COLEGIO DE POSTGRADUADOS

INSTITUCIÓN DE ENSEÑANZA E INVESTIGACIÓN
EN CIENCIAS AGRÍCOLAS

CAMPUS MONTECILLO

SOCIOECONOMÍA, ESTADÍSTICA E INFORMÁTICA
ESTADÍSTICA

Funciones liga no convencionales para modelos de conteo inflados por ceros

Francisco Ariel Vázquez Chávez

T E S I S

PRESENTADA COMO REQUISITO PARCIAL PARA
OBTENER EL GRADO DE:

MAESTRO EN CIENCIAS

MONTECILLO, TEXCOCO, EDO. DE MÉXICO
2016

La presente tesis titulada: **Funciones liga no convencionales para modelos de conteo inflados por ceros**, realizada por el alumno: **Francisco Ariel Vázquez Chávez**, 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. Ciro Velasco Cruz

ASESOR

Dr. Javier Suárez Espinosa

ASESOR

Dr. Mario Alberto Vázquez Peña

Montecillo, Texcoco, Estado de México, Abril de 2016.

Funciones liga no convencionales para modelos de conteo inflados por ceros

Francisco Ariel Vázquez Chávez, MC

Colegio de Postgraduados, 2016

En el presente trabajo se propone utilizar, en el modelo ZIP (Zero-Inflated Poisson), las ligas Normal Sesgada (NS) con parámetro ν y Aranda-Ordaz 2 ($AO2$) con parámetro τ. El modelo *ZIP* es ampliamente utilizado cuando la variable respuesta Y presenta más ceros que los que se esperarían cuando se supone que tiene distribución Poisson. El objetivo es modelar la probabilidad p de los ceros estructurales usando las funciones liga NS y $AO2$. Las ligas *probit* y *media normal* son casos particulares de la liga NS cuando $\nu = 0$ y $\nu \to \infty$, mientras que la *logit* y la liga *cloglog* se obtienen cuando $\tau = 1$ y $\tau \to 0$ repectivamente. Tanto la liga NS como $AO2$ son flexibles ya que permiten modelar a p de forma asimétrica. El problema de estimación se aborda utilizando el algoritmo Esperanza-Maximización (EM). El algoritmo se implementa en el programa R. Se presentan algunos resultados de simulación y un ejemplo de aplicación con datos reales.

Palabras clave: Regresión Poisson, Algoritmo EM, Exceso de ceros.

Non-conventional link function for Zero-inflated count models

Francisco Ariel Vázquez Chávez, MC

Colegio de Postgraduados, 2016

In this work, we propose to use in the Zero-Inflated Poisson (ZIP) model, the following link functions: *SN* (Skew-normal) with parameter ν and Aranda-Ordaz 2 (*AO2*) with parameter τ. The ZIP model is widely used when the response variable Y has more zeros than expected under the Poisson model. The goal is to model the probability p of structural zeros using *SN* and *AO2* link functions. Suggested link functions are generalizations of *probit* and half normal link function which are obtained when its shape parameter ν is equal to 0 and $\nu \to \infty$, whereas the *logit* and *cloglog* link functions are obtained when $\tau = 1$ and $\tau \to 0$ respectively. Link functions *SN* and *AO2* are flexible because they model probability p in a skewed form. Inference problems can be solved using the Expectation-Maximization (EM) algorithm. This algorithm is implemented in R software. We present simulation results and an application with real data.

Key words: Poisson regression, EM algorithm, Zero-inflated.

AGRADECIMIENTOS

Quiero expresar mis más sinceros agradecimientos al Colegio de Postgraduados, que me brindó la oportunidad de estudiar la maestría, ya que significó un cambio muy importante en mi vida y fue una etapa que disfruté mucho.

También a todos mis profesores del Colegio de Postgraduados por haberme transmitido su invaluable conocimiento, en especial a los integrantes de mi Consejo Particular:

Dr. Paulino Pérez Rodríguez

Dr. Ciro Velasco

Dr. Javier Suárez

Dr. Mario Alberto Vázquez

También me siento muy agradecido con mis compañeros de clase (Beres, Ochoa, Alejandro, Ivonne, Nancy, Omar y Julio, Humberto), y personal administrativo que me apoyaron en mi estancia durante la maestría.

Agradezco a CONACYT por el apoyo económico brindado durante mis estudios.

DEDICATORIA

A mis padres Estela Chávez Sánches y Francisco Vázquez Lugo por su apoyo y amor incondicional, ya que sin ellos no hubiera sido posible haber estado aqui.

A mi hermana Isabel Vázquez Chávez, quien siempre sabe como hacer frente a los retos y adversidades de la vida.

A mi sobrina Alondra Zoé.

A mis amigos de esta maravillosa etapa de mi vida: Alejandro Ramírez, Omar, Julio, Nancy y Nallely.

◇

Contenido

Lista de tablas

Lista de figuras

Capítulo 1

Introducción

En muchas aplicaciones se utilizan variables discretas o datos de conteos, debido a que es de interés conocer, en promedio, el número de veces que ocurre un evento en un periodo determinado. Generalmente, para modelar los conteos se suele utilizar la regresión Poisson, la cual supone que la variable de interés Y tiene distribución Poisson cuya media λ puede depender de otras covariables $X_1, X_2, \ldots, X_p$ fijas con $p < n$, siendo n el tamaño de muestra. El modelo de regresión Poisson es muy restrictivo para datos de conteos debido a un fuerte supuesto de que la media y varianza de los conteos son iguales.

El problema de datos con exceso de ceros aparecen naturalmente en varias disciplinas, como por ejemplo en ingeniería industrial, economía, salud pública, etc. Naya *et al.* (2008) estudiaron los factores que producen manchas negras en la lana de borrego, Mouatassim y Ezzahid (2012) modelaron el número de reclamos en un esquema de seguros de salud privado en Marruecos, Lambert (1992) estudió el número de artículos defectuosos en la manufactura; cuando una máquina está desalineada, se producen artículos defectuosos con distribución Poisson (conteos poisson) pero si la máquina funciona perfectamente entonces se producen productos perfectos (ceros estructurales), entre otros. El modelo de regresión Poisson subestima la probabilidad de que un conteo sea cero de lo que se observó en la muestra (Cameron y Trivedi, 2005). El exceso de ceros en los conteos también conduce al problema de *sobredispersión* en el cual la varianza es mayor a la media de los conteos.

Existen diversas metodologías para abordar el problema de exceso de ceros. Una de ellas es la regresión *hurdle* (Mullahy, 1986), la cual considera que la variable respuesta puede ser cero o que proviene de una distribución Poisson truncada a valores positivos; este modelo supone una única fuente que produce los conteos de ceros. Los modelos *hurdle* tiene una interpretación que refleja una toma de decisión en dos etapas, por lo que son muy populares en áreas como econometría (Cameron y Trivedi, 2005). Heilbron (1989, 1994) propuso modelos de conteos con exceso de ceros, incluyendo la regresión Poisson. Lambert (1992) propuso la regresión *ZIP* (Zero-inflated Poisson, por sus siglas en inglés)

para modelar conteos con exceso de ceros. Esta regresión supone que los ceros ocurren con probabilidad $p_i,\ i = 1, 2, \ldots, n$ y los conteos ocurren con probabilidad $1 - p_i,\ i = 1, 2, \ldots, n$ y se pueden modelar con una variable aleatoria Poisson con media $\lambda_i,\ i = 1, 2, \ldots, n$, por lo que los coeficientes se estiman con modelos de regresión para variables binarias y regresión Poisson; la regresión *ZIP* utiliza la liga *logit* que es simétrica para modelar la probabilidad p_i y la liga logaritmo para modelar la media de los conteos λ_i.

Sin embargo no siempre es conveniente utilizar una liga simétrica para ajustar la probabilidad para todos los conjuntos de datos debido a que el sesgo y el error cuadrado medio de los estimadores de máxima verosimilitud pueden ser grandes (Czado y Santner, 1992); por ello se han propuesto funciones liga que son más flexibles.

Aranda-Ordaz (1981) propuso dos funciones liga: una liga simétrica que incluye la liga lineal y la liga *logit* y otra asimétrica a la cual se hará referencia de aquí en adelante como *AO2*, cuyos casos particulares son la liga *logit* cuando su parámetro de forma τ es 1 y la liga *cloglog* cuando $\tau \to 0$; esta liga se ha considerado en modelos de dosis-respuesta como alternativa cuando la función liga no es seleccionada correctamente (Huang, 2002). Bazán *et al.* (2006) utilizaron la liga *normal sesgada* (Azzalini, 1985) a la cual se hará referencia como *NS* de aquí en adelante. La liga *NS* tiene como casos particulares, la liga *probit* cuando el parámetro de forma asociado es $\nu = 0$ y la liga *media normal* cuando el el parámetro de forma $\nu \to \infty$.

En este trabajo se pretende utilizar las funciones liga *NS* y *AO2* en un modelo de regresión ZIP con el fin de modelar la probabilidad p de que $Y = 0$ de una manera más flexible. Asimismo se pretende comparar mediante simulación el ajuste de las estimaciones de $\hat{p}_i$ cuando se simulan probabilidades verdaderas p_i suponiendo ligas *probit*, *logit* y *cloglog*.

El objetivo general del trabajo es generar la metodología para la estimación en el modelo de regresión Poisson inflados con ceros (ZIP) utilizando las ligas *NS* y *AO2*. También se pretende comparar con la bondad de ajuste del modelo con la liga propuesta contra algunas de las existentes, por ejemplo liga *logit* o *probit* y realizar estudios de simulación que permitan cuantificar los errores cometidos al seleccionar de forma incorrecta una función liga y que permitan verificar la consistencia de los estimadores.

La estructura del trabajo es la siguiente. En el capítulo 2 se revisará la literatura referente al tema a estudiar; la regresión ZIP, la distribución normal sesgada y algunos métodos de estimación. Posteriormente en el capítulo 3 se propondrá la metodología para estimar los parámetros de la regresión Poisson inflada con ceros usando la función liga *NS* y *AO2* mediante el enfoque clásico. En el capítulo 4 se realizarán estudios de simulación con el fin de comprobar las propiedades asintóticas de los estimadores. En el capítulo 5 se mostrarán algunos ejemplos de aplicación y sus resultados. Finalmente se mencionarán algunas conclusiones y recomendaciones. En los anexos se proporcionan los códigos de los programas utilizados en el presente trabajo.

Capítulo 2

Revisión de literatura

En este capítulo se hace una revisión de los aspectos relevantes de la regresión Poisson inflada con ceros, así como otros conceptos y metodologías utilizados para el desarrollo del trabajo.

2.1. Distribuciones de mezclas

Definición 2.1 (Mezclas de densidades finitas) *Sea* $\mathbf{y}$ *una variable aleatoria con función de densidad* $g(\mathbf{y};\boldsymbol{\theta})$ *de dimensión* d *que depende de un vector de parámetros* $\boldsymbol{\theta}$ *de dimensión* m *y sea* $H(\boldsymbol{\theta})$ *una función de distribución acumulada de dimensión* m*. Entonces* $f(\boldsymbol{y}) = \int g(\boldsymbol{y};\boldsymbol{\theta})\ dH(\boldsymbol{\theta})$*es llamada una* ***mezcla de densidades****.* H *es llamada la* ***distribución de mezcla****.*

La definición 2.1 es muy general, por lo que aquí se utilizará el caso en que H es discreta y asigna probabilidades positivas a un número *finito* de puntos; en este caso la integral se reemplaza por una suma finita y se obtiene una *mezcla finita*, es decir:

$$f(\mathbf{y}) = \sum_{i=1}^{n} g(\mathbf{y}, \boldsymbol{\theta}_i) H(\boldsymbol{\theta}_i). \tag{2.1}$$

De (2.1) se pueden identificar tres tipos de parámetros. El primer tipo consiste solamente de n componentes de la mezcla finita, el segundo tipo consiste de los componentes de vectores de parámetros $\boldsymbol{\theta}_i$ y el tercero consiste en una mezcla de proporciones [1] de $H_i(\boldsymbol{\theta}_i)$, usualmente denotada como p_i (Everitt y Hand, 1981).

[1] En el presente trabajo se enfoca a este tipo de mezcla.

En otras palabras si $f_1(y), f_2(y), \cdots, f_k(y)$ son *funciones de densidad* que podrían depender de parámetros y $p_1, p_2, \cdots, p_k$ son una *sucesión de parámetros*, tales que $\sum_{i=0}^{k} p_i$ entonces $\sum_{i=0}^{k} p_i f_i(y)$ es una función de densidad llamada distribución contagiada o mezcla (Mood *et al.*, 1974).

Existen varios métodos para estimar parámetros de una mezcla de distribuciones; sin embargo, el método más utilizado es el algoritmo Esperanza-Maximización (Dempster *et al.*, 1977; Tanner, 1996), conocido comúnmente como EM.

2.2. Algoritmo Esperanza-Maximización

El algoritmo Esperanza-Maximización (EM) fue propuesto por Dempster *et al.* (1977) y considera que las n observaciones $\mathbf{y}$ pueden ser vistas como datos incompletos, por lo que supone la existencia de n variables aleatorias no observable $\mathbf{Z}$ que permite realizar la estimación de manera más sencilla. Esto también es conocido como *aumentación de datos* (Tanner, 1996).

En otras palabras, la distribución a los datos incompletos $f_{\mathbf{Y}}(\mathbf{y}; \boldsymbol{\theta})$ se le aumenta una variable latente $\mathbf{Z}$, y la distribución de los datos completos es $f_{\mathbf{Y},\mathbf{Z}}(\mathbf{y}, \mathbf{z}; \boldsymbol{\theta})$ donde $\boldsymbol{\theta}$ es un vector de parámetros de interés de dimensión $p < n$.

El algoritmo EM es un método iterativo que consiste de dos pasos: i) *paso E*, en el que obtiene la esperanza condicional de la log-verosimilitud de los datos completos $l_{\mathbf{Y},\mathbf{Z}}(\boldsymbol{\theta})$ con respecto a $\mathbf{Z}$ dada $\mathbf{Y}$, es decir,

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(j)}) = \mathbb{E}_{\mathbf{Z}|\mathbf{Y}}(l_{\mathbf{Y},\mathbf{Z}}(\boldsymbol{\theta})) = \int \mathbf{Z} f_{\mathbf{Z}|\mathbf{Y}}(\mathbf{Z}|\mathbf{y}; \boldsymbol{\theta}) d\mathbf{Z}, \tag{2.2}$$

ii) *paso M*, en el que se maximiza esta esperanza con respecto al vector de parámetros desconocidos $\boldsymbol{\theta}^{(j)}$ de la j-ésima iteración. Usualmente para resolver el problema de maximización se recurre al uso de derivadas, es decir

$$\frac{\partial}{\partial \theta_l^{(j)}} Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(j)}) = 0, \; l = 1, 2, \cdots, p, \tag{2.3}$$

donde p es el número de parámetros desconocidos. Al tomar las derivadas se obtiene un sistema de p ecuaciones y p incógnitas las cuales se resuelven de manera simultánea y se encuentran los estimadores de la siguiente iteración $\boldsymbol{\theta}^{(j+1)}$. Tanner (1996) recomienda

repetir el algoritmo hasta que máx $|\boldsymbol{\theta}^{(j+1)} - \boldsymbol{\theta}^{(j)}|$ o $|Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(j+1)}) - Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(j)})|$ sea menor a una cantidad $\epsilon > 0$ pequeña, por ejemplo 0.001. Note que máx $|\boldsymbol{\theta}^{(j+1)} - \boldsymbol{\theta}^{(j)}|$ indica que hay que tomar el máximo de las diferencias en valor absoluto entre el vector de parámetros en la iteración actual y la iteración anterior.

En ocasiones no es posible obtener las derivadas (2.3) de forma analítica, por lo que es necesario recurrir a técnicas de optimización numéricas. Estas técnicas consisten en, dada una función objetivo g que depende de una o más variables, encontrar los valores que hagan máximo (mínimo) el valor de la función g; usualmente es difícil encontrar un valor global que maximice (o minimice) a g, por lo que se buscan valores locales en una vecindad; algunas de estas técnicas no requieren el cálculo de primeras derivadas (Press *et al.*, 1992). La función `optim` de R (R Core Team, 2015) tiene implementadas estas rutinas de optimización.

El algoritmo EM tiene propiedades interesantes:

- El valor de $l_{\mathbf{Y},\mathbf{Z}}(\boldsymbol{\theta})$ *no decrece* en cada iteración.
- Es poco sensible a los valores de inicio.
- No es necesario evaluar la primera y segunda derivada de la log-verosimilitud.

Sin embargo en algunas ocasiones, no es fácil encontrar los valores $\hat{\boldsymbol{\theta}}$ que maximicen a $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(j)})$ (2.2) en el paso E, por lo que se utiliza el algoritmo EM generalizado (GEM) (Watanabe y Yamaguchi, 2003). Este algoritmo es una variante del algoritmo EM, en el cual se reemplaza el paso M, con un paso para encontrar $\boldsymbol{\theta}^{(j+1)}$ que satisface:

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(j+1)}) \geq Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(j)}), \tag{2.4}$$

es decir (2.4) indica que es suficiente encontrar un vector de parámetros $\boldsymbol{\theta}^{(j+1)}$ tal que al evaluarlo en $Q(\boldsymbol{\theta}^{(j+1)})$, sea mayor que $Q(\boldsymbol{\theta}^{(j)})$.

El algoritmo EM no produce de manera automática la matriz de varianzas-covarianzas de los estimadores $\mathbb{V}(\boldsymbol{\theta})$; Oakes (1999) propuso un método para calcularlas de manera aproximada:

$$\widehat{\mathbb{V}(\boldsymbol{\theta})} \approx \left(\frac{\partial^2 l_{\mathbf{Y}}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}^2} \right)^{-1}, \tag{2.5}$$

donde $\frac{\partial^2 l_{\mathbf{Y}}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}^2}$ es la matriz de información de Fisher y $l_{\mathbf{Y}}(\boldsymbol{\theta})$ es la log-verosimilitud de los datos incompletos.

Una vez obtenida la matriz de varianzas-covarianzas, se pueden obtener los errores estándar de los estimadores $ee(\hat{\theta}_l), l = 1, \ldots, p$ aplicando raíz cuadrada positiva a la diagonal de la matriz de varianzas-covarianzas. Si el tamaño de muestra es grande, se pueden calcular intervalos de confianza asintóticamente normales para los estimadores mediante $\left(\hat{\theta}_l - z_{1-\frac{\alpha}{2}} \times ee(\hat{\theta}_l), \hat{\theta}_l + z_{1-\frac{\alpha}{2}} \times ee(\hat{\theta}_l)\right)$, donde $z_{1-\frac{\alpha}{2}}$ es el cuantil de la distibución normal estándar que cumple con $\mathbb{P}(Z > z_{1-\frac{\alpha}{2}}) = \frac{\alpha}{2}$.

En general será difícil calcular las derivadas de manera analítica, por lo que se tendrá que recurrir a derivadas numéricas (Press *et al.*, 1992). En R (R Core Team, 2015) existe una función llamada `optimHess` que, dado un vector de parámetros y una función objetivo, que calcula la matriz Hessiana con métodos numéricos, por lo que al invertir dicha matriz, se obtiene la matriz de varianza-covarianza de los estimadores.

2.3. Verosimilitud Perfil

Cuando la función de verosimilitud $L(\boldsymbol{\theta})$ depende de varios parámetros $\boldsymbol{\theta} = (\boldsymbol{\vartheta}, \boldsymbol{\eta})^t$, puede ocurrir que únicamente estemos interesados en un subconjunto de parámetros $\boldsymbol{\vartheta}$ y se puede considerar al resto de parámetros $\boldsymbol{\eta}$ como *parámetros de ruido*.

Usualmente para resolver este problema, se calcula los estimadores de máxima verosimilitud $\hat{\boldsymbol{\eta}}$ de los parámetros de ruido y estos valores se reemplazan en la función de verosimilitud $L(\boldsymbol{\theta})$; a esta función de verosimilitud se le conoce como *verosimiltud perfil* (Sprott, 2000).

Definición 2.2 (Verosimilitud perfil) *Dada la verosimilitud conjunta $L(\boldsymbol{\vartheta}, \boldsymbol{\eta})$, la verosimilitud perfil de $\boldsymbol{\vartheta}$ es: $L_P(\boldsymbol{\vartheta}) = \max_{\boldsymbol{\eta}} L(\boldsymbol{\vartheta}, \boldsymbol{\eta})$, donde la maximización es realizada para $\boldsymbol{\vartheta}$ fijo.*

Note que de la definición 2.2 se desprende que para un vector de parámetros $\boldsymbol{\vartheta}$ fijo, el estimador de máxima verosimilitud de $\boldsymbol{\eta}$ es generalmente una función de $\boldsymbol{\vartheta}$ y podemos representarlo con la notación $L(\boldsymbol{\vartheta}, \hat{\boldsymbol{\eta}}_{\boldsymbol{\vartheta}})$. La verosimilitud perfil se puede tratar como una verosimilitud común (Pawitan, 2001).

2.4. Modelo de regresión binaria

Supóngase que se desea *modelar la probabilidad de éxito* $0 < p < 1$ y tenemos una variable aleatoria Y que denota dos posibles resultados: éxito $\mathbb{P}(Y = 1) = p$ o fracaso $\mathbb{P}(Y = 0) = 1 - p$, es decir $Y \sim Bin(1, p)$ por lo que $\mathbb{E}(Y) = p$.

Según Agresti (2007), si se tienen n observaciones, el vector de variables $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_n)^t$, tiene distribución binomial con un ensayo Bernoulli y probabilidad $\mathbf{p} = (p_1, p_2, \ldots, p_n)^t$; es decir $\mathbf{Y} \sim Bin(\mathbf{1}, \mathbf{p})$.

En este modelo de regresión, el valor de p_i puede variar dependiendo de los valores de una variable o k variables explicativas $x_1, x_2, \ldots, x_k$, es decir, el vector de probabilidades de éxito de dimensión n depende de las covariables; esto es $p_i = g(x_1, x_2, \ldots, x_k)$.

Usualmente, la relación entre las probabilidades p_i y las k covariables $x_1, x_2, \ldots, x_k$ no es lineal; para ello se utiliza una *función liga* $g(\cdot)$, la cual es diferenciable y monótona (Dobson, 2002). La función liga relaciona de manera lineal a las probabilidades p_i con los valores ajustados o predictor lineal $\beta_0 + \beta_{i,1}x_1 + \beta_{i,2}x_2 + \ldots + \beta_{i,k}x_k,\ i = 1, 2, ..., n$ donde $\beta_0, \beta_1, ..., \beta_k$ son parámetros de interés.

El modelo de regresión binaria se puede representar en forma matricial de la siguiente manera

$$g(\mathbf{p}) = \mathbf{X}_i\boldsymbol{\beta} \leftrightarrow \begin{pmatrix} g(p_1) \\ g(p_2) \\ \vdots \\ g(p_n) \end{pmatrix} = \begin{bmatrix} 1 & x_{1,1} & \cdots & x_{1,k} \\ 1 & x_{2,1} & \cdots & x_{2,k} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n,1} & \cdots & x_{n,k} \end{bmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{pmatrix}, \tag{2.6}$$

donde $\boldsymbol{\beta}$ es el vector de parámetros de interés de dimensión $k \times 1$, $\mathbf{X}$ es la matriz diseño de dimensión $n \times k,\ n > k$ que afectan a la probabilidad $\mathbf{p}$ y $\mathbf{X}\boldsymbol{\beta}$ es el predictor lineal.

2.4.1. Funciones liga

En los modelos de regresión binaria se puede utilizar cualquier función de distribución $F_X(x)$ de una variable aleatoria continua para modelar las probabilidades p_i, con el fin de asegurar que las probabilidades estén en el intervalo $[0, 1]$ (Dobson, 2002).

En la regresión logística se modela la probabilidad p_i con la liga *logit* la cual es la función inversa de la distribución acumulada $F_X^{-1}(x)$ de una variable aleatoria logística estándar

$$logit(p_i) = \log\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_{i,1}x_1 + \beta_{i,2}x_2 + \cdots + \beta_{i,k}x_k, \tag{2.7}$$

al despejar la probabilidad p_i se obtiene el *modelo de regresión logística o logit*, es decir:

$$p_i = \frac{e^{\mathbf{X}_i\boldsymbol{\beta}}}{1 + e^{\mathbf{X}_i\boldsymbol{\beta}}};\ 1 - p_i = \frac{1}{1 + e^{\mathbf{X}_i\boldsymbol{\beta}}}, \tag{2.8}$$

donde $\mathbf{X}_i\boldsymbol{\beta} = \beta_0 + \beta_{i,1}x_1 + \beta_{i,2}x_2 + \cdots + \beta_{i,k}x_k$ es el predictor lineal (2.6), esta notación representa el producto interior del renglón i de la matriz diseño $\mathbf{X}$ con el vector de parámetros $\boldsymbol{\beta}$. Note que la matriz diseño $\mathbf{X}$ es de dimensión $n \times (k+1)$ y $n > (k+1)$.

Por otro lado, el modelo de regresión *probit*, utiliza la *función liga probit* denotada por $\Phi^{-1}(\cdot)$; esta función transforma probabilidades en cuantiles de la distribución normal estándar.

$$probit(p_i) = \Phi^{-1}(p_i) = \mathbf{X}_i\boldsymbol{\beta}, \tag{2.9}$$

despejando las probabilidades p_i se obtiene el *modelo de regresión probit*.

$$p_i = \Phi(\mathbf{X}_i\boldsymbol{\beta}), \tag{2.10}$$

donde $\Phi(\cdot)$ representa la función de distribución acumulada normal estándar.

En ocasiones se puede utilizar una función liga asimétrica o de valores extremos (Dobson, 2002), por ejemplo se puede utilizar la función liga *cloglog* la cual tiene la forma

$$\ln(-\ln(1-p_i)) = \mathbf{X}_i\boldsymbol{\beta}. \tag{2.11}$$

Note que al despejar p_i se obtiene

$$p_i = 1 - e^{-e^{\mathbf{X}_i\boldsymbol{\beta}}}. \tag{2.12}$$

En la Figura 2.1 se muestran las ligas más comunes, nótese que la liga *logit* (2.7) y la liga *probit* (2.9) toman el valor cero cuando $p = 0.5$ y la liga *logit* es muy parecida a la liga *cloglog* (2.11) cuando p es cercana a cero.

Existe una generalización de la función liga *logit* propuesta por Aranda-Ordaz (1981), la cual es una liga asimétrica (véase Figura 2.2), que se denominará *AO2*

$$AO2(p_i, \tau) = \ln\left(\frac{(1-p_i)^{-\tau} - 1}{\tau}\right) = \mathbf{X}_i\boldsymbol{\beta}. \tag{2.13}$$

Note que (2.13) incluye a la liga *logit* cuando $\tau = 1$ y a la liga *cloglog* cuando $\tau \to 0$. Al obtener la función inversa se tiene:

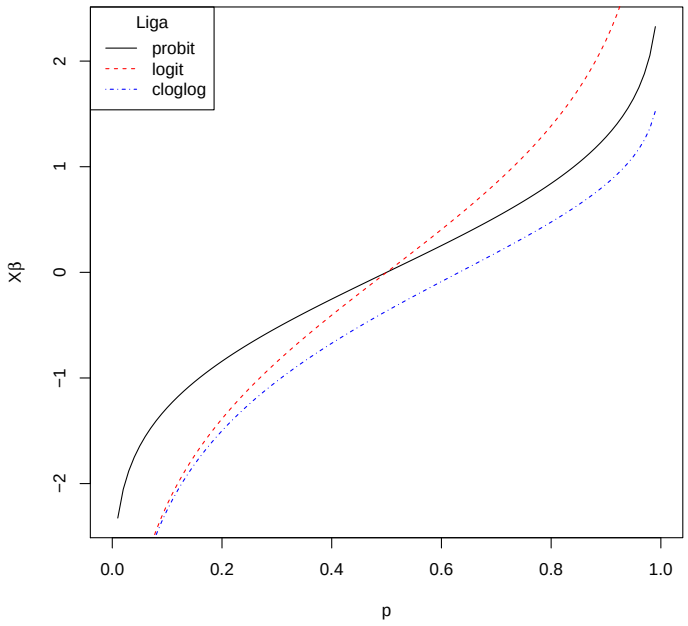

Figura 2.1: Funciones liga más comunes para regresión binaria.

$$p_i = \begin{cases} 1 - (1 + \tau e^{\mathbf{X}_i\boldsymbol{\beta}})^{-\frac{1}{\tau}} & (\tau e^{\mathbf{X}_i\boldsymbol{\beta}}) > -1 \\ 1 & \text{de otro modo.} \end{cases}, \tag{2.14}$$

Note que si $\tau = 1$, entonces las probabilidades se pueden modelar mediante (2.7) y si $\tau \to 0$ entonces se obtiene (2.12).

De acuerdo con Morgan (1992), se puede mostrar que (2.14) es la función de distribución de una variable aleatoria "log-Burr"; Achen (2002) aclara que (2.14) es la distribución de una Burr-exponencial y le llama distribución *scobit* (skew-logit), por lo tanto en este trabajo suponemos que el espacio parametral de τ es positivo.

Aranda-Ordaz (1981) también propuso una función liga simétrica que se denominará *AO1*. Esta liga se reduce a la liga lineal cuando $\tau = 1$ y se obtiene la liga *logit* cuando $\tau \to 0$.

$$AO1(p_i, \tau) = \frac{2}{\tau}\frac{p_i^\tau - (1 - p_i)^\tau}{p_i^\tau - (1 + p_i)^\tau} = \mathbf{X}_i\boldsymbol{\beta} \tag{2.15}$$

Al invertir (2.15) se obtiene:

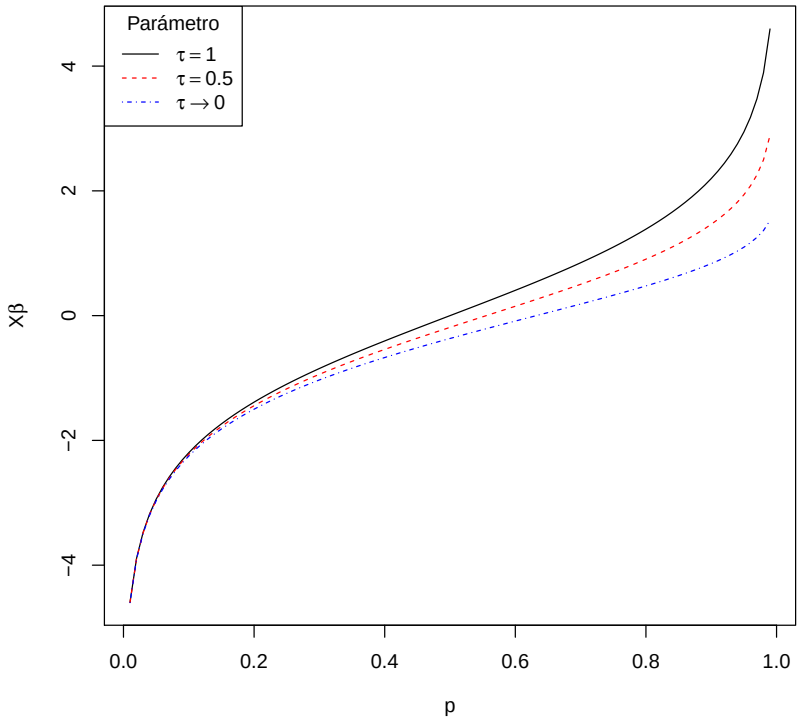

Figura 2.2: Liga Aranda-Ordaz con diferentes parámetros τ.

$$p_i = \begin{cases} 0 & \left(\frac{1}{2}\tau \boldsymbol{X}_i\boldsymbol{\beta} \leq -1\right), \\ \frac{\left(1+\frac{1}{2}\tau \boldsymbol{X}_i\boldsymbol{\beta}\right)^{1/\tau}}{\left(1+\frac{1}{2}\tau \boldsymbol{X}_i\boldsymbol{\beta}\right)^{1/\tau}+\left(1-\frac{1}{2}\tau \boldsymbol{X}_i\boldsymbol{\beta}\right)^{1/\tau}} & \left(\left|\frac{1}{2}\tau \boldsymbol{X}_i\boldsymbol{\beta}\right| < 1\right), \\ 1 & \left(\frac{1}{2}\tau \boldsymbol{X}_i\boldsymbol{\beta} \geq 1\right). \end{cases} \tag{2.16}$$

La liga Aranda-Ordaz asimétrica se ha utilizado en varios modelos de regresión binaria o modelos de dosis-respuesta, por ejemplo Huang (2002) consideró a (2.16) como alternativa cuando la función liga no es seleccionada correctamente en modelos de dosis-respuesta, Morgan (1992) presenta a la función liga Aranda-Ordaz asimétrica (2.15) como una generalización de la liga *logit*, este autor muestra gráficas de la función liga cuando cambia el parámetro de forma.

Tanto la liga *logit* (2.7) como la liga *probit* (2.9) son las funciones liga más conocidas debido que en el caso de la liga *logit* es muy sencillo despejar las probabilidades p_i, como se puede apreciar en (2.8), mientras que en el caso de la liga *probit* es posible obtener las probabilidades (2.10) mediante algún software como R (R Core Team, 2015).

2.5. Modelo de regresión Poisson

Supóngase que se desea modelar el número esperado de eventos $\lambda > 0$ que ocurren en determinado tiempo y tenemos una variable aleatoria Y que tiene distribución Poisson con parámetro $\lambda > 0$, por lo que $\mathbb{E}(Y) = \mathbb{V}(Y) = \lambda$ (Agresti, 2007).

Si se tienen n observaciones, los conteos $Y_i,\ i = 1, 2, \ldots, n$, tiene distribución Poisson parámetro $\lambda_i,\ i = 1, 2, \ldots, n$; en notación matricial, se expresa como $\mathbf{Y} \sim Poi(\boldsymbol{\lambda})$, donde $\mathbf{Y} = (Y_1, Y_2, ..., Y_n)^t$ y $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, ..., \lambda_n)^t$.

Al igual que en la regresión binaria, el valor de λ_i pueden depender de ciertas variables explicatorias $x_1, x_2, \ldots, x_k$. En el modelo de regresión Poisson se puede utilizar la función liga *logaritmo natural*, por lo que tiene la forma:

$$\ln(\lambda_i) = \mathbf{X}_i\boldsymbol{\beta}, \tag{2.17}$$

donde $\mathbf{X}_i\boldsymbol{\beta}$ es el predictor lineal que resulta del producto interior entre el i-ésimo renglón de la matriz diseño y el vector de parámetros de interés. La matriz diseño $\mathbf{X}$ de dimensión $n \times (k+1),\ n > (k+1)$ contiene las covariables que afectan a la media de los conteos λ_i.

El modelo de regresión Poisson es muy restrictivo y en la práctica es usual encontrar datos que exhiben sobredispersión, es decir, en los que la varianza de los conteos es mayor a su media. En el caso de conteos inflado por ceros es usual que se presente el problema de la sobredispersión, por lo que no es recomendable utilizar la regresión Poisson, ya que se subestima las probabilidades asociadas a las ocurrencias de ceros (Cameron y Trivedi, 2005). Existen muchas aplicaciones en los que se tienen datos de conteos que exhiben exceso de ceros, por ejemplo en aplicaciones industriales, salud pública, mejoramiento genético, economía, etc. Naya *et al.* (2008) estudiaron el número de manchas negras que tiene un borrego, Mouatassim y Ezzahid (2012) modelaron el número de reclamos en un esquema de seguros de salud, entre otros. Para ajustar este tipo de datos, se puede utilizar la regresión Poisson inflada por ceros (*Zero-Inflated Poisson Regression*) propuesta por Lambert (1992), quien estudió el número de artículos defectuosos en la manufactura (véase la Figura 2.3); cuando una máquina está desalineada, se producen artículos defectuosos con distribución Poisson (conteos poisson) pero si la máquina funciona perfectamente entonces se producen productos perfectos (ceros estructurales). Esta regresión es muy utilizada en muchas aplicaciones, sin embargo no es muy popular en disciplinas como la econometría (Cameron y Trivedi, 2005).

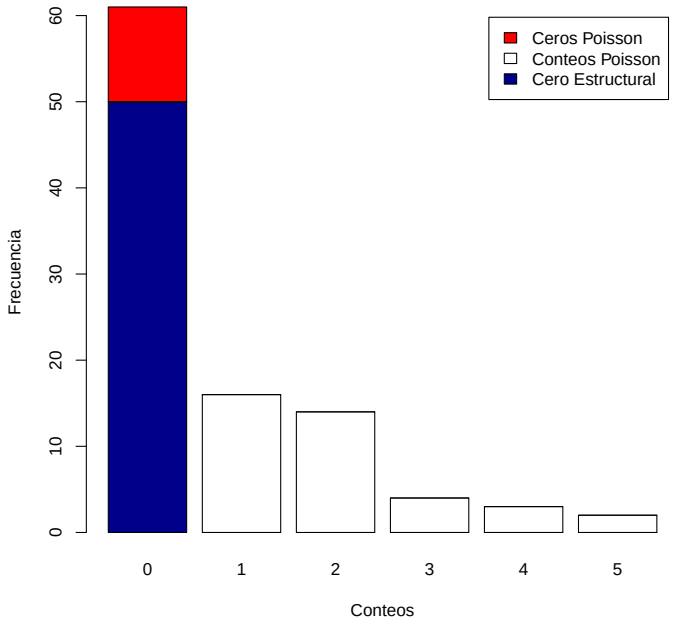

Figura 2.3: Ilustración del número de artículos defectuosos. Los artículos perfectos ocurren cuando la máquina está alineada (barra azul), si la máquina está desalineada se producen productos defectuosos (barras blancas), aunque podrían ocurrir productos perfectos con máquinas desalinaedas (barra roja).

2.6. Regresión ZIP

En el modelo de regresión *ZIP*, se supone que la respuesta Y tiene distribución *ZIP* con parámetros λ y p, esto es:

$$f_Y(y;p,\lambda) = \begin{cases} p + (1-p)\, e^{-\lambda} & \text{si } y = 0;\ \ 0 < p < 1; \lambda > 0 \\ (1-p)\, \frac{\lambda^y}{y!} e^{-\lambda} & \text{si } y = 1, 2, \ldots;\ \ 0 < p < 1; \lambda > 0 \\ 0 & \text{de otro modo.} \end{cases} \tag{2.18}$$

La función generadora de momentos de Y es $m_Y(t) = p + (1-p)e^{\lambda(e^t-1)}\ \forall t$, por lo que su media es $\lambda(1-p)$ y su varianza $\lambda(1-p)(1+p\lambda)$. En este caso el cociente $\frac{\lambda(1-p)}{\lambda(1-p)(1+p\lambda)} = \frac{1}{1+p\lambda} < 1\ \ \forall\ \ p, \lambda$, lo cual indica que la varianza es siempre mayor que la media, por lo tanto el modelo ZIP es capaz de modelar de forma natural datos de conteos con problemas de sobredispersión. Note que si $p = 0$, se obtiene la distribución Poisson con parámetro λ; si $p = 1$ se tiene una distribución degenerada en cero.

La distribución ZIP es una mezcla de Poisson y Bernoulli, y constituye un caso particular de la distribución de series de potencias inflados con cero (*Zero-Inflated Power Series*) donde $Y = (1 - V)U$ donde V es una variable aleatoria Bernoulli y U es una variable aleatoria que tiene distribución serie de potencias como la Poisson o binomial negativa (Ghosh *et al.*, 2006).

Cuando se tienen n observaciones, entonces $Y_i \sim ZIP(\lambda_i, p_i),\ i = 1, 2, \ldots, n$. En la regresión ZIP es de interés modelar la media de los conteos λ_i y la probabilidad de los ceros p_i como función de covariables no relacionadas, mediante las funciones liga logaritmo (2.17) y *logit* (2.7)

$$\begin{aligned} \ln(\lambda_i) &= \beta_0 + \beta_1 b_{i,1} + \cdots + \beta_{k_1} b_{i,k_1} \\ \ln\left(\frac{p_i}{1-p_i}\right) = \operatorname{logit}(p_i) &= \gamma_0 + \gamma_1 g_{i,1} + \cdots + \gamma_{k_2} g_{i,k_2}, \end{aligned} \tag{2.19}$$

debido a que la varianza es mayor que la media, la regresión ZIP también es útil para modelar sobre dispersión.

Al despejar λ_i y p_i de (2.19) se obtiene:

$$\begin{aligned} \lambda_i &= e^{\mathbf{B}_i \boldsymbol{\beta}}, \\ p_i &= \frac{e^{\mathbf{G}_i \boldsymbol{\gamma}}}{1 + e^{\mathbf{G}_i \boldsymbol{\gamma}}}, \end{aligned} \tag{2.20}$$

donde $\mathbf{B}_i = (1, b_{i,1}, \cdots, b_{i,k_1})$ y $\mathbf{G}_i = (1, g_{i,1}, \cdots, g_{i,k_2})$ son los renglones de las matrices diseño de dimensiones $n \times (k_1+1)$ y $n \times (k_2+1)$ que afectan a la media del estado imperfecto λ_i y a la probabilidad del estado perfecto p_i respectivamente; $\boldsymbol{\beta} = (\beta_0, \beta_1, ..., \beta_{k_1})^t$ y $\boldsymbol{\gamma} = (\gamma_0, \gamma_1, ...\gamma_{k_2})^t$ son los vectores de parámetros de interés de dimensión $(k_1 + 1) \times 1$ y $(k_2 + 1) \times 1$ respectivamente; además suponemos que λ_i y p_i no están relacionados $\forall\, i$.

Utilizando la parametrización (2.20), la log-verosimilitud es:

$$\begin{aligned} l_{\mathbf{Y}}(\boldsymbol{\beta}, \boldsymbol{\gamma}) \;=\; & \sum_{y_i \equiv 0} \ln\left(e^{\mathbf{G}_i \boldsymbol{\gamma}} + e^{-e^{\mathbf{B}_i \boldsymbol{\beta}}}\right) + \sum_{y_i \neq 0} \left(y_i \mathbf{B}_i \boldsymbol{\beta} - e^{\mathbf{B} \boldsymbol{\beta}}\right) \\ & - \sum_{i=1}^{n} \ln\left(1 + e^{\mathbf{G}_i \boldsymbol{\gamma}}\right) - \sum_{y_i \neq 0} \ln\left(y_i!\right), \end{aligned} \tag{2.21}$$

donde la notación $\mathbf{B}_i\boldsymbol{\beta}$ y $\mathbf{G}_i\boldsymbol{\gamma}$ con $i = 1, 2, \ldots, n$ representan el producto punto del i-ésimo renglón de la matriz diseño correspondiente por su respectivo vector de parámetros. Note que en (2.21) se dividen los datos en dos grupos, el primer grupo lo conforman aquellos datos cuya variable respuesta es cero y el otro en que la variable respuesta es mayor a cero.

Se estiman los parámetros de interés $\boldsymbol{\beta}$ y $\boldsymbol{\gamma}$ mediante el algoritmo EM, definiendo una variable latente $Z_i = 1$ si Y_i proviene del cero estructural con probabilidad p_i y $Z_i = 0$ cualquier otro caso con probabilidad $1-p_i$, por lo que la función de densidad de los datos completos está dada por:

$$f_{Y_i,Z_i}\left(z_i, y_i; p_i, \lambda_i\right) = \left[p_i\right]^{z_i}\left[(1-p_i)e^{-\lambda_i}\lambda_i^{y_i}\right]^{1-z_i}\left[\frac{1}{y_i!}\right]^{1-z_i} I_{\{0,1\}}(z_i)I_{\{0,1,2,\cdots\}}(y_i), \tag{2.22}$$

la log-verosimilitud de los datos completos se puede expresar como:

$$l_{\mathbf{Y},\mathbf{Z}}(\boldsymbol{\lambda}, \mathbf{p}) = \sum_{i=1}^{n}\left(z_i\mathbf{G}_i\boldsymbol{\gamma} - \ln\left(1+e^{\mathbf{G}_i\boldsymbol{\gamma}}\right)\right) + \sum_{i=1}^{n}(1-z_i)\left(y_i\mathbf{B}_i\boldsymbol{\beta} - e^{\mathbf{B}_i\boldsymbol{\beta}}\right) - \sum_{i=1}^{n}(1-z_i)\ln\left(y_i!\right). \tag{2.23}$$

De (2.23) se observa que $\boldsymbol{\gamma}$ y $\boldsymbol{\beta}$ se pueden maximizar por separado, por lo que esta verosimilitud es más fácil de maximizar que (2.21).

En el paso E (2.2), se calcula la esperanza condicional de Z_i dado Y_i, obteniendo

$$\mathbb{E}(Z_i|Y_i) = z_i^{(k+1)} = \frac{p_i^{(k)}}{p_i^{(k)} + (1-p_i^{(k)})e^{-\lambda_i^{(k)}}} I_{\{y_i=0\}} = \frac{e^{\mathbf{G}\boldsymbol{\gamma}^{(k)}}}{e^{\mathbf{G}\boldsymbol{\gamma}^{(k)}} + e^{-e^{\mathbf{B}\boldsymbol{\beta}^{(k)}}}} I_{\{y_i=0\}}, \tag{2.24}$$

por lo que la función a maximizar es:

$$\begin{aligned} Q\left(\boldsymbol{\theta}^{(k+1)}|\boldsymbol{\theta}^{(k)}\right) &= \sum_{i=1}^{n}\ln\left(z_i^{(k+1)}\mathbf{G}_i\boldsymbol{\gamma} - \log\left(1+e^{\mathbf{G}_i\boldsymbol{\gamma}}\right)\right) \\ &+ \sum_{i=1}^{n}(1-z_i^{(k+1)})\left(y_i\mathbf{B}_i\boldsymbol{\beta} - e^{\mathbf{B}_i\boldsymbol{\beta}}\right) - \sum_{i=1}^{n}(1-z_i^{(k+1)})\ln\left(y_i!\right). \end{aligned} \tag{2.25}$$

La maximización de (2.25) puede realizarse numéricamente, por ejemplo con el algoritmo de Nelder y Mead (1965); diversos softwares estadísticos pueden ajustar un modelo ZIP, por ejemplo en `R` se puede utilizar la función `zeroinfl` que está en la librería `pscl` (Zeileis *et al.*, 2008).

Ghosh *et al.* (2006) analizan el modelo de regresión ZIP desde un punto de vista Bayesiano. Cabe señalar que dichos autores ejemplificaron los datos de manufacturas de Lambert (1992).

2.7. Análisis Bayesiano

En la inferencia Bayesiana, se consideran a los parámetros de interés $\boldsymbol{\theta} = (\theta_1, \theta_2, \cdots, \theta_k)^t$ como cantidades desconocidas, por lo que se puede expresar esa incertidumbre mediante una función de distribución $f(\boldsymbol{\theta})$ conocida como **distribución a priori**.

También se considera que los datos tienen una distribución condicionada a los valores particulares que tome $\boldsymbol{\theta}$, es decir $f(\mathbf{x}|\boldsymbol{\theta})$; a esta distribución se le conoce como **distribución de los datos o verosimilitud**.

Nuestro interés es combinar la información previa del parámetro con la información de los datos, mediante el **teorema de Bayes**, para obtener nuevo conocimiento a partir de los datos, es decir

$$f(\boldsymbol{\theta}|\mathbf{x}) = \frac{f(\mathbf{x}|\boldsymbol{\theta})f(\boldsymbol{\theta})}{\int f(\mathbf{x}|\boldsymbol{\theta})f(\boldsymbol{\theta})d\boldsymbol{\theta}}, \tag{2.26}$$

a esta distribución le llamaremos **distribución a posteriori**.

Debido a que $f_{\mathbf{X}}(\mathbf{x}) = \int f(\mathbf{x}|\boldsymbol{\theta})f(\boldsymbol{\theta})d\boldsymbol{\theta}$ no depende de $\boldsymbol{\theta}$, se considera como constante y (2.26) se expresa:

$$f(\boldsymbol{\theta}|\mathbf{x}) \propto f(\mathbf{x}|\boldsymbol{\theta})f(\boldsymbol{\theta}) \tag{2.27}$$

Sin embargo, en muchas ocasiones es muy complicado calcular la distribución *a posteriori* (2.26) ó (2.27), por ello se utilizan algunos métodos de simulación como *Cadenas de Markov Monte Carlo* (Markov Chain Monte Carlo o MCMC).

Las MCMC son métodos generales que muestrean valores de $\boldsymbol{\theta}$ de distribuciones conocidas, estas muestras se van corrigiendo para aproximarse mejor a $f(\boldsymbol{\theta}|\mathbf{x})$.

Un aspecto importante es que *las muestras son obtenidas secuencialmente y únicamente dependen del valor inmediato anterior.*

En general, se inicia en el valor $\boldsymbol{\theta}^{(0)}$ y se van muestreando valores $\boldsymbol{\theta}^{(j)}$, $j = 1, 2, \cdots$, de cierta *distribución de transición* $f(\boldsymbol{\theta}^{(j)}|\boldsymbol{\theta}^{(j-1)})$ que depende del valor muestreado anterior $\boldsymbol{\theta}^{(j-1)}$; si la distribución de transición está bien construida, los valores de la cadena $\boldsymbol{\theta}^{(t)}$ deberán converger a una única distribución estacionaria que es la distribución **a posteriori** $f(\boldsymbol{\theta}|\mathbf{x})$ (Gelman *et al.*, 2004).

A continuación se describen de manera general algunos algoritmos que se utilizan para encontrar las distribuciones *a posteriori*; se pueden consultar mayores detalles en Tanner (1996) o Robert y Casella (2004).

2.7.1. Muestreador de Gibbs

El muestreador de Gibbs (Geman y Geman, 1984) genera secuencias dependientes de parámetros $\boldsymbol{\theta}^{(j)}$, $j = 1, 2, \cdots$, mediante las *distribuciones condicionales completas*

$$\begin{gathered} f(\theta_1|\theta_2, \theta_3, \cdots, \theta_k, \mathbf{x}) \\ f(\theta_2|\theta_1, \theta_3, \cdots, \theta_k, \mathbf{x}) \\ \vdots \\ f(\theta_k|\theta_1, \theta_2, \cdots, \theta_{k-1}, \mathbf{x}) \end{gathered}$$

Para obtener para implementar el muestreador de Gibbs, se sigue el siguiente algoritmo:

1. Dado un vector de valores iniciales $\boldsymbol{\theta}^{(j)} = \left(\theta_1^{(j)}, \theta_2^{(j)}, \cdots, \theta_k^{(j)}\right)^t$, simular $\theta_1^{(j+1)}$ de $f(\theta_1|\theta_2^{(j)}, \cdots, \theta_k^{(j)}, \mathbf{x})$.
2. Simular $\theta_2^{(j+1)}$ de $f(\theta_2|\theta_1^{(j+1)}, \theta_3^{(j)} \cdots, \theta_k^{(j)}, \mathbf{x})$.
3. Seguir simulando consecutivamente hasta $\theta_k^{(j+1)}$ de $f(\theta_k|\theta_1^{(j+1)}, \theta_2^{(j+1)} \cdots, \theta_{k-1}^{(j+1)}, \mathbf{x})$.
4. Repetir m veces el algoritmo desde el paso 1.

Este algoritmo generará una sucesión de valores dependientes

$$\boldsymbol{\theta}_1 = \begin{pmatrix} \theta_1^{(1)} \\ \theta_1^{(2)} \\ \vdots \\ \theta_1^{(m)} \end{pmatrix}, \ \boldsymbol{\theta}_2 = \begin{pmatrix} \theta_2^{(1)} \\ \theta_2^{(2)} \\ \vdots \\ \theta_2^{(m)} \end{pmatrix}, \ \cdots, \ \boldsymbol{\theta}_m = \begin{pmatrix} \theta_k^{(1)} \\ \theta_k^{(2)} \\ \vdots \\ \theta_k^{(m)} \end{pmatrix}$$

La sucesión de valores $\boldsymbol{\theta}^{(j)}$ es incondicionalmente independiente de $\boldsymbol{\theta}^{(0)}, \boldsymbol{\theta}^{(1)}, \cdots, \boldsymbol{\theta}^{(j-2)}$ dado $\boldsymbol{\theta}^{(j-1)}$, es decir $\boldsymbol{\theta}^{(j)}$ dependerá únicamente de $\boldsymbol{\theta}^{(j-1)}$ (Hoff, 2009).

2.7.2. Algoritmo de Metropolis y Metropolis-Hastings

Este algoritmo fue propuesto originalmente por Metropolis *et al.* (1953) y posteriormente generalizado por Hastings (1970).

El algoritmo de Metropolis es una adaptación de una caminata aleatoria que usa una regla de aceptación y rechazo para converger a una distribución objetivo $f(\boldsymbol{\theta}|\mathbf{x})$ a partir de una distribución inicial $f^*(\boldsymbol{\theta}|\mathbf{x})$.

1. Obtener un valor $\boldsymbol{\theta}^{(j)}$ $j = 1, 2, \cdots, m$ de la distribución inicial $f^*(\boldsymbol{\theta}|\mathbf{x})$.
2. Proponer una *distribución de transición* $t(\boldsymbol{\theta}^{(*)}|\boldsymbol{\theta}^{(j)})$. La distribución propuesta debe satisfacer la condición de simetría de sus distribuciones condicionales, es decir $t(\boldsymbol{\theta}_1|\boldsymbol{\theta}_2) = t(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\ \forall j = 1, 2, \cdots$
3. Calcular la razón de densidades de Metropolis $r = \frac{f^*(\boldsymbol{\theta}^{(*)}|\mathbf{x})}{f^*(\boldsymbol{\theta}^{(j)}|\mathbf{x})}$
4. Al tiempo $j+1$ Muestrear un valor $\boldsymbol{\theta}^{(*)}$ de $t(\boldsymbol{\theta}^{(*)}|\boldsymbol{\theta}^{(j)})$ y utilizar la siguiente regla:
   - Aceptar $\boldsymbol{\theta}^{(*)}$ con probabilidad $\alpha = \text{mín}\,(1, r)$.
   - Rechazar $\boldsymbol{\theta}^{(*)}$ y hacer $\boldsymbol{\theta}^{(j+1)} = \boldsymbol{\theta}^{(j)}$ con probabilidad $1 - \alpha$.
5. Repetir m veces el algoritmo desde el paso 1.

El algoritmo Metropolis-Hastings es más flexible ya que no requiere se cumpla la condición $t(\boldsymbol{\theta}_1|\boldsymbol{\theta}_2) = t(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1)\ \forall j = 1, 2, \cdots$. En este algoritmo de Metroplolis se utiliza la razón $r = \frac{f^*(\boldsymbol{\theta}^{(*)}|\mathbf{x})t(\boldsymbol{\theta}^{(j)})|\boldsymbol{\theta}^{(*)})}{f^*(\boldsymbol{\theta}^{(j)}|\mathbf{x})t(\boldsymbol{\theta}^{(*)}|\boldsymbol{\theta}^{(j)})}$ y el resto de los pasos del algoritmo se realiza de manera similar. La convergencia a la distribución objetivo $f(\boldsymbol{\theta}|\mathbf{x})$ resulta ser en la misma forma que el algoritmo de Metropolis (Gelman *et al.*, 2004).

2.8. Distribución normal sesgada

Definición 2.3 *Sea W una variable aleatoria continua con función de densidad $f_W(w;\nu) = 2\phi(w)\,\Phi(\nu w)\,I_{(-\infty,\infty)}(w)$ donde $\phi(\cdot)$ denota la densidad normal estándar y $\Phi(\cdot)$ denota la función de distribución de la normal estándar y $\nu \in \mathbb{R}$. Entonces se dice que W tiene distribución normal sesgada con parámetro de forma ν, se denota como $NS(\nu)$.*

De la definición 2.3 se puede deducir que la distribución normal sesgada tiene como caso particular a la distribución normal estándar cuando $\nu = 0$ y a la media normal cuando $\nu \to \infty$ (véase Figura 2.4).

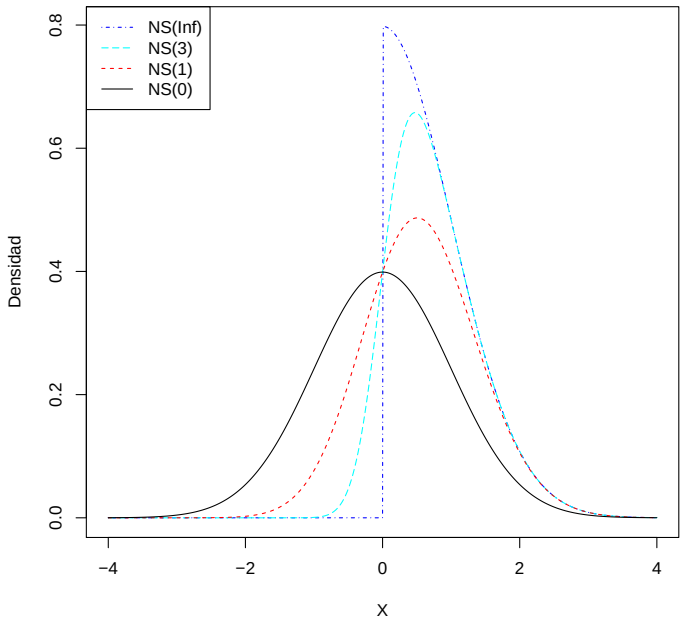

Figura 2.4: Densidad de la Normal Sesgada con distintos parámetros de forma ν. Note que cuando $\nu = 0$ se obtiene la densidad normal estándar y si $\nu \to \infty$ se obtiene la distribución media normal.

La función generadora de momentos de una variable aleatoria Normal Sesgada W es: $M_W(t) = 2e^{\frac{t^2}{2}}\Phi(\delta t)\ \forall t$, donde $\delta = \frac{\nu}{\sqrt{1+\nu^2}}$ y $-1 \leq \delta \leq 1$, la media de W es $\mathbb{E}(W) = \sqrt{\frac{2}{\pi}}\frac{\nu}{\sqrt{1+\nu^2}}$, y su varianza es $\mathbb{V}(W) = 1 - \left(\sqrt{\frac{2}{\pi}}\frac{\nu}{\sqrt{1+\nu^2}}\right)^2$.

De acuerdo con Bazán *et al.* (2006), la función de distribución de W se puede calcular con

$$F_W(w;\nu) = \mathbb{P}(W \leq w) = 2\Phi_2((w,0);-\delta), \tag{2.28}$$

donde $\Phi_2((w,0);-\delta)$ corresponde a la distribución normal estándar bivariada con coeficiente de correlación $-\delta$, evaluada en el punto $(w,0)$.

Una propiedad de la normal sesgada es la siguiente:

$$\text{Si } W \sim NS(\nu) \text{ entonces } -W \sim NS(-\nu). \tag{2.29}$$

Se pueden encontrar más detalles en Azzalini (1985).

La librería de funciones `sn` desarrollado por Azzalini (2014) permite calcular densidades, probabilidades, cuantiles y números aleatorios de la distribución Normal Sesgada. Mediante la función `qsn` del paquete `sn` de `R`, es posible graficar las funciones liga de una normal sesgada para diferentes parámetros ν y compararla con la liga probit.

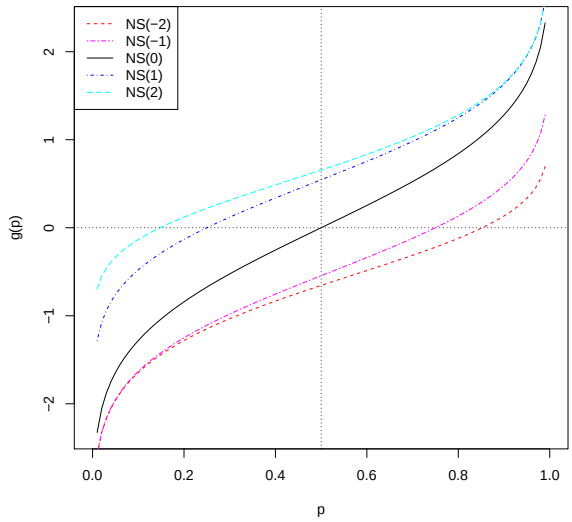

Figura 2.5: Funciones liga de la normal sesgada y probit como caso particular.

Como se observa en la Figura 2.5 al cambiar los valores del parámetro ν, se obtienen distintas curvas con forma asimétrica. Por ejemplo si $\nu > 0$, entonces la liga *NS* se encuentra por arriba de la liga probit y se aproxima a ésta cuando $p \to 1$; en caso contrario la liga *NS* se encontrará por debajo la liga probit y será similar a ésta cuando $p \to 0$.

En la Tabla 2.1 se muestran algunos valores de ν y $\delta = \frac{\nu}{\sqrt{1+\nu^2}}$. Nótese que cuando ν se acerca a ∞, δ se aproxima forma asintótica a 1; además se verifica que cuando que $\delta \approx \nu$ alrededor del cero.

2.8.1. Problemas de inferencia de la distribución normal sesgada

Diversos autores, por ejemplo Sartori (2006) y Pal *et al.* (2012), han descrito problemas de estimación del parámetro de forma ν en la distribución normal sesgada, aún así ésta se ha utilizado para construir funciones liga para datos binarios (Bazán *et al.*, 2006).

ν	δ	ν	δ
-0.5	-0.4472	1	0.7071
-0.3	-0.2873	2.5	0.9285
-0.1	-0.0995	5	0.9806
-0.05	-0.0499	9	0.9939
0	0	100	0.99995

Tabla 2.1: Comparación de los valores del parámetro ν y su transformación δ en la densidad normal sesgada.

Considérese una muestra aleatoria $\mathbf{W} = (W_1, W_2, \cdots, W_n)^t$ de tamaño n que proviene de la distribución normal sesgada. La función de log-verosimilitud es

$$\begin{aligned} l_{\mathbf{W}}(\nu) &= n \ln 2 + \sum_{i=1}^{n} \ln\left(\phi\left(w_i\right)\right) + \sum_{i=1}^{n} \ln\left(\Phi\left(\nu w_i\right)\right) \\ &\propto \sum_{i=1}^{n} \ln\left(\Phi\left(\nu w_i\right)\right) \end{aligned} \tag{2.30}$$

En la Figura 2.6 se muestra la función de log-verosimilitud para los datos de frontera[2], que provienen de una simulación de 50 datos de la distribución normal sesgada con $\nu = 5$, note que la función de log-verosimilitud es casi plana alrededor de 5, y a partir de 8 comienza a descender lentamente.

El problema se complica aún más si $\nu > 0$ y todas las observaciones son positivas, entonces (2.30) es monótona creciente, por lo que el estimador de máxima verosimilitud no existe (Sartori, 2006).

[2]Los datos se encuentran disponibles en http://azzalini.stat.unipd.it/SN/frontier.dat.

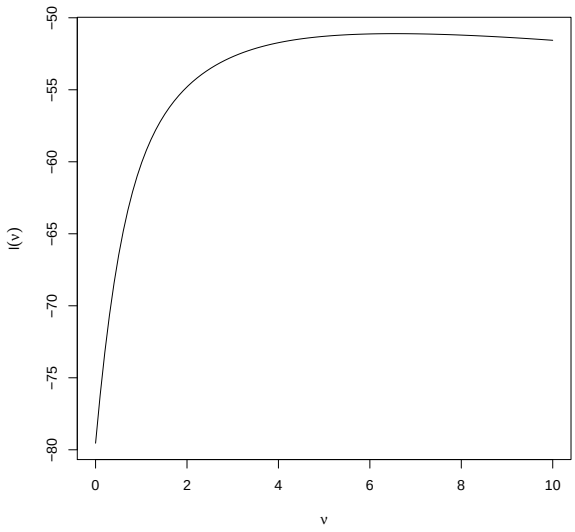

Figura 2.6: Ilustración del problema de estimación en la distribución normal sesgada.

Capítulo 3

Regresión Poisson inflada con ceros usando ligas no convencionales

3.1. Planteamiento del problema

Considere el modelo de regresión *ZIP* (2.18) y suponga que se desea estimar el vector de probabilidades $\boldsymbol{p}$ de que se presente un cero estructural mediante dos funciones ligas generalizadas: la liga Normal Sesgada *NS* con parámetro ν o la liga Aranda-Ordaz asimétrica *AO2* con parámetro τ.

$$F_W^{-1}(p_i, \nu) = \mathbf{G}_i\boldsymbol{\gamma}, \tag{3.1}$$
$$AO2(p_i, \tau) = \mathbf{G}_i\boldsymbol{\gamma}, \tag{3.2}$$

donde $F_W^{-1}(p_i, \nu)$ es la función cuantil de la liga *NS* con parámetro $\nu \in \mathbb{R}$ y $AO2(p_i, \tau)$ es la liga Aranda-Ordaz asimétrica con parámetro $\tau > 0$ (2.13); $\mathbf{G}_i$ es el i-ésimo renglón de la matriz diseño que afecta a la probabilidad de que la respuesta provenga de un cero estructural y $\boldsymbol{\gamma}$ es el vector de parámetros de interés.

Se pretende modelar con estas ligas las probabilidades p_i, modificando sus parámetros de forma ν y τ respectivamente. Para realizar el ajuste de los modelos, se considerará la estimación con algoritmo EM.

Debido a que se tiene interés en modelar ligas asimétricas, no se tomará en cuenta la liga Aranda-Ordaz simétrica (2.15). En las siguientes secciones se describe el proceso de estimación.

3.2. Estimación con parámetro de forma conocido

Utilizaremos el algoritmo EM (Dempster *et al.*, 1977) para estimar $\boldsymbol{\beta}$ y $\boldsymbol{\gamma}$, donde

$$s = \begin{cases} \nu & \text{si es liga } NS \\ \tau & \text{si es liga } AO2 \end{cases},$$

se supondrá conocido.

Para llevar a cabo la estimación, se procede como en (2.22), con la diferencia de que se utilizará las ligas mostradas en (3.1) y (3.2), por lo que las probabilidades se pueden calcular mediante

$$p_i = p\left(\mathbf{G}_i\boldsymbol{\gamma}, s\right) = \begin{cases} F_W\left(\mathbf{G}_i\boldsymbol{\gamma}, \nu\right) & \text{si es liga } NS \\ \mathbf{1} - \left(\mathbf{1} - \tau e^{\mathbf{G}_i\boldsymbol{\gamma}}\right)^{-\frac{1}{\tau}} & \text{si es liga AO2} \end{cases}, \tag{3.3}$$

donde F_W es la función de distribución normal sesgada, definida en (2.28) con parámetro de sesgo ν.

Debido a que el algoritmo EM es un método iterativo, se requieren ciertos valores iniciales:

$$\boldsymbol{\beta}^{(j)} = \begin{pmatrix} \beta_0^{(j)} \\ \beta_1^{(j)} \\ \vdots \\ \beta_{k_1}^{(j)} \end{pmatrix} \quad \boldsymbol{\gamma}^{(j)} = \begin{pmatrix} \gamma_0^{(j)} \\ \gamma_1^{(j)} \\ \vdots \\ \gamma_{k_2}^{(j)} \end{pmatrix}.$$

En el paso E (2.2) se utiliza la esperanza condicional del modelo ZIP de forma iterativa (2.24) con la diferencia de que las probabilidades se ajustan con (3.3)

$$\mathbb{E}(Z_i|Y_i) = z_i^{(j+1)} = \frac{p_i^{(j)}}{p_i^{(j)} + (1 - p_i^{(j)})e^{-\lambda_i^{(j)}}} I_{\{y_i=0\}}, \tag{3.4}$$

donde:

$$p_i^{(j)} = p\left(\mathbf{G}\boldsymbol{\gamma}^{(j)}, s\right), \tag{3.5}$$

$$\lambda_i^{(j)} = e^{\mathbf{B}\boldsymbol{\gamma}_i^{(j)}}. \tag{3.6}$$

Sustituyendo (3.4), (3.5) y (3.6) en (2.22) se obtiene la función Q:

$$\begin{aligned} Q\left(\theta^{(j+1)}|\theta^{(j)}\right) &= \sum_{i=1}^{n} z_i^{(j+1)} \ln\left(p\left(\mathbf{G}_i\boldsymbol{\gamma}^{(j)}, s\right)\right) \\ &+ \sum_{i=1}^{n}\left(1 - z_i^{(j+1)}\right) \ln\left(\left(1 - p\left(\mathbf{G}_i\boldsymbol{\gamma}^{(j)}, s\right)\right) e^{y_i\mathbf{B}_i\boldsymbol{\beta}^{(j)} - e^{\mathbf{B}_i\boldsymbol{\beta}^{(j)}}}\right) \\ &- \sum_{i=1}^{n}\left(1 - z_i^{(j+1)}\right) \ln\left(y_i!\right) \end{aligned}$$

Note que el tercer término no contiene ningún parámetro de interés, por lo que es equivalente a calcular:

$$\begin{aligned} Q\left(\boldsymbol{\theta}^{(j+1)}|\boldsymbol{\theta}^{(j)}\right) &\propto \sum_{i=1}^{n} z_i^{(j+1)} \ln\left(p\left(\mathbf{G}_i\boldsymbol{\gamma}^{(j)}, s\right)\right) \\ &+ \sum_{i=1}^{n}\left(1 - z_i^{(j+1)}\right) \ln\left(\left(1 - p\left(\mathbf{G}_i\boldsymbol{\gamma}^{(j)}, s\right)\right) e^{y_i\mathbf{B}_i\boldsymbol{\beta}^{(j)} - e^{\mathbf{B}_i\boldsymbol{\beta}^{(j)}}}\right) \end{aligned} \tag{3.7}$$

Para calcular el paso M se maximiza (3.7) obteniendo las estimaciones

$$\boldsymbol{\beta}^{(j+1)} = \begin{pmatrix} \beta_0^{(j+1)} \\ \beta_1^{(j+1)} \\ \vdots \\ \beta_{k_1}^{(j+1)} \end{pmatrix} \quad \boldsymbol{\gamma}^{(j+1)} = \begin{pmatrix} \gamma_0^{(j+1)} \\ \gamma_1^{(j+1)} \\ \vdots \\ \gamma_{k_2}^{(j+1)} \end{pmatrix}. \tag{3.8}$$

Posteriormente se vuelve a repetir el procedimiento anterior utilizando los valores obtenidos en (3.8) como semillas iniciales.

En resumen, los pasos a seguir son los siguientes:

1. Elegir valores iniciales $\boldsymbol{\theta}^{(j)} = \left(\boldsymbol{\gamma}^{(j)}, \boldsymbol{\beta}^{(j)}\right)^t$ con $j = 0$ para iniciar el algoritmo, para un valor fijo s.
2. Calcular la esperanza condicional $\mathbb{E}(Z_i|Y_i) = z_i^{(j+1)}$ (3.4) con los valores del paso anterior.
3. Obtener la función a maximizar $Q\left(\boldsymbol{\theta}^{(j+1)}|\boldsymbol{\theta}^{(j)}\right)$ (3.7).

4. Obtener los valores $\boldsymbol{\theta}^{(j+1)} = \left(\boldsymbol{\gamma}^{(j+1)}, \boldsymbol{\beta}^{(j+1)}\right)^t$ que maximizan $Q\left(\boldsymbol{\theta}^{(j+1)}|\boldsymbol{\theta}^{(j)}\right)$.

5. Repetir desde el paso 2 al 5 hasta convergencia.

En el anexo A se presenta una función que permite calcular los parámetros de interés utilizando el algoritmo descrito anteriormente.

3.3. Estimación con parámetro de forma desconocido

En esta sección se utiliza el enfoque de verosimilitud perfil (Sprott, 2000) para encontrar los estimadores de interés $\hat{\boldsymbol{\beta}}$ y $\hat{\boldsymbol{\gamma}}$. Note que el espacio paramétrico Θ de s depende de la función liga seleccionada:

$$\Theta(s) = \begin{cases} \nu \in \mathbb{R} & \text{si es liga } NS \\ \tau > 0 & \text{si es liga } AO2. \end{cases}$$

Para llevar a cabo este procedimiento, se debe hacer una rejilla con valores posibles de s y en cada punto de la rejilla se debe estimar $\hat{\boldsymbol{\beta}}$ y $\hat{\boldsymbol{\gamma}}$ y calcular su verosimilitud. Se elegirá el conjunto de valores que maximice la verosimilitud (véase Figura 3.1).

Se recomienda utilizar este método de estimación, debido a que en la práctica al realizar la maximización de todos los parámetros en forma simultánea con los algoritmos de Nelder y Mead y BFGS se presentan problemas numéricos y es posible también obtener soluciones que no están dentro del espacio parametral, sobre todo, para el parámetro τ.

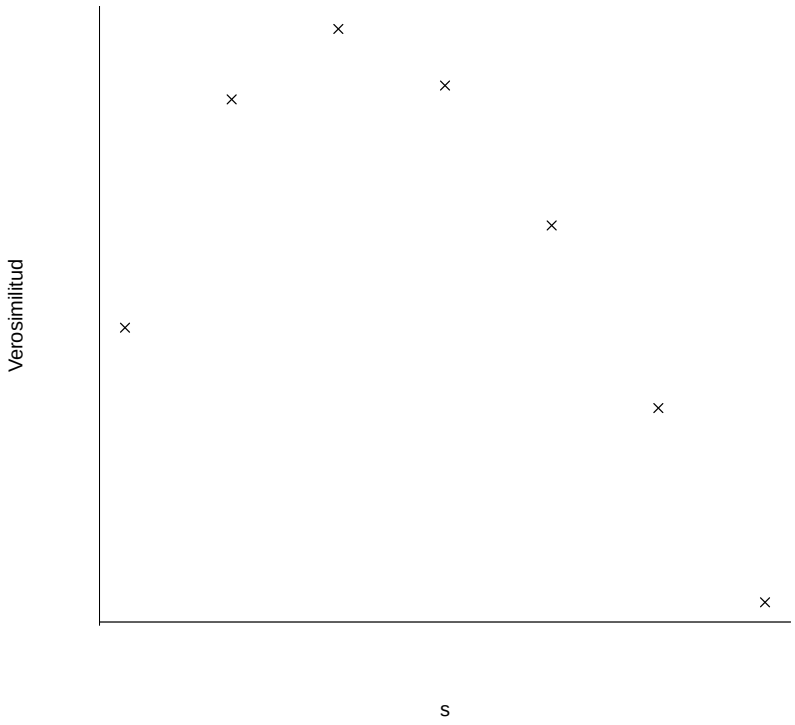

Figura 3.1: Ilustración del método de verosimilitud perfil

3.4. Estimación Bayesiana

La estimación de los parámetros de interés se puede realizar mediante métodos Bayesianos. Este método utiliza de forma general el teorema de Bayes mostrado en (2.27) para realizar la inferencia de los parámetros de interés $\boldsymbol{\beta}, \boldsymbol{\gamma}$ y s.

Supóngase que se tiene una muestra aleatoria $\mathbf{Y} = (Y_1, Y_2, ..., Y_n)^t$ que tiene distribución ZIP, es decir, $f_{Y_i}(y_i; p_i, \lambda_i) = f(y_i|p_i, \lambda_i) \sim ZIP(p_i, \lambda_i)$. La distribución de los datos $f(\mathbf{y}|\boldsymbol{\theta})$ dependerá de la función liga elegida a través del parámetro p_i.

$$f(\mathbf{y}|\mathbf{p}, \boldsymbol{\lambda}) \propto \prod_{i=1}^{n} p_i + (1-p_i)e^{-\lambda_i} I_{\{y_i=0\}} + (1-p_i)e^{-\lambda_i}\lambda_i^{\{y_i\}} I_{\{y_i>0\}} \tag{3.9}$$

La distribución *a priori* de los parámetros se supondrá independiente

$$\begin{aligned} f(\boldsymbol{\theta}) &= f(\boldsymbol{\beta})f(\boldsymbol{\gamma})f(s) \\ f(\boldsymbol{\beta}) \sim N(\mathbf{0}, \sigma^2 I)\ , f(\boldsymbol{\gamma}) &\sim N(\mathbf{0}, \sigma^2 I) \end{aligned} \tag{3.10}$$

$f(s)$ dependerá de la función liga elegida.

Debido a que no es fácil encontrar las distribuciones condicionales completas de forma cerrada, en este trabajo utilizará el algoritmo de Metropolis Metropolis *et al.* (1953); Ghosh *et al.* (2006) utilizaron el método de aumento de datos.

El procedimiento `MCMC` del software SAS (SAS Institute Inc., 2011b) estima la distribución final de los parámetros $f(\boldsymbol{\theta}|\mathbf{y})$, este procedimiento usa el algoritmo de Metropolis con una distribución de transición normal.

Dicho procedimiento asume que todas las observaciones son independientes; se calcula el logaritmo de la densidad *a posteriori* de la siguiente manera:

$$\log\left(f(\boldsymbol{\theta}|\mathbf{y})\right) = \log\left(f(\boldsymbol{\theta})\right) + \sum_{i=1}^{n} \log\left(f(y_i|\boldsymbol{\theta})\right) \tag{3.11}$$

La rutina `MCMC` utiliza bloque de parámetros y actualiza cada bloque de parámetros mientras mantienen los demás constantes (SAS Institute Inc., 2011a).

Capítulo 4

Estudios de simulación

En este capítulo se presentan los resultados de simulación con el fin de comprobar si los estimadores tienen propiedades deseables, asimismo verificar si el algoritmo tiene problemas de convergencia.

4.1. Generación de variables aleatorias

Para simular los conteos inflados con ceros Y que tienen distribución *ZIP* (2.18), se utilizará el procedimiento descrito por Lambert (1992). Se fija n, $\boldsymbol{\beta} = (0.5, 2)^t$ y $\boldsymbol{\gamma} = (0, 1)^t$ y se considera el caso de dos covariables b y g relacionadas con las medias de los conteos y los ceros estructurales. El valor mínimo para la covariable b se fija en 0 y al valor máximo en 2. En el caso de la covariable g el valor mínimo se fija en -3 y el máximo en 3. Para n dado, se genera una secuencia de valores para las covariables de acuerdo al siguiente esquema:

$$\begin{array}{rcl} b_i & = & 0 + \Delta_b \times i, \\ g_i & = & -3 + \Delta_g \times i, \end{array} \tag{4.1}$$

donde $\Delta_b = (2-0)/(n-1)$, $\Delta_g = (3-(-3))/(n-1) = 6/(n-1)$, $i = 1, ..., n$.

Los valores de b_i y g_i, se utilizan para generar las matrices diseño $\boldsymbol{B}$ y $\boldsymbol{G}$, respectivamente; es decir, la i-ésima fila de las matrices correspondientes se construye de la siguiente forma: $\boldsymbol{B}_i = (1 \;\; b_i)$ y $\boldsymbol{G}_i = (1 \;\; g_i)$, $i = 1, ..., n$.

A continuación se describe algoritmo para generar números aleatorios para la distribución

ZIP.

INICIO;
Elegir un tamaño de muestra n;
Construir las covariables b_i y g_i (4.1), con $i = 1, 2, ..., n$;
Calcular $\lambda_i = e^{0.5+2*b_i}$;
Calcular $p_i = p(g_i, s)$ con la liga NS o $AO2$ (3.3);
$i := 1$;
mientras $i <= n$,
    Generar $U_i \sim U(0, 1)$;
    si $u_i < p_i$, **entonces**
        $Y_i := 0$;
    en otro caso
        $Y_i \sim Poisson(\lambda_i)$
    fin
    $i := i + 1$;
fin
FIN;

Algoritmo 1: Generación de variables ZIP.

En el anexo A se muestra el programa en R (R Core Team, 2015) utilizado para generar los números aleatorios de la distribución *ZIP*.

4.2. Consistencia y especificación incorrecta de la función liga

En esta sección se simularán distintos escenarios con las ligas NS y $AO2$ con el fin de verificar si los estimadores obtenidos son consistentes. La idea en esencia es estimar sesgos y errores estándar y analizar su comportamiento conforme aumenta el tamaño de muestra ($n \to \infty$); Lambert (1992) utilizó este procedimiento para verificar el comportamiento asintótico de los estimadores de $\boldsymbol{\beta}$ y $\boldsymbol{\gamma}$ de la regresión ZIP, bajo la liga *logit*.

En este trabajo se calcularán los errores estándar numéricamente, tomando la raíz cuadrada de la diagonal de la matriz de varianzas-covarianzas de los parámetros (2.5), la cual será estimada mediante el método de Oakes (1999).

El estudio de simulación presentado también permitirá analizar el sesgo de los estimadores cuando se especifica de manera *incorrecta* la función liga. Se consideran 2 casos:

1. Se simulan datos *ZIP* usando la liga $AO2$ con el parámetro τ fijo y posteriormente se estiman los parámetros $\boldsymbol{\beta}$ y $\boldsymbol{\gamma}$ para ciertos valores de τ.

2. Se simulan datos *ZIP* usando la liga *NS* con el parámetro ν fijo, se estiman $\boldsymbol{\beta}$ y $\boldsymbol{\gamma}$ para ciertos valores de ν.

En este estudio de simulación se realizará bajo las siguientes condiciones:

- El número de repeticiones r será de 1000.
- Se considerarán diferentes tamaños de muestra n: 50, 100 y 500.
- Los parámetros verdaderos serán $\boldsymbol{\beta} = \begin{pmatrix} 0.5 \\ 2 \end{pmatrix}$ y $\boldsymbol{\gamma} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$.

En el anexo C se muestra el programa que realiza el algoritmo para realizar la simulación Monte Carlo. Los estudios de simulación utilizados en este apartado, siguen de forma general el siguiente orden:

INICIO;
Elegir un tamaño de muestra n;
Construir las covariables b_i y g_i (4.1), con $i = 1, 2, ..., n$;
Calcular $\lambda_i = e^{0.5+2*b_i}$;
Seleccionar la función liga y fijar el parámetro de forma s;
Calcular $p_i = p(g_i, s)$ (3.3);
$j := 1$;
$r := 1000$;
mientras $j <= r$
    Generar los conteos Y_i con el Algoritmo 1, utilizando λ_i y p_i;
    Estimar los parámetros de interés $\hat{\boldsymbol{\beta}}^{(j)}$ y $\hat{\boldsymbol{\gamma}}^{(j)}$ con un valor s fijo;
    $j := j + 1$;
fin
Calcular el sesgo y el error estándar de $\hat{\boldsymbol{\beta}} = \left\{\hat{\boldsymbol{\beta}}^{(1)}, \hat{\boldsymbol{\beta}}^{(2)}, ..., \hat{\boldsymbol{\beta}}^{(1000)}\right\}$ y $\hat{\boldsymbol{\gamma}} = \left\{\hat{\boldsymbol{\gamma}}^{(1)}, \hat{\boldsymbol{\gamma}}^{(2)}, ..., \hat{\boldsymbol{\gamma}}^{(1000)}\right\}$;
FIN;

Algoritmo 2: Verificación de consistencia de los estimadores.

Cabe recordar que el sesgo de un estimador se calcula como $\mathbb{E}(\hat{\theta}) - \theta$; $\mathbb{E}(\hat{\theta})$ será estimado como el promedio de las estimaciones obtenidas.

A continuación se explicará con mayor detalle los estudios de simulación, primero se analizará el caso para la liga *AO2* y posteriormente se abordará el caso de la liga *NS*.

4.2.1. Liga *AO2*

En este apartado se estudiarán algunas propiedades asintóticas de los estimadores para distintos valores del parámetro de forma τ de la función liga *AO2*.

Se fijará el parámetro de forma verdadero $\tau \to 0$, es decir se generarán las probabilidades verdaderas con la liga *cloglog*; se estimarán los parámetros $\boldsymbol{\beta}$ y $\boldsymbol{\gamma}$ considerando los valores de $\tau = 0, 1, 2$.

Para obtener el sesgo y verificar la consistencia de los estimadores cuando se estima con la liga *AO2*, se utiliza el algoritmo siguiente:

INICIO;
Elegir un tamaño de muestra n;
Construir las covariables b_i y g_i (4.1), con $i = 1, 2, ..., n$;
Calcular $\lambda_i = e^{0.5+2*b_i}$;
Fijar el parámetro de forma $\tau \to 0$ y calcular $p_i = 1 - e^{-e^{g_i}}$ (2.12);
$j := 1$;
$r := 1000$;
mientras $j <= r$
    Simular n datos con la distribución *ZIP* (Algoritmo 1) con λ_i y p_i;
    Estimar los parámetros $\boldsymbol{\beta}^{(j)}$ y $\boldsymbol{\gamma}^{(j)}$ con un valor τ fijo;
    $j := j + 1$;
fin
Obtener el sesgo y la desviación estándar de $\hat{\boldsymbol{\beta}} = \left\{\hat{\boldsymbol{\beta}}^{(1)}, \hat{\boldsymbol{\beta}}^{(2)}, ..., \hat{\boldsymbol{\beta}}^{(1000)}\right\}$ y $\hat{\boldsymbol{\gamma}} = \left\{\hat{\boldsymbol{\gamma}}^{(1)}, \hat{\boldsymbol{\gamma}}^{(2)}, ..., \hat{\boldsymbol{\gamma}}^{(1000)}\right\}$;
FIN;

Algoritmo 3: Algoritmo para comprobar consistencia de los estimadores usando la liga *AO2*.

De acuerdo con la Tabla 4.1, cuando se obtiene $\hat{\beta}_0$, $\hat{\beta}_1$, $\hat{\gamma}_0$ y $\hat{\gamma}_1$ con la liga correcta (*cloglog*), los estimadores son insesgados asintóticamente y su error estándar disminuye cuando el tamaño de muestra n crece.

Sin embargo al obtener $\hat{\beta}_0$, $\hat{\beta}_1$, $\hat{\gamma}_0$ y $\hat{\gamma}_1$ con la función liga incorrecta (*logit* y $AO2(\tau = 2)$), los estimadores de γ_0 y γ_1 son muy sesgados y los errores estándar son más grandes de lo que deberían ser. Note que el problema es aún más grave cuando se asume un valor de τ mayor, por ejemplo los sesgos son más grandes cuando se asume que $\tau = 2$ que cuando $\tau = 1$.

Se puede notar también que la mala especificación de la función liga no afecta demasiado a $\hat{\beta}_0$, $\hat{\beta}_1$, esto se debe a que el algoritmo EM estima de manera separada a $\boldsymbol{\beta}$ y $\boldsymbol{\gamma}$.

Estimación con liga correcta				
LV*		$n = 50$		
-56.8049	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.49	2.01	0.09	1.27
Error est.	0.24	0.31	0.45	0.78
LV		$n = 100$		
-116.24	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.49	2.01	0.04	1.11
Error est.	0.17	0.21	0.23	0.33
LV		$n = 500$		
-593.214	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.50	2.00	0.00	1.01
Error est.	0.07	0.08	0.09	0.10
Estimación con liga *logit*				
LV		$n = 50$		
-57.1314	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.47	2.03	0.94	1.97
Error est.	0.24	0.31	0.67	1.09
LV		$n = 100$		
-116.868	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.47	2.03	0.87	1.75
Error est.	0.17	0.21	0.36	0.48
LV		$n = 500$		
-596.249	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.48	2.03	0.82	1.63
Error est.	0.07	0.09	0.14	0.15
Estimación con liga $AO2$ con $\tau = 2$				
LV		$n = 50$		
-57.4065	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.45	2.04	1.97	2.83
Error est.	0.25	0.31	0.97	1.62
LV		$n = 100$		
-117.497	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.45	2.05	1.85	2.46
Error est.	0.17	0.21	0.53	0.70
LV		$n = 500$		
-599.77	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.46	2.04	1.76	2.28
Error est.	0.07	0.09	0.21	0.22

Tabla 4.1: Propiedades asintóticas de los estimadores para distintos valores de τ.
* Logaritmo de la verosimilitud.

En la Figura 4.1 se muestra el efecto del parámetro de forma τ al estimar las probabilidades p_i de ocurrencia de un cero estructural en la distribución ZIP, para un tamaño de muestra 500. La línea sólida representa la probabilidad verdadera, es decir cuando $\gamma_0 = 0$ y $\gamma_1 = 1$; la línea con guiones representa la estimación de la probabilidad suponiendo la liga correcta (*cloglog*); la línea punteada muestra la probabilidad estimada con la función liga *logit* y finalmente la línea con guiones y puntos representa la probabilidad estimada cuando se asume la liga $AO2$ con $\tau = 2$.

Note que existe una gran diferencia entre las probabilidades estimadas con ligas distintas a la *cloglog*. Se puede observar que las probabilidades estimadas se alejan más de la probabilidad verdadera a medida que el parámetro de sesgo τ incrementa.

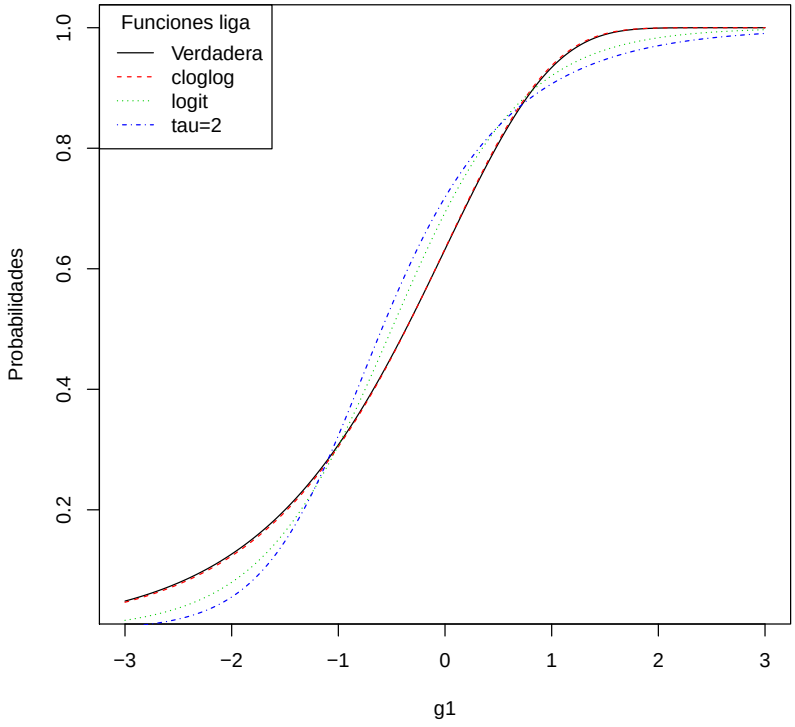

Figura 4.1: Estimación de las probabilidades variando el parámetro τ.

4.2.2. Liga *NS*

En este apartado se estudiará algunas propiedades asintóticas de los estimadores para distintos valores del parámetro de forma ν de la función liga *NS*.

El estudio de simulación será el siguiente:

INICIO;
Elegir un tamaño de muestra n;
Construir las covariables b_i y g_i (4.1), con $i = 1, 2, ..., n$;
Calcular $\lambda_i = e^{0.5+2*b_i}$;
Fijar el parámetro de forma $\nu = 2$;
Calcular $p_i = F_W(g_i, 2)$ con la función `psn(x=`g_i`,alpha=2)` de la librería `sn` de `R`;
$j := 1$;
$r := 1000$;
mientras $j <= r$
    Simular n datos con la distribución *ZIP* (Algoritmo 1) con λ_i y p_i;
    Estimar los parámetros $\boldsymbol{\beta}^{(j)}$ y $\boldsymbol{\gamma}^{(j)}$ con un valor ν fijo;
    $j := j + 1$;
fin
Obtener el sesgo y la desviación estándar de $\hat{\boldsymbol{\beta}} = \left\{\hat{\boldsymbol{\beta}}^{(1)}, \hat{\boldsymbol{\beta}}^{(2)}, ..., \hat{\boldsymbol{\beta}}^{(1000)}\right\}$ y $\hat{\boldsymbol{\gamma}} = \left\{\hat{\boldsymbol{\gamma}}^{(1)}, \hat{\boldsymbol{\gamma}}^{(2)}, ..., \hat{\boldsymbol{\gamma}}^{(1000)}\right\}$;
FIN;

Algoritmo 4: Algoritmo para comprobar consistencia de los estimadores usando la liga *NS*.

La Tabla 4.2 muestra que cuando se estiman los parámetros de interés β_0, β_1, γ_0 y γ_1 con la liga correcta (*NS* con $\nu = 2$), los estimadores son insesgados asintóticamente y su error estándar disminuye con tamaños de muestra n grandes.

Por otro lado, cuando se estiman los parámetros utilizando la funciones liga incorrectas, (*NS* con $\nu = 1$ y *probit*), los estimadores $\hat{\gamma}_0$ y $\hat{\gamma}_1$ son sesgados y tienen mayor error estándar que los estimadores que se calcularon con $\nu = 2$.

Note que el sesgo y error estándar de $\hat{\gamma}_0$ y $\hat{\gamma}_1$ son mayores cuando se estiman el parámetro ν se aleja del verdadero valor, en otras palabras, el error de estimación es mayor cuando se asume la liga *probit* que cuando se utiliza la liga con *NS* con $\nu = 1$.

No se consideraron valores negativos en esta simulación debido a la propiedad de simetría de la distibución normal sesgada con su parámetro de forma, dicha propiedad está enunciada en (2.29).

Estimación con la liga correcta				
LV*		$n = 50$		
-78.16	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.50	2.00	-0.05	1.12
Error est.	0.17	0.17	0.26	0.27
LV		$n = 100$		
-159.72	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.50	2.00	-0.04	1.10
Error est.	0.12	0.12	0.19	0.19
LV		$n = 500$		
-809.48	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.50	2.00	-0.01	1.00
Error est.	0.05	0.05	0.09	0.09
Estimación con la liga *NS* con $\nu = 1$				
LV		$n = 50$		
-78.12	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.50	2.00	-0.40	1.40
Error est.	0.17	0.17	0.38	0.42
LV		$n = 100$		
-159.72	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.50	2.00	-0.37	1.30
Error est.	0.12	0.12	0.28	0.30
LV		$n = 500$		
-809.79	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.50	2.00	-0.30	1.20
Error est.	0.05	0.05	0.11	0.11
Estimación con la liga *probit*				
LV		$n = 50$		
-78.07	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.50	2.00	-1.30	1.80
Error est.	0.17	0.17	0.65	0.74
LV		$n = 100$		
-159.76	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.50	2.00	-1.10	1.60
Error est.	0.12	0.12	0.36	0.41
LV		$n = 500$		
-810.11	$\hat{\beta}_0$	$\hat{\beta}_1$	$\hat{\gamma}_0$	$\hat{\gamma}_1$
Media	0.50	2.00	-1.00	1.40
Error est.	0.05	0.05	0.13	0.14

Tabla 4.2: Propiedades asintóticas de los estimadores para distintos valores de ν.
* Logaritmo de la verosimilitud.

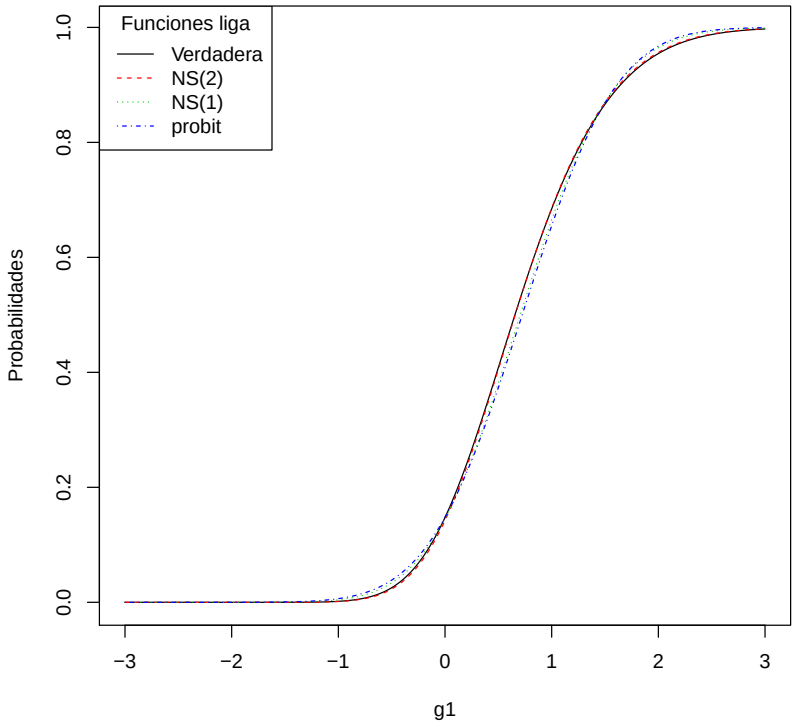

Figura 4.2: Estimación de las probabilidades para distintos valores de ν.

En la Figura 4.2 se comparan las probabilidades de ocurrencia de un cero estructural en la regresión *ZIP* con las probabilidades estimadas con los distintos valores de ν.

La línea sólida representa la probabilidad verdadera, la línea con guiones representa la estimación de la probabilidad suponiendo la liga *NS* con $\nu = 2$, la línea punteada estima la probabilidad con $\nu = 1$ y finalmente la línea con guiones y puntos muestra la probabilidad estimada cuando se asume que $\nu = 0$.

A pesar de que los estimadores son sesgados cuando se estima usando la liga *probit* o liga *NS* con $\nu = 1$, las probabilidades estimadas son cercanas a la verdadera probabilidad, por lo que se puede decir que se tiene un ajuste aceptable.

Cabe señalar que este programa presentó problemas de numéricos debido a que no se podía evaluar de manera correcta los valores iniciales. Para lograr que los estimadores fueran calculados de manera correcta, se fijaron semillas iniciales en $\beta_0^{(0)} = 0.55$, $\beta_1^{(0)} = 1.95$, $\gamma_0^{(0)} = 0.3$, $\gamma_1^{(0)} = 0.90$.

4.3. Error de cuadrado medio de los estimadores

En este apartado se estudia el error cuadrado medio (ECM) de los estimadores al variar el parámetro de forma de las funciones liga *AO2* y *NS*; Czado y Santner (1992) realizaron un estudio similar donde compararon la ligas *logit*, Box-Cox, Cauchy y Burr.

Cabe recordar que el Error Cuadrado Medio ECM de un estimador T se calcula como $\mathbb{V}(T) + (\theta - \mathbb{E}(T))^2$ (Mood *et al.*, 1974), esto es la varianza del estimador $\mathbb{V}(T)$ más el sesgo del estimador elevado al cuadrado.

Para llevar a cabo este ejercicio, se creó una rejilla de valores, que depende de la función liga a estudiar. A continuación, se describe de manera general el algoritmo a utilizar:

INICIO;
Crear una rejilla de valores s_k de tamaño m, con $k = 1, 2, ..., m$ para el parámetro de forma s;
Elegir un tamaño de muestra n;
Construir las covariables b_i y g_i (4.1), con $i = 1, 2, ..., n$;
Calcular $\lambda_i = e^{0.5+2*b_i}$;
Fijar el parámetro de forma $s = s_0$;
Calcular $p_i = p\left(g_i, s_0\right)$ (3.3);
$j := 1$;
$r := 1000$;
desde $k = 1$, **hasta** m, **incrementando** 1
    mientras $j <= r$
        Simular n datos con la distribución *ZIP* (Algoritmo 1) con λ_i y p_i;
        Estimar los parámetros $\boldsymbol{\beta}_k^{(j)}$ y $\boldsymbol{\gamma}_k^{(j)}$ con un valor s_k fijo;
        $j := j + 1$;
    fin
    Calcular el error cuadrado medio de $\hat{\boldsymbol{\beta}}_k = \left\{\hat{\boldsymbol{\beta}}_k^{(1)}, \hat{\boldsymbol{\beta}}_k^{(2)}, ..., \hat{\boldsymbol{\beta}}_k^{(1000)}\right\}$ y $\hat{\boldsymbol{\gamma}}_k = \left\{\hat{\boldsymbol{\gamma}}_k^{(1)}, \hat{\boldsymbol{\gamma}}_k^{(2)}, ..., \hat{\boldsymbol{\gamma}}_k^{(1000)}\right\}$;
fin
FIN;

Algoritmo 5: Algoritmo general para calcular el error cuadrado medio de los estimadores.

En las siguientes secciones se presentan los resultados obtenidos para cada función liga.

4.3.1. Liga *AO2*

En este apartado, se mostrará el error cuadrado medio que se obtiene al variar el parámetro τ. Se elegirá una rejilla de 21 valores desde 0 hasta 5 y se estimarán los vectores de parámetros $\boldsymbol{\beta}$ y $\boldsymbol{\gamma}$.

En este ejercicio se fijó el valor de τ en 2, de tal forma que las probabilidades fuesen sesgadas. Con esto se pretende verificar si el error cuadrado de los estimadores propuestos, es mínimo cuando un valor de la rejilla sea cercano a 2.

Se seguirá el siguiente procedimiento:

INICIO;
Crear una rejilla de valores $\tau_1, \tau_2, ..., \tau_{21}$ para el parámetro de forma τ;
Elegir un tamaño de muestra n;
Construir las covariables b_i y g_i (4.1), con $i = 1, 2, ..., n$;
Calcular $\lambda_i = e^{0.5+2*b_i}$;
Fijar el parámetro de forma $\tau = 2$;
Calcular $p_i = 1 - (1 + 2e^{g_i})^{-\frac{1}{2}}$ (2.14);
$j := 1$;
$r := 1000$;
desde $k = 1$, **hasta** 21, **incrementando** 1
    mientras $j <= r$
        Simular n datos con la distribución *ZIP* (Algoritmo 1) con λ_i y p_i;
        Estimar los parámetros $\boldsymbol{\beta}_k^{(j)}$ y $\boldsymbol{\gamma}_k^{(j)}$ con un valor $\tau = \tau_k$ fijo;
        $j := j + 1$;
    fin
    Calcular el error cuadrado medio de $\hat{\boldsymbol{\beta}}_k = \left\{\hat{\boldsymbol{\beta}}_k^{(1)}, \hat{\boldsymbol{\beta}}_k^{(2)}, ..., \hat{\boldsymbol{\beta}}_k^{(1000)}\right\}$ y $\hat{\boldsymbol{\gamma}}_k = \left\{\hat{\boldsymbol{\gamma}}_k^{(1)}, \hat{\boldsymbol{\gamma}}_k^{(2)}, ..., \hat{\boldsymbol{\gamma}}_k^{(1000)}\right\}$;
fin
FIN;

Algoritmo 6: Algoritmo para calcular el error cuadrado medio de los estimadores usando la liga *AO2*.

En la Figura 4.3 se puede observar que el error cuadrado medio se incrementa al incrementarse el parámetro de forma; note que para muestras pequeñas, un error de especificación conlleva a errores más grandes. Note también que en general el método propuesto minimizó el error cuadrado medio de γ_0 y γ_1 cuando τ se encontraba cerca de 2.

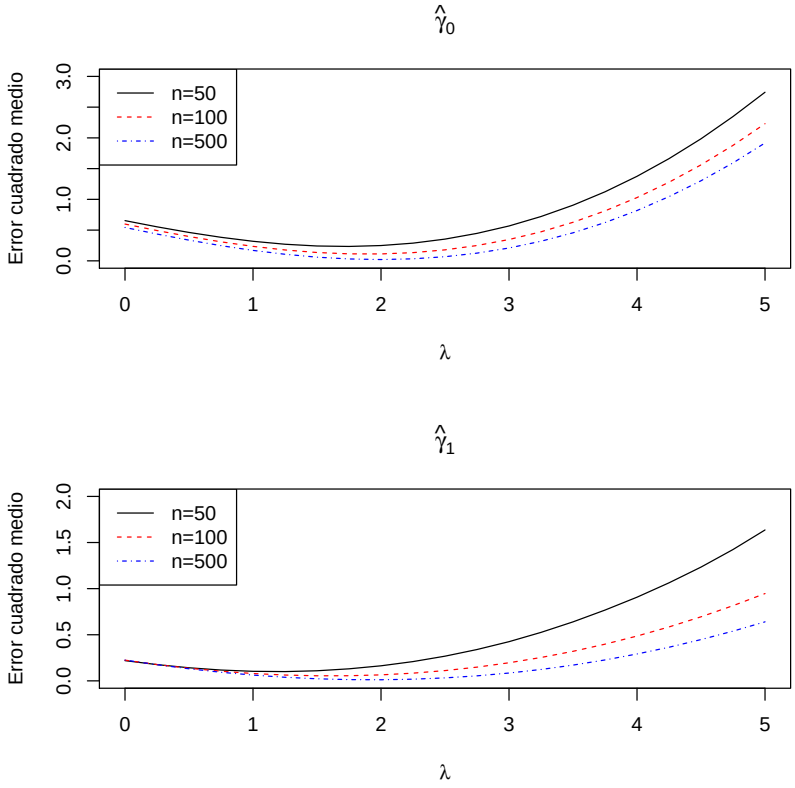

Figura 4.3: Error cuadrado medio de los estimadores en función de τ.

4.3.2. Liga *NS*

En este apartado, se mostrará el error cuadrado medio que se obtiene al variar el parámetro ν. Se elegirá una rejilla de 16 valores equidistantes desde 1 hasta 2.5 y se estimarán los parámetros fijando el valor de ν en 2.

Debido a que dicho algoritmo es muy lento, se eligió reducir el número de réplicas de 1000 a tan solo 100 y únicamente se considerarán los tamaños de muestra 50 y 100. Cada una de las simulaciones tardó aproximadamente 3 horas y en varias ocasiones el algoritmo tuvo problemas de convergencia, por lo que tuvo que volver a simular.

Se seguirá el siguiente procedimiento:

INICIO;
Crear una rejilla de valores $\nu_1, \nu_2, ..., \nu_{21}$ para el parámetro de forma ν;
Elegir un tamaño de muestra n;
Construir las covariables b_i y g_i (4.1), con $i = 1, 2, ..., n$;
Calcular $\lambda_i = e^{0.5+2*b_i}$;
Fijar el parámetro de forma $\nu = 2$;
Calcular $p_i = F_W(g_i, 2)$ con la función `psn(x=`g_i`,alpha=2)` de la librería `sn` de `R`;
$j := 1$;
$r := 100$;
desde $k = 1$, **hasta** 21, **incrementando** 1
    mientras $j <= r$
        Simular n datos con la distribución *ZIP* (Algoritmo 1) con λ_i y p_i;
        Estimar los parámetros $\boldsymbol{\beta}_k^{(j)}$ y $\boldsymbol{\gamma}_k^{(j)}$ con un valor $\nu = \nu_k$ fijo;
        $j := j + 1$;
    fin
    Calcular el error cuadrado medio de $\hat{\boldsymbol{\beta}}_k = \left\{\hat{\boldsymbol{\beta}}_k^{(1)}, \hat{\boldsymbol{\beta}}_k^{(2)}, ..., \hat{\boldsymbol{\beta}}_k^{(1000)}\right\}$ y $\hat{\boldsymbol{\gamma}}_k = \left\{\hat{\boldsymbol{\gamma}}_k^{(1)}, \hat{\boldsymbol{\gamma}}_k^{(2)}, ..., \hat{\boldsymbol{\gamma}}_k^{(1000)}\right\}$;
fin
FIN;

Algoritmo 7: Algoritmo para calcular el error cuadrado medio de los estimadores usando la liga *NS*.

Este ejercicio de simulación fue poco preciso debido a que requiere muchos recursos computacionales para mostrar gráficas más finas; sin embargo se puede apreciar que conforme se incrementa el tamaño de muestra, el error estándar presenta cierta tendencia a disminuir y se esperaría que conforme se incremente el número de réplicas, la líneas presentarían menos variabilidad.

En la Figura 4.4 se puede observar que el error cuadrado medio se incrementa cuando el parámetro de forma ν está cerca del 1. Note que el algoritmo minimizó el error cuadrado medio de γ_0 y γ_1 para valores cercanos a 2 y vuelven a incrementarse a partir del 2.5.

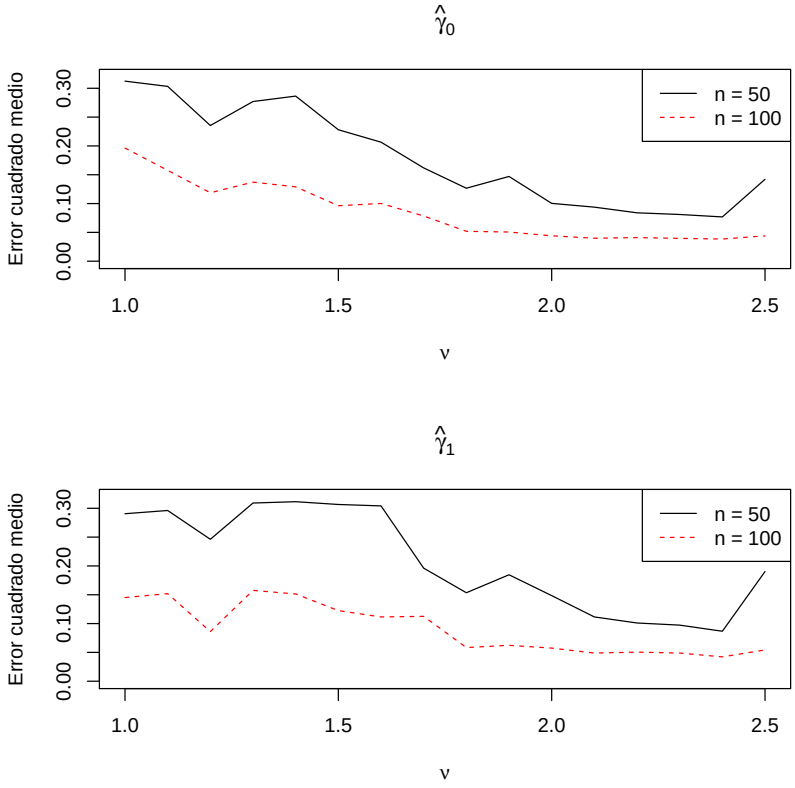

Figura 4.4: Error cuadrado medio de los estimadores en función de ν.

Capítulo 5

Ejemplos de Aplicación

5.1. Demanda de cuidado médico para ancianos

Deb y Trivedi (1997) analizaron datos de la Encuesta Nacional de Gastos Médicos (National Medical Expenditure Survey) de Estados Unidos de 1987; se sabe que estos datos tienen una gran proporción de conteos que son ceros (ver Figura 5.1). Estos autores utilizaron modelos de mezclas finitas y los compararon con modelos de regresión *hurdle* y binomial negativa.

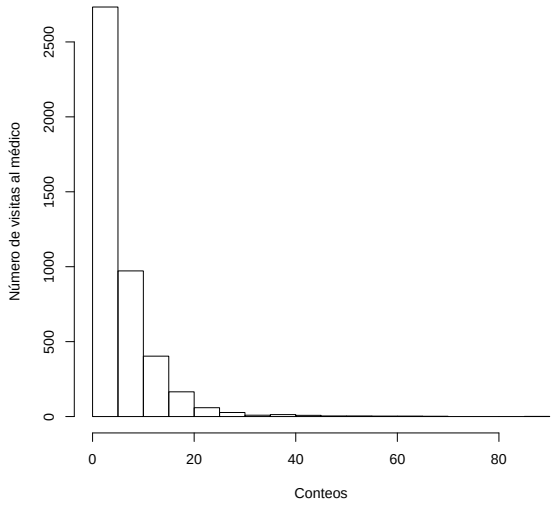

Figura 5.1: Frecuencia de visitas al médico.

Zeileis *et al.* (2008) retomaron estos datos y ajustaron diversos modelos de conteo, entre

ellos la regresión binomial negativa inflada con ceros. Consideraron al número de visitas al médico (OFS) como variable respuesta, y utilizaron como variables explicativas de los conteos a las variables número de días en el hospital (HOSP), estado de salud (HEALTH), número de condiciones crónicas (NUMCHRON), género (GENDER), número de años de educación (SCHOOL) y seguro privado (PRIVINS), mientras que las variables que explican la presencia de ceros fueron HOSP, NUMCHRON, PRIVINS y GENDER. La variable HEALTH es categórica con niveles: pobre (poor), promedio (average) y excelente (excellent); la variable GENDER es categórica con 2 niveles: femenino (female) y masculino (male), mientras que PRIVINS es una variable con categórica con niveles: si (yes) y no (no).

Se pretende modelar el número de visitas al médico (OFS) utilizando las funciones liga mencionadas previamente; se utilizaran las mismas covariables que modelanron Zeileis *et al.* (2008).

5.2. Uso de la liga Aranda-Ordaz Asimética para seleccionar modelos

Se desea decidir cual función liga es más adecuada para estimar las probabilidades de ocurrencia de cero, mediante el uso del método propuesto. Se buscará el parámetro de forma τ que maximice la verosimilitud en una rejilla de 0 a 3 con incrementos de 0.25.

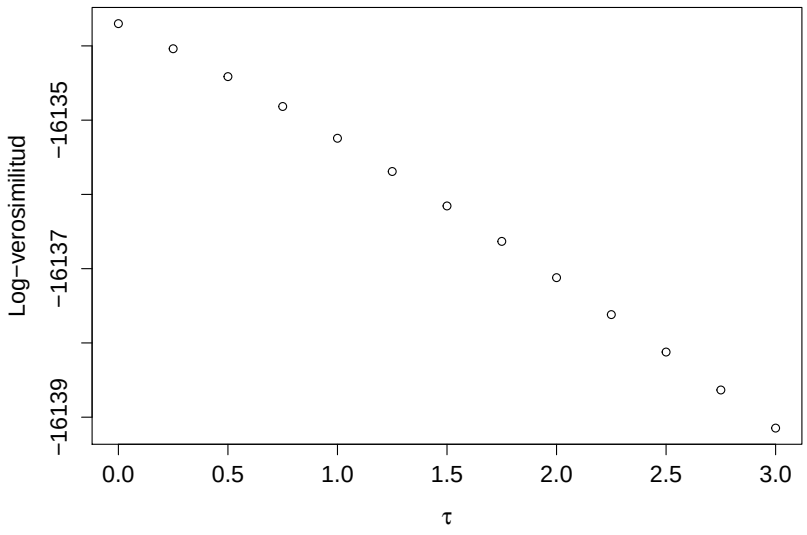

Figura 5.2: Estimación de parámetro τ para los datos de Deb y Trivedi (1998).

De acuerdo con la Figura 5.2, el valor que maximiza la verosimilitud es 0, lo cual implicaría

que una liga *cloglog* se ajusta mejor que la *logit*.

En la Tabla 5.1 se muestra los coeficientes y errores estándar de ambos modelos, note que con los resultados mostrados en la table es posible construir intervalos de confianza y hacer pruebas de Wald para los coeficientes de regresión.

Si se obtiene el Criterio de Información de Akaike (AIC) para ambos modelos, se puede observar que el AIC para el modelo que utiliza la liga *cloglog* 32295.4, es menor al AIC del modelo con la liga *logit* 32298.49. Por lo tanto esto indicaría que para este conjunto de datos, la liga *cloglog* es más adecuada que la liga *logit*.

Liga cloglog			Liga logit		
Conteos	Coeficientes	Error est.	Conteos	Coeficientes	Error est.
Intercepto	1.4062	0.0241	Intercepto	1.4054	0.0241
hosp	0.159	0.006	hosp	0.159	0.006
healthpoor	0.2535	0.0177	healthpoor	0.2535	0.0177
healthexcellent	-0.3069	0.0312	healthexcellent	-0.3074	0.0312
numchron	0.1017	0.0047	numchron	0.1018	0.0047
gendermale	-0.0624	0.013	gendermale	-0.0622	0.013
school	0.0191	0.0018	school	0.0191	0.0018
privinsyes	0.0807	0.0171	privinsyes	0.0806	0.0171
Ceros	Coeficientes	Error est.	Ceros	Coeficientes	Error est.
Intercepto	-0.3368	0.1182	Intercepto	-0.0607	0.1404
hosp	-0.2895	0.0852	hosp	-0.3058	0.0911
numchron	-0.499	0.0404	numchron	-0.5397	0.0441
privinsyes	-0.6543	0.0889	privinsyes	-0.7526	0.1021
school	-0.0485	0.0106	school	-0.0555	0.0121
gendermale	0.3672	0.0789	gendermale	0.4192	0.0891

Tabla 5.1: Resultado de los modelos con liga *cloglog* y *logit*.

A continuación se resuelve el problema de estimación desde el punto de vista Bayesiano. Debido a que el espacio paramétrico de τ es positivo, se puede asumir una distribución *a priori* Lognormal con media igual a 1 y varianza 1 para τ.

Se estimó un modelo *ZIP* con liga *AO2* con las siguientes condiciones:

- $\boldsymbol{\beta} \sim N(\mathbf{0}, 10^6 \times \mathbf{I})$.
- Valores iniciales: $\boldsymbol{\beta}^{(0)} = (0,0,0,0,0,0,0,0)^{(t)}$.
- $\boldsymbol{\gamma} \sim N(\mathbf{0}, 10^6 \times \mathbf{I})$.
- Valores iniciales: $\boldsymbol{\gamma}^{(0)} = (0,0,0,0,0,0)^{(t)}$.
- $\tau \sim LogN(1,1)$.

- Valor inicial: $\tau^{(0)} = 0.2$.

Para estimar las densidades *a posteriori* de los parámetros de interés $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$ y τ se emplearon 50,000 muestras y 10,000 fueron utilizadas de calentamiento eliminando una de cada 10 con una semilla inicial de 1; se utilizó el software SAS (SAS Institute Inc., 2011b) con el procedimiento `MCMC`. Esta simulación tardó alrededor de 15 minutos.

Bajo estas condiciones, la distribución *a posteriori* no convergió a la distribución estacionaria como se muestra en la Figura 5.3.

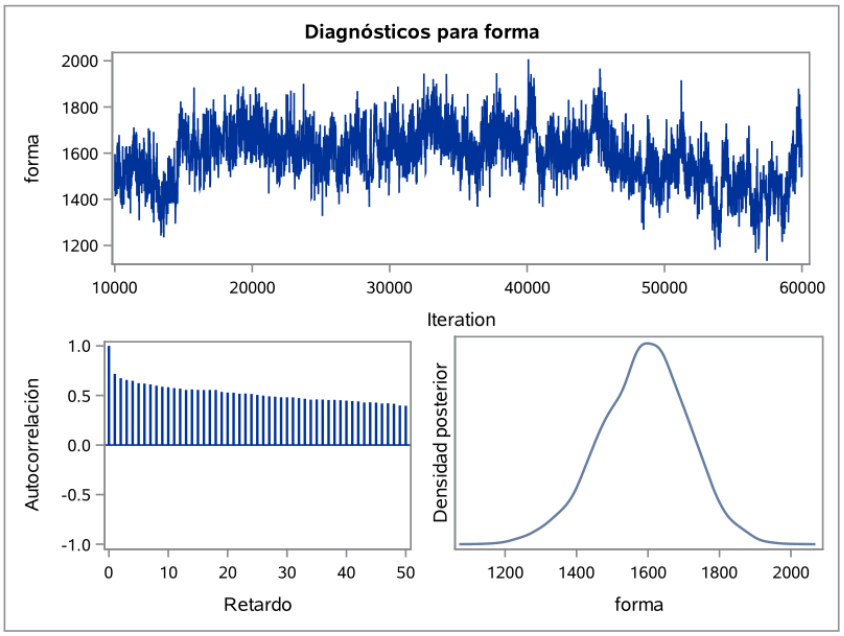

Figura 5.3: No convergencia de la distribución *a posteriori* del parámetro de forma τ.

Se volvió a simular la distribución *a posteriori* utilizando una distribución *a priori* con una alta densidad en el intervalo 0, 1; se utilizó una distribución lognormal con media e^{-2} y varianza 1, utilizando un valor inicial de 0.2.

En la Tabla 5.2 se muestran los resultados de la estimación mediante el método Bayesiano. Note que los resultados, tanto de estimación puntual como errores estándar, son muy cercanos a la estimación con el algoritmo EM.

En la Figura 5.4 se puede apreciar que la distribución final tiene una mayor masa de probabilidad en un intervalo cercano a cero, por lo que sugeriría que es más correcto elegir una liga cloglog para dicho conjunto de datos.

Conteos	Media	Error estándar	HDP 0.025	Mediana	HDP 0.975
Intercepto	1.4056	0.0239	1.3613	1.4054	1.4541
hosp	0.1586	0.0060	0.1468	0.1587	0.1707
healthpoor	0.2532	0.0172	0.2180	0.2530	0.2851
healthexcellent	-0.3078	0.0304	-0.3710	-0.3074	-0.2527
numchron	0.1018	0.0046	0.0931	0.1018	0.1114
gendermale	-0.0627	0.0129	-0.0887	-0.0627	-0.0379
school	0.0192	0.0018	0.01688	0.0157	0.0231
privinsyes	0.0803	0.0171	0.0471	0.0800	0.1136
Ceros	Media	Error estándar	HDP 0.025	Mediana	HDP 0.975
Intercepto	-0.2918	0.1308	-0.5577	-0.2950	-0.0482
hosp	-0.3002	0.0869	-0.4697	-0.2971	-0.1327
numchron	-0.5078	0.0417	-0.5941	-0.5075	-0.4313
privinsyes	-0.6693	0.033	-0.8538	-0.6690	-0.4824
school	-0.0498	0.0108	-0.0708	-0.0498	-0.0284
male	0.3759	0.0803	0.2200	0.3373	0.5349
Forma	Media	Error estándar	HDP 0.025	Mediana	HDP 0.975
$\hat{\tau}$	0.1780	0.1848	0.00548	0.1195	0.5247

Tabla 5.2: Estimación de parámetros mediante el método bayesiano.

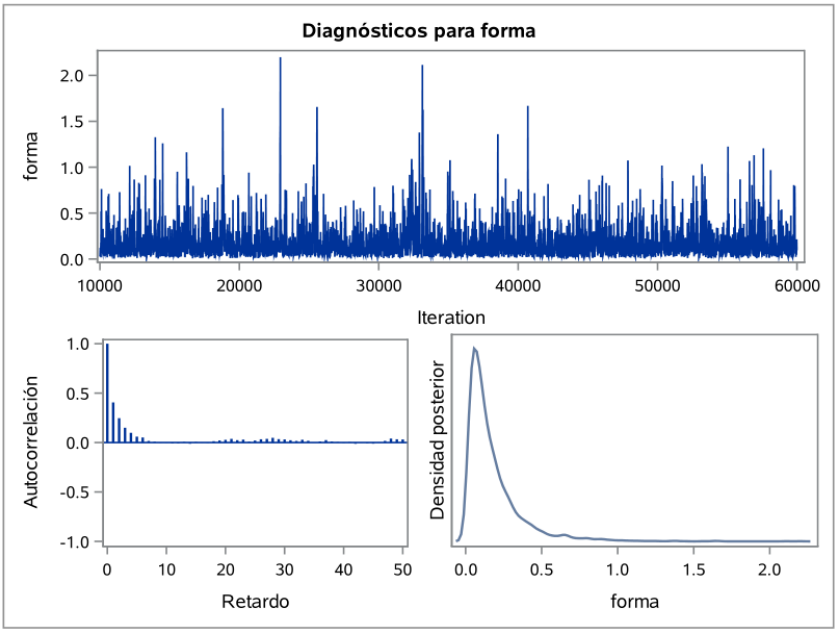

Figura 5.4: Convergencia de la distribución final de τ.

5.3. Uso de la liga Normal Sesgada para elegir un mejor modelo

En esta sección se utiliza la liga Normal Sesgada para verificar si las probabilidades pueden modelarse correctamente con una liga *probit*.

Al igual que en la sección anterior, se buscará el parámetro de forma ν que maximice la verosimilitud. Se observó que este método de estimación es muy lento, por lo que se decidió usar una rejilla de -5 a 5 con incrementos de 1.

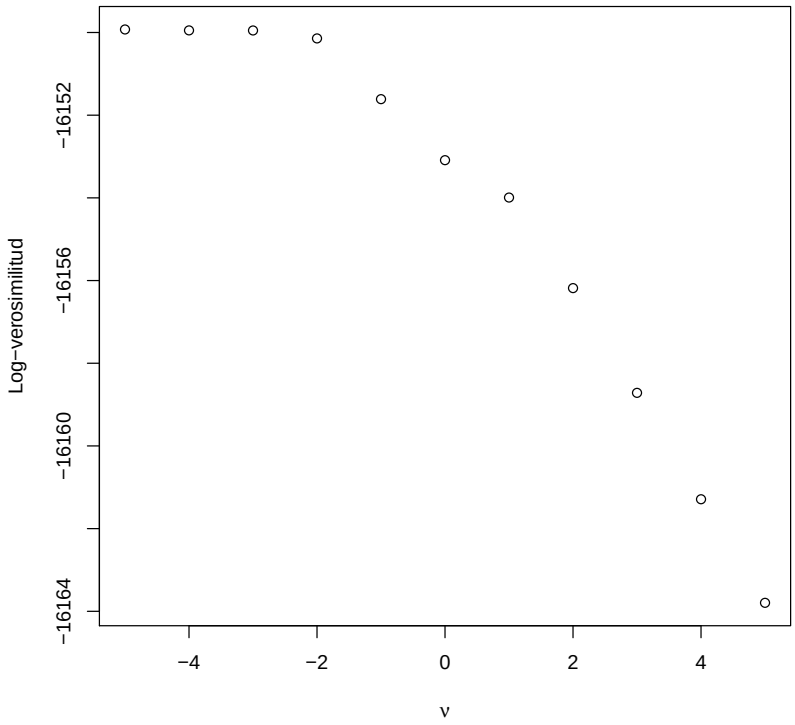

Figura 5.5: Verosimilitud para elegir el parámetro de la liga normal sesgada.

Como se puede observar en la Figura 5.5, la log-verosimiltud cuando $\nu < -2$ es mayor que la verosimilitud cuando $\nu = 0$, por lo que se sugiere utilizar una liga Normal sesgada con parámetro de forma negativo para modelar las probabilidades de ocurrencia de ceros estructurales.

Liga NS($\nu = -2$)			Liga Probit NS($\nu = 0$)		
Conteos	Coeficientes	Error est.	Conteos	Coeficientes	Error est.
Intercepto	1.4059	0.0241	Intercepto	1.4050	0.0241
hosp	0.1591	0.0061	hosp	0.1591	0.006
healthpoor	0.2536	0.0177	healthpoor	0.2536	0.0177
healthexcellent	-0.3074	0.0310	healthexcellent	-0.3077	0.0312
numchron	0.1018	0.0047	numchron	0.1019	0.0047
gendermale	-0.0622	0.0130	gendermale	-0.0624	0.013
school	0.0191	0.0019	school	0.0192	0.0018
privinsyes	0.0806	0.0171	privinsyes	0.0810	0.0171
Ceros	Coeficientes	Error est.	Ceros	Coeficientes	Error est.
Intercepto	-0.7013	0.1314	Intercepto	-0.1273	0.1184
hosp	-0.1117	0.0641	hosp	-0.1233	0.0572
numchron	-0.2280	0.0335	numchron	-0.2684	0.0300
privinsyes	-0.3399	0.0962	privinsyes	-0.4284	0.0864
school	-0.0245	0.0111	school	-0.0303	0.0099
gendermale	0.1849	0.0802	gendermale	0.2282	0.0718

Tabla 5.3: Resultados de la estimación de parámetros mediante la liga Normal sesgada.

En la Tabla 5.3 se muestran los resultados de la estimación de parámetros utilizando la liga Normal Sesgada con parámetro de forma $\nu = -2$ y $\nu = 0$.

Para el modelo propuesto (liga NS con $\nu = -2$), se obtuvo una log-verosimilitud de -16139.43 y un criterio AIC de 32306.86; sin embargo el algoritmo tomó alrededor de 4 horas para estimar los parámetros. Por otro lado, utilizando la liga usual (liga *probit*), la log-verosimilitud fue -16142.46 y el AIC que se obtuvo fue de 32312.92.

Capítulo 6

Conclusiones y Recomendaciones

De los resultados obtenidos en esta investigación se pudo observar que la liga *NS*, aunque permite modelar la probabilidad de ocurrencia de cero estructurales p_i de manera flexible, en general es complicado estimar su parámetro de forma ν. En muchas simulaciones fue difícil obtener tanto su valor debido a que el algoritmo es lento o no es posible encontrar un valor que maximice la verosimilitud.

Por otra parte, la liga *AO2* fue más fácil de estimar debido a que tiene forma cerrada, lo cual hace que el algoritmo sea más rápido. Se recomienda utilizar esta función liga cuando se sospecha que las probabilidades pueden modelarse con liga *logit* o *cloglog*.

Es posible implementar métodos Bayesianos más elaborados para estimar los parámetros de forma de ambas ligas; Bazán *et al.* (2006) lograron estimar el parámetro ν de la liga Normal Sesgada en modelos de regresión binaria, utilizando la técnica de aumento de datos.

Referencias

Achen, C. H. (2002). Toward A New Political Methodology: Microfoundations And Art. *Annual Review of Political Science*, 5, 1, 423–450.

Agresti, A. (2007). *An Introduction to Categorical Data Analysis*. Wiley Series in Probability and Statistics. Wiley. ISBN 9780470114742.

Aranda-Ordaz, F. J. (1981). On Two Families of Transformations to Additivity for Binary Response Data. *Biometrika*, 68, 2, 357–363.

Azzalini, A. (1985). A Class of Distributions which Includes the Normal Ones. *Scandinavian Journal of Statistics*, 12, 2, pp. 171–178. ISSN 03036898.

Azzalini, A. (2014). *The R Package* `sn`*: The Skew-Normal and Skew-t Distributions (version 1.1-2)*. Università di Padova, Italia.

Bazán, J. L., Branco, M. D. y Bolfarine, H. (2006). A Skew Item Response Model. *Bayesian Analysis*, 1, 4, 861–892.

Cameron, A. y Trivedi, P. (2005). *Microeconometrics: Methods and Applications*. Cambridge University Press.

Czado, C. y Santner, T. J. (1992). The Effect Of Link Misspecification On Binary Regression Inference. *Journal of Statistical Planning and Inference*, 33, 2, 213 – 231.

Deb, P. y Trivedi, P. K. (1997). Demand for Medical Care by the Elderly: a Finite Mixture Approach. *Journal of Applied Econometrics*, 12, 3.

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*, 39, 1, 1–38.

Dobson, A. J. (2002). *An Introduction to Generalized Linear Models*. Chapman & Hall.

Everitt, B. y Hand, D. J. (1981). *Finite Mixture Distributions*. Chapman and Hall, London.

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

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, 6, 721–741.

Ghosh, S., Mukhopadhyay, P. y Lu, J.-C. (2006). Bayesian Analysis of Zero-Inflated Regression Models. *Journal of Statistical Planning and Inference*, 136, 4, 1360–1375.

Hastings, W. K. (1970). Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57, 1, pp. 97–109.

Heilbron, D. C. (1989). Generalized linear models for altered zero probabilities and overdispersion in count data. Technical report, University of California, San Francisco, Departament of Epidemiology and Biostatistics.

Heilbron, D. C. (1994). Zero-altered and other regression models for count data with added zeros. *Biometrical Journal*, 36, 531–547.

Hoff, P. (2009). *A First Course in Bayesian Statistical Methods*. Springer New York.

Huang, Y. (2002). Robustness of Choice of Number of Doses for Maximum Likelihood Estimation of the ED50 in Bioassay. *Statistics in Medicine*, 21, 15, 2215–2223.

Lambert, D. (1992). Zero-Inflated Poisson Regression, with an Application to Defects in Manufacturing. *Technometrics*, 34, 1, 1–14.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. y Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *The Journal Of Chemical Physics*, 21, 6, 1087–1092.

Mood, A., Graybill, F. y Boes, D. (1974). *Introduction to the Theory of Statistics*. International Student edition. McGraw-Hill. ISBN 9780070428645.

Morgan, B. J. T. (1992). *Analysis of Quantal Response Data*. Monographs on statistics and applied probability; 46. Chapman and Hall.

Mouatassim, Y. y Ezzahid, E. H. (2012). Poisson regression and Zero-inflated Poisson regression: application to private health insurance data. *European Actuarial Journal*, 2, 2, 187–204.

Mullahy, J. (1986). Specification and Testing of Some Modified Count Data Models. *Journal of Econometrics*, 33, 3.

Naya, H., Urioste, J. I., Chang, Y., Rodrigues-Motta, M., Kremer, R. y Gianola, D. (2008). A Comparison Between Poisson and Zero-Inflated Poisson Regression Models with an Application to Number of Black Spots in Corriedale Sheep. *Genetics Selection Evolution*, 40, 4, 379–394.

Nelder, J. A. y Mead, R. (1965). A Simplex Algorithm for Function Minimization. *Computer Journal*, 4, 7, 308–313.

Oakes, D. (1999). Direct Calculation of the Information Matrix via the EM. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61, 2, 479–482. ISSN 1467-9868.

Pal, N., Lim, W. K. y Thongteeraparp, A. (2012). Inferences on The Standard Skew-Normal Distribution. *Thailand Statistician*, 10, 2, 225–246. Contributed paper.

Pawitan, Y. (2001). *In All Likelihood: Statistical Modelling and Inference Using Likelihood*. Oxford science publications. OUP Oxford.

Press, W., Flannery, B., Teukolsky, S. y Vetterling, W. (1992). *Numerical Recipes in FORTRAN 77: The Art of Scientific Computing*, tomo 1 de *Numerical Recipes in FORTRAN*. Cambridge University Press.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Robert, C. P. y Casella, G. (2004). *Monte Carlo Statistical Methods*. Springer-Verlag.

Sartori, N. (2006). Bias Prevention of Maximum Likelihood Estimates for Scalar Skew Normal and Skew t Distributions. *Journal of Statistical Planning and Inference*, 136, 12, 4259 – 4275.

SAS Institute Inc. (2011a). *SAS/STAT 9.3 Users Guide*. SAS Institute Inc., Cary, NC.

SAS Institute Inc. (2011b). *SAS/STAT Software, Version 9.3*. Cary, NC.

Sprott, D. (2000). *Statistical inference in science*. Springer.

Tanner, M. A. (1996). *Tools for Statistical Inference: Methods for the Exploration of Posterior Distributions and Likelihood Functions*. Springer-Verlag.

Watanabe, M. y Yamaguchi, K. (2003). *The EM Algorithm and Related Statistical Models*. CRC Press.

Zeileis, A., Kleiber, C. y Jackman, S. (2008). Regression Models for Count Data in R. *Journal of Statistical Software*, 27, 8.

Anexos

Anexo A: Estimación de parámetros en R

En este apartado se muestran los códigos utilizados en el presente trabajo para calcular los parámetros de interés de la regresión *ZIP* con liga normal sesgada y Aranda-Ordaz 2. Los métodos de simulación se presentan en la sección siguiente.

En las siguientes funciones se necesita al vector de observaciones `y` de dimensión n, además `X.Poi` y `X.Cero` deben ser **matrices diseño** de dimensiones $n \times (k_1 + 1)$ y $n \times (k_2 + 1)$ que contienen las covariables que explican a la media de los conteos Poisson y a la probabilidad del estado perfecto respectivamente, dichas matrices pueden construirse usando la función `model.matrix`.

Para calcular probabilidades con la liga $NS(2.28)$, se requiere instalar previamente el paquete `sn` (Azzalini, 2014) con el comando `install.packages("sn")`.

La función `pAO2` permite crear el vector de probabilidades p_i con la liga $AO2$ (2.14), esta función requiere de dos argumentos: un vector numérico de dimensión n (el predictor lineal) `eta` y el parámetro de forma `tau`; si no se especifica, tomará el valor 1.

```
pAO2 = function(eta, tau = 1)
{
  if ( tau > 0)
  {
    ans1 = 1 - (1 + tau * exp(eta )) ^ (-1/tau)
  } else
  {
    warning("El parámetro tau debería ser positivo. Se usará liga cloglog.")
    ans1 = 1 - exp( - exp(eta))
  }
  ans1
}
```

A continuación se muestra el código para calcular el valor esperado de la variable Z dado Y (3.4). Se requiere que `theta` y `y` sean un vectores numéricos, mientras que `X.Poi` y `X.Cero`

sean matrices diseño que contengan valores numéricos. Si no se especifica la función liga ni los parámetros, se asume una liga *logit*.

```
library(sn)
E_z = function(theta, X.Poi, X.Cero, y, liga = "AO", tau = 1, nu = 0)
{
  if (!((liga = = "AO") | (liga = = "NS")) = = TRUE)
  {
    stop("Debe especificar la liga: 'AO' o 'NS'.")
  }
  k = length(theta)
  k1 = ncol(X.Poi)
  b = theta[1 : k1]
  g = theta[(k1 + 1) : k]
  n = length(y)
  ind = (y = = 0)
  eta = as.vector(X.Cero[ind, ] % *% g)
  if (liga = = "AO")
  {
    p = pAO2(eta = eta, tau = tau)
  }
  if (liga = = "NS")
  {
    p = psn(x = eta, alpha = nu)
  }
  l = exp(as.vector(X.Poi[ind, ] % *% b))
  z = rep(0, n)
  z[ind] = p/(p + (1 - p) * exp(-l))
  return(z)
}
```

El siguiente código se utiliza para crear la función Q (3.7) que se va a maximizar en el algoritmo EM, note que regresa un valor negativo. Requiere del valor esperado que regresa la función `E_Z`

```
Q = function(theta, X.Poi, X.Cero, y, z, liga = "AO", tau = 1, nu = 0)
{
  if (!((liga = = "AO") | (liga = = "NS")) = = TRUE)
  {
    stop("Debe especificar la liga: 'AO' o 'NS'.")
  }
  k = length(theta)
  k1 = ncol(X.Poi)
  b = theta[1:k1]
  g = theta[(k1 + 1):k]
  eta = as.vector(X.Cero % *% g)
  if (liga = = "AO")
  {
    if (tau > 0)
```

```
    {
      p = pAO2(eta, tau)
      q1 = sum(z * log(p))
      q2 = sum((1 - z) * log((1 - p) * exp(y * X.Poi % *% b - exp(X.Poi % *%
    b))))
      qq = -q2 - q1
    } else
    {
      q1 = sum(z * log(1 - exp(-exp(eta))))
      q2 = sum((1 - z) * (y * X.Poi % *% b - exp(eta) - exp(X.Poi % *% b)))
      qq = -q2 - q1
    }
  }
  if (liga = = "NS")
  {
    p = psn(as.vector(X.Cero %*% g), alpha = nu)
    q1 = sum(z * log(p))
    q2 = sum((1 - z) * log((1 - p) * exp(y * X.Poi % *% b - exp(X.Poi % *% b)
    )))
    qq = -q2 - q1
    return(qq)
  }
  return(qq)
}
```

Los errores estándar de los parámetros se pueden calcular con una función auxiliar que evalúe la verosimilitud, pero que no dependa directamente de la función liga seleccionada, por ejemplo (2.22).

```
log_like.aux = function(theta, X.Poi, X.Cero, y, liga = "AO", tau = 1, nu = 0)
{
  if (!((liga = = "AO") | (liga = = "NS")) = = TRUE)
  {
    stop("Debe especificar la liga: 'AO' o 'NS'.")
  }
  k = length(theta)
  k1 = ncol(X.Poi)
  b = theta[1:k1]
  g = theta[(k1 + 1):k]
  index = (y = = 0)
  conte = (y > 0)
  if (liga = = "AO")
  {
    if (tau > 0)
    {
      l1 = sum(log(((1 + tau * exp(X.Cero[index, ] % *% g)) ^ (1/tau)) -  1 +
     exp(-exp(X.Poi[index,] % *%b))))
```

```
18      l2 = sum(y[conte] * (X.Poi[conte, ] % *% b) - exp(X.Poi[conte, ] % *% b
    ))
19      l3 = sum(log((1 + tau * exp(X.Cero % *% g)) ^ (1/tau)))
20      l = l3 - l2 - l1
21    } else
22    {
23      l1 = sum(log(1 - exp(-exp(X.Cero[index,] % *% g)) + exp(-exp(X.Cero[
    index,] % *% g) - exp(X.Poi[index, ] % *% b))))
24      l2 = sum(y[conte] * (X.Poi[conte, ] % *% b) - exp(X.Poi[conte, ] % *% b
    ) - exp(X.Cero[conte, ]% *% g))
25      l = -l1 - l2
26    }
27  }
28  if (liga = = "NS")
29  {
30    mu = exp(X.Poi % *% b)[,1]
31    p = psn(as.vector(X.Cero % *% g),alpha = nu)
32    loglik0 = log( p + exp( log(1 - p) - mu ) )
33    loglik1 = log(1 - p) + dpois(y, lambda = mu, log = TRUE)
34    l = -1*(sum(loglik0[y = = 0]) + sum(loglik1[y > 0]))
35  }
36  return(l)
37 }
```

Para estimar los parámetros de interés por el algoritmo EM, se ejecutan las funciones anteriores y se utiliza la función `optim`. Se puede ayudar al algoritmo usando semillas iniciales con la función `glm.fit`.

```
1 ZIP.EM = function(inicial = NULL, X.Poi, X.Cero, y, liga="AO", tau = 1, nu =
    0, epsilon = 1/1000, max.iter = 100, verbose = FALSE)
2 {
3   if (!((liga = = "AO") | (liga = = "NS")) = = TRUE)
4   {
5     stop("Debe especificar la liga: 'AO' o 'NS'.")
6   }
7   if (is.null(inicial))
8   {
9     t1 = glm.fit(X.Poi, y, family = poisson())
10    if (liga = = "AO")
11    {
12      t2 = glm.fit(X.Cero, as.integer(y = = 0), family = binomial(link = "
    logit"))
13    } else
14    {
15      t2 = glm.fit(X.Cero, as.integer(y = = 0), family = binomial(link = "
    probit"))
16    }
```

```
17    inicial = c(t1$coefficients, t2$coefficients)
18  }
19  semilla = inicial
20  converge = FALSE
21  if (max.iter <= 20)
22  {
23    warning("Número de iteraciones insuficientes", "\n", "Algoritmo EM podría
      no alcanzar un óptimo razonable")
24  }
25  if (epsilon >= 0.01)
26  {
27    warning("Criterio de paro muy débil", "\n", "Algoritmo EM podría no
      alcanzar un óptimo razonable")
28  }
29  distancia = 1
30  paso = 0
31  q0 = E_z(theta = inicial,  X.Poi = X.Poi, X.Cero = X.Cero, y = y,liga = liga
      ,tau = tau, nu = nu)
32  Q0 = Q(theta = inicial, X.Poi = X.Poi, X.Cero = X.Cero, y = y, z = q0, liga
      = liga,tau = tau, nu = nu)
33  t2 <- proc.time()
34  while (paso < max.iter & abs(distancia) > epsilon)
35  {
36    final = optim(par = inicial, fn = Q, X.Poi = X.Poi, X.Cero = X.Cero, y = y
      , z = q0,liga = liga,tau = tau, nu = nu)$par
37    q0 = E_z(theta = final, X.Poi = X.Poi, X.Cero = X.Cero, y = y,liga = liga,
       tau = tau, nu = nu)
38    Q1 = Q(theta = final, X.Poi = X.Poi, X.Cero = X.Cero, y = y, z = q0, liga
      = liga, tau = tau,nu = nu)
39    distancia = abs(Q1 - Q0)
40    paso = paso + 1
41    inicial = final
42    Q0 = Q1
43    if (verbose)
44    {
45      cat("Iteración: ", paso, "\n", "Estimación: ", round(final, 4), "\n", "
      Distancia: ", distancia, "\n")
46    }
47  }
48  nombrep = names(X.Poi); nombrep[1] <- "Intercepto"
49  nombrec = names(X.Cero); nombrec[1] <- "Intercepto"
50  k1 = ncol(X.Poi)
51  k = length(final)
52  b = final[1:k1]
53  g = final[(k1 + 1):k]
54  names(b) <- nombrep; names(g) <- nombrec
55  inip <- semilla[1:k1]; names(inip) <- nombrep
```

```
inic <- semilla[(k1 + 1):k]; names(inic) <- nombrec
mu = exp(X.Poi % *% b)[, 1]
eta = as.vector(X.Cero % *% g)
if (liga = = "AO")
{
  p = pAO2(eta = eta,tau =  tau)
  rval = (1 - p) * mu
  loglik0 = log(p + exp(log(1 - p) - mu))
  loglik1 = log(1 - p) + dpois(y, lambda = mu, log = TRUE)
  loglik = sum(loglik0[y = = 0]) + sum(loglik1[y > 0])
  aic = 2 * length(final) - 2 * loglik
  forma = tau
} else
{
  p = psn(x = eta, alpha = nu)
  rval = (1 - p) * mu
  loglik0 = log(p + exp(log(1 - p) - mu))
  loglik1 = log(1 - p) + dpois(y, lambda = mu, log = TRUE)
  loglik = sum(loglik0[y = = 0]) + sum(loglik1[y > 0])
  aic = 2 * length(final) - 2 * loglik
  forma = nu
}
if (paso < max.iter & abs(distancia) < epsilon)
{
  converge = TRUE
  H = optimHess(par = final, fn = log_like.aux, tau = tau,
  X.Poi = X.Poi, X.Cero = X.Cero, y = y)
  errores = sqrt(diag(solve(H)))
  eeb = errores[1:k1]
  eeg = errores[(k1 + 1):k]
  names(eeb) <- nombrep; names(eeg) <- nombrec
} else
{
  errores = rep(NA, length(final))
  eeb = errores[1:k1]
  eeg = errores[(k1 + 1):k]
}
t1 <- proc.time() - t2
if (converge = = FALSE)
{
  warning("El algoritmo no convergió")
} else
{
  cat("\n","Estimación de parámetros de los conteo: ","\n","Coef: ",round(b
  ,4),"\n","ee:",round(eeb,4),"\n", "Estimación de parámetros de los ceros:
  ","\n","Coef: ",round(g,4),"\n","ee: ",round(eeg,4),"\n","Log verosimilitud
  : ",loglik,"\n")
```

```
100     cat("\n","El algoritmo convergió en ",paso," iteraciones. ","\n","Tiempo
        de ejecución: ",round(as.numeric(t1[3]),2)," segundos.","\n")
101   }
102   salida <- list(Coeficientes = list(Conteos = b,Ceros = g),EE = list(EE.
        Poisson = eeb,  EE.Ceros = eeg),LogVero = loglik,AIC = aic,Y_est = rval,
        Semillas = list(Sem.Conteos = inip, Sem.Ceros = inic),Liga = liga,Forma =
        forma)
103   return(salida)
104 }
```

Note que la función `ZIP.EM` regresa un lista que contiene un indicador de convergencia, la log-verosimilitud calculada, el criterio de información AIC, los conteos estimados, los estimadores de $\boldsymbol{\beta}$, los estimadores de $\boldsymbol{\gamma}$, el parámetro de forma inicial y los errores estándar de los estimadores obtenidos con la función `optimHess`.

Anexo B: Simulación Monte Carlo

En esta sección se presenta el código utilizado para realizar las simulaciones mostradas en el capítulo 4. En primer lugar se muestra una función para generar números aleatorios de la distribución *ZIP*.

Generación de variables aleatorias

Para utilizar esta función, se requiere definir el vector numérico `p` de dimensión `n` cuyos valores estén entre 0 y 1, este valor se obtiene utilizando la función liga *AO2* o *NS* (3.3); también se debe definir el vector numérico `lambda` de dimensión `n` con valores mayores a cero.

```
yes = function(n, p, lambda)
{
  if(any(p > 1) | any(p < 0))
  {
    stop("Las probabilidades deben estar en el intervalo 0 <= p <= 1")
  }
  if(any(p <= 0) )
  {
    stop("El parámetro 'lambda' debe ser positivo")
  }
  u = runif(n)
  ind = u > p
  y = rep(0, n)
  m = sum(ind)
  y[ind] = rpois(n = m, lambda = lambda[ind])
  return(y)
}
```

Esta función devuelve un vector numérico de dimensión n.

Simulación Monte Carlo

Para llevar a cabo los estudios de simulación, se puede definir una función que repita dichas simulaciones. La función `montecarlo` llama a la función `yes` para generar n números aleatorios de la distribución ZIP y estima los parámetros con la función `ZIP.EM`; también requiere especificar la función liga junto con el parámetro de forma y el vector de parámetros de probabilidades; el algoritmo se repetirá según el valor del argumento `r`.

El algoritmo permite volver a simular la distribución ZIP si el existe un error de estimación y manda un mensaje en la consola para advertir del error.

La función devuelve una lista, cuyos elementos son de dimensión r.

```
montecarlo = function(n, r,  p, liga = "AO", tau = 1, nu = 0)
{
  if (!((liga == "AO") | (liga == "NS")) == TRUE)
  {
    stop("Debe especificar la liga: 'AO' o 'NS'.")
  }
  loglike = auxb0 = auxb1 = auxg0 = auxg1 = eeb0 = eeb1 = eeg0 = eeg1 = rep(NA
    , r)
  j = 1
  while (j <= r)
  {
    cat("Replica: ", j, "\n")
    y = yes(n = n, p = p, lambda = ll)
    aux = try(ZIP.EM(X.Poi = B, X.Cero = G, y = y, liga = "AO",tau = tau, nu =
     nu, verbose = TRUE))
    if (class(aux) == "list")
    {
      loglike[j] = aux$LogVero
      auxb0[j] = aux$Coeficientes$Conteos[1]
      auxb1[j] = aux$Coeficientes$Conteos[2]
      auxg0[j] = aux$Coeficientes$Ceros[1]
      auxg1[j] = aux$Coeficientes$Ceros[2]
      eeb0[j] = aux$EE$EE.Poisson[1]
      eeb1[j] = aux$EE$EE.Poisson[2]
      eeg0[j] = aux$EE$EE.Ceros[1]
      eeg1[j] = aux$EE$EE.Ceros[2]
      j = j + 1
    } else
    {
      warning("Algoritmo no convergio", "\n", "Se volvera a simular")
      j = j
    }
  }
  return(list(beta0 = auxb0, beta1 = auxb1, gama0 = auxg0, gama1 = auxg1, logv
     = loglike,eeb0 = eeb0, eeb1 = eeb1, eeg0 = eeg0, eeg1 = eeg1))
}
```

Anexo C: Estimación bayesiana con SAS

En este anexo se presenta los códigos para estimar los parámetros mediante el procedimiento `MCMC`, note que dicho procedimiento está únicamente disponible para la versión 9.2 o posterior. Asimismo requiere utilizar el procedimiento `FCMP`, el cual permite crear funciones compiladas definidas por el usuario y pueden ser llamadas dentro del procedimiento `MCMC`.

En primer lugar, se deben crear la funciones para calcular las probabilidades mediante el procedimiento `FCMP`, dichas funciones se deben guardar en una biblioteca definida previamente y en un catálogo; si no se ha definido ninguna biblioteca, se puede usar la biblioteca temporal `work`.

A continuación se define la función para calcular las probabilidades con la liga Normal Sesgada (2.28), donde el argumento `a` corresponde al parámetro δ:

```
PROC FCMP LIBRARY = work.examples OUTLIB = work.examples.psn;
  FUNCTION psn(x, a);
    pp = 2*probbnrm( x, 0, -a);
    RETURN(pp);
  ENDSUB;
RUN;
```

Para calcular las probabilidades con la liga Aranda-Ordaz asimétrica, se utiliza el siguiente código:

```
PROC FCMP LIBRARY = work.examples OUTLIB = work.examples.pao2;
  FUNCTION pao2(x, a);
    if a > 0 THEN
      pp=1-(1+a*EXP(x))**(-1/a);
    ELSE
      pp=1-EXP(-EXP(x));
    RETURN(pp);
  ENDSUB;
RUN;
```

Se recomienda utilizar variables dummy si en la base de datos existen variables categóricas. Para llamar a las funciones creadas, se debe especificar la biblioteca y el catálogo en donde se guardaron las funciones y así se puede ejecutar el procedimiento `MCMC`.

A continuación se muestra la estimación de parámetros con la liga Aranda-Ordaz asimétrica, se utilizó una distribución *a priori* lognormal para el parámetro τ:

```
OPTIONS CMPLIB = work.examples;
ODS graphics ON;

PROC MCMC OUTPOST = outdatos DATA = datos NMC = 50000 SEED = 1 MCHISTORY =
    detailed STATS = all DIAG = all NTHIN = 10 NBI = 10000 PLOTS = all
```

```
    SIMREPORT = 3;
  PARMS beta0 0 beta1 0 beta2 0 beta3 0 beta4 0 beta5 0 beta6 0 beta7 0;
  PARMS gama0 0 gama1 0 gama2 0 gama3 0 gama4 0 gama5 0;
  PARMS forma 0.2;
  PRIOR beta0 beta1 beta2 beta3 beta4 beta5 beta6 beta7 ~ normal(mean = 0, var
    = 1e6);
  PRIOR gama0 gama1 gama2 gama3 gama4 gama5 ~ normal(mean = 0, var = 1e6);
  PRIOR forma ~ lognormal(-2, sd = 1);
  lambdai=EXP(beta0+beta1*hosp+beta2*health_p+beta3*health_e+beta4*numchron+
    beta5*male+beta6*school+beta7*privins);
  pi=PAO2(gama0 + gama1*hosp+gama2*numchron+gama3*privins+gama4*school+gama5*
    male, forma);
  llike=LOG(pi*(ofp eq 0) + (1-pi)*PDF("poisson", ofp, lambdai));
  MODEL GENERAL(llike);
RUN;
```

Finalmente se presenta el código para estimar los parámetro con la liga Normal sesgada:

```
OPTIONS CMPLIB = work.examples;
ODS graphics ON;

PROC MCMC OUTPOST = outdatos DATA = datos SEED = 1 MCHISTORY = detailed STATS
    = all DIAG = all PLOTS = all SIMREPORT = 3;
  PARMS beta0 0 beta1 0 beta2 0 beta3 0 beta4 0 beta5 0 beta6 0 beta7 0;
  PARMS gama0 0 gama1 0 gama2 0 gama3 0 gama4 0 gama5 0;
  PARMS forma 0;
  PRIOR beta0 beta1 beta2 beta3 beta4 beta5 beta6 beta7 ~ normal(mean=0, var=1
    e6);
  PRIOR gama0 gama1 gama2 gama3 gama4 gama5 ~ normal(mean=0, var=1e6);
  PRIOR forma ~ uniform(-1,1);
  lambdai=EXP(beta0 +beta1*hosp+beta2*health_p+beta3*health_e+beta4*numchron+
    beta5*male+beta6*school+beta7*privins);
  pi=PSN(gama0 + gama1*hosp+gama2*numchron+gama3*privins+gama4*school+gama5*
    male, forma);
  llike=LOG(pi*(ofp eq 0) + (1-pi)*PDF("poisson", ofp, lambdai));
  MODEL GENERAL(llike);
run;
```