=theta[2]
    c1=lambda1/(2*sqrt(2*lambda2*sigma2))
    c2=(lambda1^2*sum(tau)+4*lambda2^2*
        sum(beta^2*tau/(tau-1)))/(8*lambda2*sigma2)
    p*log(lambda1)-p*log(sqrt(pi)*Erfc(c1))-c2
}

# Función a minimizar
# theta=c(lambda1,lambda2),
# beta: matriz con p columnas y M filas (salida del muestreador de Gibbs)
# sigma2: vector de longitud M (salida del muestreador de Gibbs)
# tau: matriz con p columnas y M filas (salida del muestreador de Gibbs)
# wt: vector de pesos
Q=function(theta,beta,sigma2,tau,wt=NULL)
{
    cat("Optimizing... lambda1=",theta[1],"lambda2=",theta[2],"\n")
    p=ncol(beta)
    m=nrow(beta)
    ans=-Inf
    if(is.null(wt)) wt=rep(1,m)
    if(theta[1]>0 & theta[2]>0)
    {
        suma=0
        for(i in 1:m)
        {
            suma=suma+
                wt[i]*logL(theta,p,beta[i,],sigma2[i],tau[i,])
        }
        ans=-suma/sum(wt)
    }
    return(ans)
}

gr=function(theta,p,beta,sigma2,tau)

```

```

{
  lambda1=theta[1]
  lambda2=theta[2]
  c1=1/(sqrt(pi)*Erfc(lambda1/(2*sqrt(2*lambda2*sigma2))))*
    psi((lambda1^2)/(8*sigma2*lambda2))*1/sigma2
  c2=tau/sigma2
  c3=tau*beta^2/((tau-1)*sigma2)
  g1=p/lambda1+p*lambda1/(4*lambda2)*
    c1-lambda1/(4*lambda2)*sum(c2)
  g2=-p*lambda1^2/(8*lambda2^2)*
    c1-1/2*sum(c3)+lambda1^2/(8*lambda2^2)*sum(c2)
  return(c(g1,g2))
}

# Función Gradiente promedio
gradient_avg=function(theta,beta,sigma2,tau,wt=NULL)
{
  p=ncol(beta)
  m=nrow(beta)
  if(is.null(wt)) wt=rep(1,m)
  sums=rep(0,2)
  for(i in 1:m) sums=sums+wt[i]*
    gr(theta,p,beta[i,],sigma2[i],tau[i,])
  return(-1*sums/sum(wt))
}

# Función Gradiente
gradient=function(theta,beta,sigma2,tau)
{
  p=ncol(beta)
  m=nrow(beta)
  a=matrix(NA,m,2)
  for(i in 1:m) a[i,]=gr(theta,p,beta[i,],sigma2[i],tau[i,])
  return(a)
}

# Gibbs: función
# B: número de de iteraciones
# Burnin: periodo de calentamiento
# y: vector con la variable respuesta, NAs permitidos
# X: matriz diseño
# lambda1, lambda2: parámetros para el modelo Elastic Net
# A: matriz de pedigree
# dfu: grados de libertad para la a priori del componente de varianza u
# su: parámetro de escala para la a priori del componente de varianza u
Gibbs=function(B=10000,Burnin=5000,y,X,lambda1=1,lambda2=1,
  A=NULL,dfu=4,su=1,minAbsBeta=1e-09,optim=TRUE)

```

```

{
  # algunas constantes
  p=ncol(X)
  n=length(y)

  XtX=crossprod(X)
  sXtX=as.vector(XtX)
  dXtX=as.vector(diag(XtX))
  index=(0:(p-1))*(p+1)+1

  if(!is.null(A))
  {   Ainv=chol2inv(chol(A))
      sqrtAinv=t(chol(Ainv))
      q=nrow(A)
      pedigree=TRUE
    }
  else
  {   pedigree=FALSE
    }

  indexNA=is.na(y)
  if(sum(indexNA)==0)
  {   missing=FALSE
      yNA=NULL
    }
  else
  {   cat("The dataset contains missing values\n");
      missing=TRUE
      nNA=sum(indexNA)
      y[indexNA]=mean(y,na.rm=TRUE)
      yNA=matrix(NA,nrow=B,ncol=nNA)
    }

  # Valores iniciales
  sigma02=var(y)/2           #varE
  sigma02u=sigma02/5        #varU
  beta0=rnorm(n=p,0,sd=0.05) #Beta
  u=rnorm(n=n,0,sd=sqrt(sigma02u)) #Efecto infinitesimal u
  tau0=rep(NA,p)           #tau

  # Objetos R para guardar las simulaciones
  betas=matrix(NA, nrow=B,ncol=p)
  tau=matrix(NA,nrow=B,ncol=p)
  sigma2=rep(NA,B)
  sigma2u=rep(NA,B)
  U=matrix(NA,nrow=B,ncol=n)
  if(is.null(A) & !optim & !missing)

```

```

{   BLOD=matrix(NA,nrow=B,ncol=p)
}
else
{   BLOD=NULL
}

time <- proc.time()[3]

# Muestreador de Gibbs
for(i in 1:B)
{   # Muestrea de tau | y, sigma2, beta
    mu=sqrt(lambda1)/(2*lambda2*abs(beta0))
    lambda=rep(lambda1/(4*lambda2*sigma02),p)
    tmp=1+(1/rinvGauss(n=p, mu, lambda))
    # Corroborando
    if(any(tmp<1) | any(is.infinite(tmp)))
    {   tau[i,]=tau0
        cat("Warning, tau was not updated in iteration ",
            i," (Numeric problems)\n")
    }
    else
    {   tau[i,]=tau0=tmp
    }

    # Muestrea de beta | y, sigma2, tau
    sXtX[index]=dXtX+lambda2*tau0/(tau0-1)
    if(pedigree)
    {   r=as.vector(crossprod(y-u,X))
    }
    else
    {   r=as.vector(crossprod(y,X))
    }
    betas[i,]=beta0=.Call("sample_betas",p,sXtX,r,
                        beta0,sigma02,minAbsBeta)[[1]]

    # Debug
    if(sum(is.na(beta0))>0)
    {   tau0<<-tau0
        lambda2<<-lambda2
        y<<-y
        u<<-u
        X<<-X
        beta_ante<<-betas[i-1,]
        beta0<<-beta0
        sigma02<<-sigma02
        r<<-r
        sXtX<<-sXtX
    }
}

```

```

        betas<<-betas
    }

    # Muestrea de sigma2|y, beta, tau, u
    a=n/2+p
    e1=as.vector(X%*%beta0)
    if(pedigree)
    {
        b=(sum((y-e1-u)^2)+lambda2*
            sum(tau0*beta0^2/(tau0-1))+
            lambda1^2/(4*lambda2)*sum(tau0))/2
    }
    else
    {
        b=(sum((y-e1)^2)+lambda2*
            sum(tau0*beta0^2/(tau0-1))+
            lambda1^2/(4*lambda2)*sum(tau0))/2
    }

    flag=TRUE
    while(flag)
    {
        unif=runif(1)
        z=1/rgamma(1,shape=a,rate=b)
        gama=pgamma((lambda1^2)/(8*z*lambda2),
                    shape=1/2,scale=1,lower.tail=FALSE,log=TRUE)
        if(log(unif)<=p*(lgamma(1/2)-(lgamma(1/2)+gama)))
        {
            sigma2[i]=sigma02=z
            flag=FALSE
        }
        else
        {
            cat("Reject candidate...\n") # Rechaza candidato
        }
    }

    if(pedigree)
    {
        # Muestrea de u
        C=sigma02/sigma02u*Ainv
        dC=diag(C)
        diag(C)=dC+1
        u=.Call("sample_betas",n,as.vector(C),
                as.vector(y-e1),u,sigma02,minAbsBeta)[[1]]
        U[i,]=u

        # Muestrea de sigma02u
        scale=as.numeric(su+sum((crossprod(u,sqrtAinv))^2))
        sigma02u=scale/rchisq(1,df=dfu+q)
        sigma2u[i]=sigma02u
    }

```

```
# Actualiza valores faltantes
if(missing)
{
  if(pedigree)
  {
    yNA[i,]=y[indexNA]=e1[indexNA]+u[indexNA]+
      rnorm(n=nNA,mean=0,sd=sqrt(sigma02))
  }
  else
  {
    yNA[i,]=y[indexNA]=e1[indexNA]+
      rnorm(n=nNA,mean=0,sd=sqrt(sigma02))
  }
}

# BLOD
if(is.null(A) & !optim & !missing)
{
  for(k in 1:p)
  {
    sigma2_k=1/(n-2)*sum((y-e1+X[,k]*beta0[k])^2)
    BLOD[i,k]=2/log(10)*(n/2*log(sigma2_k/sigma02)-
      1/(2.0*sigma02)*sum((y-e1)^2)+(n-2)/2)
  }
}

# Imprime historial de iteración
tmp <- proc.time()[3]
valores=c(i,round(tmp-time,3),round(sigma02,4),
  round(sigma02u,4),round(lambda1,4),round(lambda2,4))
cat(paste(c("Iter: "," time/Iter: "," varE: "," varU: ",
  " lambda1: "," lambda2:"),valores))
cat("\n")
time=tmp
}
if(optim)
{
  return(list(beta=betas[-c(1:Burnin)],,
    sigma2=sigma2[-c(1:Burnin)],tau=tau[-c(1:Burnin),]))
}
else
{
  if(pedigree)
  {
    return(list(beta=betas[-c(1:Burnin)],,
      sigma2=sigma2[-c(1:Burnin)],sigma2u=sigma2u[-c(1:Burnin)],
      u=U[-c(1:Burnin),],yNa=yNA[-c(1:Burnin),]))
  }
  else
  {
    return(list(beta=betas[-c(1:Burnin)],,
```

```
sigma2=sigma2[-c(1:Burnin)],yNa=yNA[-c(1:Burnin),],
BL0D=BL0D[-c(1:Burnin),])
}
}
}
```

Anexo C: Rutinas en R-2.12.2 para implementar el algoritmo EM Monte Carlo de Levine y Fan

```
#####
#           Algoritmo de Levine y Fan           #
#           para seleccionar lambda1 y lambda2  #
#####

# Paso 1. Calentamiento para el algoritmo
m=1000
b=20
Psi=c(lambda1,lambda2)
for(i in 1:b)
{
  cat("i=",i,"\n")
  run=Gibbs(B=m+1,Burnin=1,X=X,y=y,lambda1=Psi[1],lambda2=Psi[2],A=A)
  output=optim(Psi,Q,gradient_avg,method="BFGS",
              beta=run$beta,sigma2=run$sigma2,tau=run$tau,wt=NULL)
  if(output$convergence==0)
  { Psi=output$par
  }
  else
  {
    stop("Burn-in: Can not optimize Q function\n")
  }
}

# Paso 2. Inicializa Psi, m ya seleccionado, genera u1,...,um~g(u|y,Psi0)
Psi0=Psir=Psi
run=Gibbs(B=m+1,Burnin=1,X=X,y=y,
         lambda1=Psi0[1],lambda2=Psi0[2],A=A)

# Repite pasos 3 a 7 hasta convergencia
maxerror=0.01
maxiter=150
error_history=matrix(NA,nrow=maxiter,ncol=2)
error1_history=rep(NA,maxiter)
lambdas=matrix(NA,nrow=maxiter,ncol=2)
lambdas_subsampling=matrix(NA,nrow=maxiter,ncol=2)
iter=1
lambdas[iter,]=Psir
m_history=rep(NA,maxiter)
continue=TRUE

while(continue & iter<maxiter)
{ # 4 y 5 - Pasos E y M
```



```

cat("iter=",iter,"\n")
cat("Optimizing parameters for the complete sample: \n")
output=optim(Psir,Q,gradient_avg,method="BFGS",
             beta=run$beta,sigma2=run$sigma2,tau=run$tau,wt=NULL)

if(output$convergence==0)
{
  iter=iter+1
  lambdas[iter,]=Psir=output$par

  # 6 Error de estimation MC

  # a) Submuestras al tiempo t
  tk=subsampling(nu=1,d=0.5,m=m)
  Nm=length(tk)

  # b) y c)
  cat("Optimizing parameters for the subsample: \n")
  output_subsampling=optim(Psir,Q,gradient_avg,method="BFGS",
                           beta=run$beta[tk,],
                           sigma2=run$sigma2[tk],
                           tau=run$tau[tk,],wt=NULL)

  if(output_subsampling$convergence==0)
  {
    Psir_subsampling=lambdas_subsampling[iter,]=output_subsampling$par

    # Calculando Sigma
    rhs=myQ2(Psir,beta=run$beta[tk,],
            sigma2=run$sigma2[tk],tau=run$tau[tk,])
    g=gradient(Psir,run$beta,run$sigma2,run$tau)
    mu=colSums(g)/m
    ans=matrix(0,2,2)
    for(z in 1:nrow(g))
    {
      ans=ans+tcrossprod(as.matrix(g[z,]-mu))
    }
    ans=ans/m
    Sigma=solve(rhs)%*%ans%*%solve(rhs)

    if(iter>2)
    {
      error_history[iter,]=abs(lambdas[iter,]-
                              lambdas[iter-1,])
      error_history[iter,1]=error_history[iter,1]/
                            (lambdas[iter-1,1]+0.001)
      error_history[iter,2]=error_history[iter,2]/
                            (lambdas[iter-1,2]+0.001)
      error1_history[iter]=max(error_history[iter,])
    }
  }
}

```

```
        if(iter>4)
        {
            if(max(error1_history[c(iter,iter-1,iter-2)])<maxerror)
                continue=FALSE
        }
        diff=as.matrix(lambdas_subsampling[iter,]-
                       lambdas_subsampling[iter-1,])
        fc=as.numeric(t(diff)%*%solve(Sigma)%*%diff)
        if(Nm*fc<qchisq(0.90,2))
        {
            m=as.integer(4/3*m)
        }
        if(continue)
            run=Gibbs(B=m+1,Burnin=1,X=X,y=y,
                    lambda1=Psir[1],lambda2=Psir[2],A=A)
    }
    m_history[iter]=m
}
else
{
    stop("Subsampling optimization loop:
        Can not optimize Q_Nm function\n")
}
}
else
{
    stop("Main optimization loop:
        Can not optimize Q function\n")
}
}

# Corrida final después de optimizar lambda1 y lambda2
run=Gibbs(B=30000,Burnin=25000,X=X,y=y,lambda1=Psir[1],
        lambda2=Psir[2],A=A,optim=FALSE)
```

Anexo D: Rutinas en lenguaje C para implementar el muestreo de β y u .

```

#include <R.h>
#include <Rmath.h>
#include <Rdefines.h>

SEXP sample_betas(SEXP q,SEXP C,SEXP r,SEXP beta,SEXP sigma2e,SEXP minAbsBeta)
{
    int i,j,cols;
    double suma,mu,s2,varE,smallBeta;
    double *pC, *pr, *pbeta, *ci;
    SEXP list;

    cols=INTEGER_VALUE(q);
    varE=NUMERIC_VALUE(sigma2e);
    smallBeta=NUMERIC_VALUE(minAbsBeta);

    PROTECT(C=AS_NUMERIC(C));
    pC=NUMERIC_POINTER(C);

    PROTECT(r=AS_NUMERIC(r));
    pr=NUMERIC_POINTER(r);

    PROTECT(beta=AS_NUMERIC(beta));
    pbeta=NUMERIC_POINTER(beta);

    ci=(double *) R_alloc(cols,sizeof(double));

    GetRNGstate();
    for(j=0; j<cols; j++)
    {
        suma=0;
        for(i=0; i<cols;i++)
        {
            ci[i]=pC[i+j*cols];
            suma+=ci[i]*pbeta[i];
        }
        suma-=ci[j]*pbeta[j];
        mu=(pr[j]-suma)/ci[j];
        s2=varE/ci[j];
        pbeta[j]=mu + sqrt(s2)*norm_rand();

        if(fabs(pbeta[j])<smallBeta)
        {
            pbeta[j]=smallBeta;
        }
    }
}

```

```
    }  
}  
  
// Creando una lista con 1 vector de elementos:  
PROTECT(list = allocVector(VECSXP, 1));  
// Agregando el vector bL a la lista:  
SET_VECTOR_ELT(list, 0, beta);  
  
PutRNGstate();  
  
UNPROTECT(4);  
  
return(list);  
}
```

Para poder usar el código C en R se debe crear una biblioteca compartida (.dll en Windows o .so en UNIX), para más detalles ver el manual “Writing R Extensions” incluido en los manuales de [R](#). Esto se puede realizar usando la rutina “R CMD SHLIB” incluida en el paquete [R](#) desde la línea de comandos del sistema operativo. Esto dará como resultado un archivo llamado “sampleBetas.dll” o “sampleBetas.so”. Una vez creada la biblioteca, las funciones se pueden usar usando combinando las rutinas “dyn.load” y “.Call”.

```
dyn.load("sampleBetas.dll")
```

Al terminar de usar el código, se corre la línea siguiente:

```
dyn.unload("sampleBetas.dll")
```