Pasar de un modelo tan estructurado a su implementación computacional no resulto fácil. Sin embargo, se logró desarrollar un algoritmo que estima todos los parámetros del modelo de una forma eficiente y que funciona en la práctica. En el fondo, el algoritmo recae en el método de Gibbs sampling propuesto en (**albert1993bayesian**), por lo que se hace una breve introducción a la escuela de inferencia bayesiana, y en el algortimo de backfitting descrito en (**hastie1986generalized**). Al algoritmo se le titula: *bayesian piece wise polinomial model (bpwpm)*. Para facilitar la utilización del modelo en diversas bases de datos, así como su validación y visualización, a la par del algoritmo se desarrolló un paquete de código abierto (con el mismo nombre) para el software estadístico `R`. Al darle un tratamiento bayesiano a los parámetros, más que estimarlos, se busca regresar una muestra de tamaño arbitrario de sus correspondientes distribuciones posteriores. La idea, es que estas distribuciones posteriores, se haya capturado toda la información de los datos de entrenamiento.

Se considera, que una buena forma de entender el algoritmo es *visualizando* tanto los datos como los objetos que componen el modelo, por lo tanto se hace un paréntesis notacional. De las ecuaciones del modelo: (**??**) a (**??**), se tienen dos grupos de parámetros por estimar, $\beta \in \mathbb{R}^{d+1}$ y $w_j \in \mathbb{R}^{N^*} \quad \forall j = 1, \ldots, d$. Donde:

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_d \end{bmatrix} \quad \text{y} \quad w_1 = \begin{bmatrix} w_{1,1} \\ \vdots \\ w_{1,N^*} \end{bmatrix} \quad \ldots \quad w_d = \begin{bmatrix} w_{d,1} \\ \vdots \\ w_{d,N^*} \end{bmatrix}$$

Se hace énfasis en que existen d vectores w_j, cada uno de tamaño N^*. Por lo tanto, se tienen sun total de $1 + d + dN^*$ parámetros. Se usa el símbolo $\mathbf{w}$ para designar todos los vectores w_j, haciendo de este una matriz, es decir: $\mathbf{w} \in \mathbb{R}^{d \times N^*}$. Cuando se habla de datos: $\{(y_i, \mathbf{x}_i)\}_{i=1}^n$, estos se pueden representar en una tabla (o matriz):

$$\left[\begin{array}{c|ccc} y_1 & x_{1,1} & \ldots & x_{1,d} \\ \vdots & \vdots & & \vdots \\ y_n & x_{n,d} & \ldots & x_{n,d} \end{array}\right]$$

Donde el vector de observaciones binarias $\mathbf{y} = (y_1, \ldots, y_n)^t$ es la primer columna de la tabla, y la matriz de covariables $\mathbf{X}$ es el resto.

Bajo esta representación, se da contexto cuando se habla de que la estimación debe reflejar los patrones *hacia abajo* y *hacia lo largo*. Hacia abajo, se está captando la información existente entre las observaciones; cada f_j, mediante su parámetro w_j, representa una transformación no lineal de la variable (o dimensión) j. Hacia lo largo, la función de proyección f suma cada f_j a través de β, ponderando los efectos individuales

de cada variable. Mantener el balance entre la estimación de β a lo largo y w_j hacia abajo, es fundamental para el algoritmo. Analizando este hecho, se concluye que la estimación de ambos grupos de parámetros, se puede ver como una regresión separada para cada uno, y por ende, estos pueden ser estimados por el mismo algoritmo. Esto responde a la dualidad que se exploró en el capítulo pasado de que ambas expresiones son expansiones en bases funcionales. El puente que conecta, y controla el balance entre ambas, son los residuales parciales. Los siguientes capítulos, se concentran en explicar e implementar este curioso patrón.

0.1. Fundamentos de la estadística bayesiana

Dado el problema de describir fenómenos bajo incertidumbre, existen dos escuelas dominantes de la estadística: la frecuentista y la bayesiana. La primera, aunque increíblemente útil, está hasta cierto punto limitada y en ocasiones termina derivando en colecciones de algoritmos. La teoría bayesiana, por el contrario, nombrada así en honor a Thomas Bayes (1702 - 1761), es una rama que enfatiza el componente *probabilista*, dando coherencia al proceso de inferencia (**mendoza2011estadistica**) y (**bernardo2001bayesian**). La estadistica bayesiana está axiomatizada bajo la *teoría de la decisión*. Esta teoría formaliza conceptos económicos como la *coherencia entre preferencias* y *utilidad*, sobre los que desarrolla un marco metodológico para la toma de decisiones.

Esta metodología, además de proveer técnicas concretas para resolver problemas, también formaliza en una forma de pensar sobre la probabilidad como una *medida racional para cuantificar la incertidumbre* condicionando sobre el conocimiento existente. Este paradigma es el que más corresponde con el sentido que usualmente se le da a la palabra. La inferencia sobre creencias (o parámetros), se realiza mediante una *actualización* de estas en luz de nueva evidencia, modificando su medida de incertidumbre. El mecanismo que permite realizar esto, es la aclamada formula de Bayes. De manera informal se puede describir como: dado un evento E bajo condiciones C, la probabilidad *posterior* del evento, es proporcional a la probabilidad *previa* que se tiene sobre este, ponderado por la probabilidad de ocurrencia de las condiciones presentes, es decir:

$$P(E|C)\ \alpha\ P(C|E)P(E) \tag{1}$$

El término central $P(C|E)$ es una medida descriptiva de las condiciones (usualmente datos) llamada *verosimilitud*. Se hace notar que para poder hacer cualquier intento de descripción, se debe especificar el *modelo probablistico* que se asume describe el estado por el que se dan las condiciones C.
En un contexto matemático más formal, la cuantificación de la incertidumbre se da a través de medidas de

probabilidad $\pi(\cdot)$, que describan el fenómeno observado. Estas medidas de probabilidad, usualmente son funciones que dependen de cantidades desconocidas llamados parámetros θ. Aunque desconocidas, se tienen ciertas creencias u conocimiento previo, *a priori*, sobre ellos, descritos por su correspondiente medida de probabilidad $\pi(\theta)$. Además, se tienen datos $\mathbf{X}$, interpretados como *evidencia*, a los cuales se les asigna un modelo de probabilidad dependiente de los parámetros, es decir, su verosimilitud: $\pi(\mathbf{X}|\theta)$. Usando la formula de Bayes, podemos actualizar el conocimiento que se tiene sobre los parámetros haciendo:

$$\pi(\theta|\mathbf{X}) \; \alpha \; \pi(\mathbf{X}|\theta)\pi(\theta) \tag{2}$$

La idea es que este proceso de actualización sea a la vez, un proceso de aprendizaje, en el cual los parámetros capturen la información contenida en los datos.

La teoría frecuentista, adopta un enfoque diferente para el aprendizaje. Se asume que no hay incertidumbre en los parámetros dado los datos y, por lo tanto, estos son tomados como fijos. El mecanismo que permite su estimación, usualmente consiste en plantear una función objetivo y optimizarla. Por ejemplo, si se escoge la verosimilitud $\pi(\mathbf{X}|\theta)$, se busca dar un estimador que la maximice, pues equivaldría a encontrar los parámetros que hagan más *posibles* los datos, bajo el modelo planteado. Si por el contrario, es escoge una función como la RSS de los modelos ANOVA (primer sumando de (**??**)), se busca la θ que minimice estos errores, así, el modelo logra capturar toda la variabilidad que puede sobre los datos. Independientemente del paradigma estadístico que se escoja, siempre es importante la validación del modelo y de sus supuestos. Además, tanto teoría bayesiana como frequentista han resultado de infinita utilidad en la practica y el avance de la estadística y ciencia en general.

Una de las dificultades que surgen en la estadística bayesiana, es que la obtención de resultados analíticos cerrados es difícil o muy tedioso una vez que los modelos se empiezan a complicar. Por ejemplo, en las ecuaciones anteriores, se ha usado el argumento de proporcionalidad α. Esto pues, para que se de la igualdad, el lado derecho de la ecuación (2) se debe de dividir entre $\pi(\mathbf{X}) = \int \pi(X|\tilde{\theta})\pi(\tilde{\theta}) \, d\tilde{\theta}$, el cual usualmente es difícil, sino imposible, de calcular. A este término se le conoce como *constante de proporcionalidad* y su función es la de reescalar la expresión del lado derecho para que en realidad se tenga una distribución en el izquierdo. Usualmente, se escoger distribuciones *conjugadas*, para que tanto la distribución a priori como la posterior sean de la misma familia y por ende conocidas. Sin embargo, con los avances en el poder computacional disponible y técnicas numéricas para resolver integrales (**robert2004monte**), se han desarrollado muchos métodos para aplicar el proceso de aprendizaje, independientemente de que tan complejo sea el modelo o las distribuciones, iniciales y resultantes. Muchos de estos métodos recaen en la

teoría de las *cadenas de Markov*, como lo es, el Gibbs sampler presentado en la sección. 0.2

Estimadores Bayesianos

Una vez realizado el proceso de actualización, el estadista se enfrenta con un problema. Se tiene una distribución posterior de probabilidad para los parámetros de interés, usualmente dada por una muestra y no por una distribución analítica. Sin embargo, por practicidad y utilidad, en ocaciones se busca dar un *estimador puntual*. Por ejemplo, si se necesita dar un estimador $\hat{\theta}$ para usarlo en otros cálculos, o si $\pi(\theta|\mathbf{X})$ es multidimensional. Para superar este problema teórico, se ha adoptado por usar *funciones de perdida* $L(\hat{\theta}, \theta)$[1]. Estas, miden las *consecuencias* que se dan, al tomar $\hat{\theta}$ como el verdadero valor del parámetro θ, es decir, las funciones de perdida evalúan que tan bien se está representando el valor de θ con un estimador puntual. Por ello, vale la pena usar funciones que penalicen la distancia entre θ y $\hat{\theta}$. Sin entrar mucho en los detalles técnicos, se tiene que calcular:

$$\hat{\theta} = \mathbb{E}[L(\hat{\theta}, \theta)] = \int_{\Theta} L(\hat{\theta}, \theta)\pi(\theta)\, d\theta \tag{3}$$

con Θ el espacio de todas las posibles valores de θ. Sin embargo, se demuestra que para funciones de perdida sencillas, pero intuitivas, se tiene que el estimador puntual posterior es alguna medida de centralidad de la distribución posterior. Por ejemplo:

- Función de pérdida cuadrática: $L(\hat{\theta}, \theta) = (\hat{\theta} - \theta)^2$, deriva en la media posterior, es decir: $\hat{\theta} = \mathbb{E}[\theta|\mathbf{X}]$
- Función de perdida valor absoluto: $L(\hat{\theta}, \theta) = |\hat{\theta} - \theta|$, deriva en la mediana de la distribución posterior.
- Función de pérdida 0-1: $L(\hat{\theta}, \theta) = I[\hat{\theta} \neq \theta]$, deriva en la moda de la distribución posterior.

En la práctica, estas cantidades son fáciles de calcular cuando se tiene una muestra simulada de θ proveniente de la distribución posterior. En el paquete, se implementa una forma sencilla de obtener estimadores puntuales con cualquiera de las 3 funciones de pérdida. Sin embargo, se verá que los resultados no varían mucho. Ver Apéndice **??**.

0.1.1. Funciones de probabilidad condicional completas

Retomando el modelo que concierne a este trabajo, se tienen dos grupos de parámetros, β y $\mathbf{w}$. Sin embargo, dados los supuestos del modelo, por el uso de la variable latente z, esta también se debe de incluir como parámetro pues es la liga entre la respuesta y y los datos $\mathbf{X}$, vista de forma bayesiana, también se debe de simular. Por lo tanto, los parámetros quedan: $\theta = (\mathbf{z}, \beta, \mathbf{w})$ con $\mathbf{z} = (z_1, \ldots, z_n)^t$.

1. Formalmente se tiene un problema de decisión.

Esta sección concierne desglosar el proceso de aprendizaje sobre ellos; esta derivación es importante en si pues es la que induce el algoritmo. Usando la notación presentada al inicio de esta sección, los supuestos propuestos en las ecuaciones del modelo (**??**) a (**??**) y sustituyendo en (2) se tiene:

$$\begin{aligned}
\pi(\mathbf{z},\beta,\mathbf{w}|\mathbf{y},\mathbf{X}) &\alpha \quad \pi(\mathbf{y}|\mathbf{X},\mathbf{z},\beta,\mathbf{w})\ \pi(\mathbf{z},\beta,\mathbf{w}) \\
&\alpha \quad \pi(\mathbf{y}|\mathbf{z})\ \pi(\mathbf{z}|\mathbf{X},\beta,\mathbf{w})\ \pi(\beta,\mathbf{w}) \\
&\alpha \quad \pi(\mathbf{y}|\mathbf{z})\ \pi(\mathbf{z}|\mathbf{X},\beta,\mathbf{w})\ \pi(\beta)\ \pi(\mathbf{w}) \\
&\alpha \quad \prod_{i=1}^{n} \mathrm{Be}[y_i|\Phi(z_i))]\ \phi[z_i|f(\mathbf{x}_i),1] \times \pi(\beta)\ \pi(\mathbf{w}) \\
&\alpha \quad \prod_{i=1}^{n} \mathrm{Be}[y_i|\Phi(z_i))]\ \phi[z_i|\beta^t\mathbf{f}(\mathbf{x}_i),1] \times \pi(\beta)\ \pi(\mathbf{w}) \qquad (4)
\end{aligned}$$

donde $\phi(\cdot|\mu,\sigma^2)$ es la función de densidad de una variables aleatoria normal con media μ y varianza σ^2; asimismo $\mathrm{Be}(\cdot|p)$ es función de densidad de una variable Bernoulli con probabilidad de éxito p. Esta factorización es válida dados los supuestos, donde se hace notar, la forma que conecta z_i a las dos partes del modelo a través de la función de proyección $f(\mathbf{x}_i) = \beta^t\mathbf{f}(\mathbf{x}_i)$ que contiene tanto a β como $\mathbf{w}$. Esta derivación es una forma extendida (aunque simplificada) de la verosimilitud para todas las observaciones $i = 1,\ldots,n$. Aunque aún no se han especificado las formas funcionales para las distribuciones a priori $\pi(\mathbf{w})$ y $\pi(\beta)$, estas se pueden separar ya que se asumen independientes. Esta propiedad, combinada con la forma funcional en expansiones de bases, lleva a que se piense en hacer una estimación *por bloques*, es decir, se estima primero β y posteriormente $\mathbf{w}$ en un bucle iterativo, pues esta es la idea de un Gibbs sampler.

0.2. Simulación bayesiana: cadenas de Markov y el Gibbs sampler

Una vez establecida el proceso de actualización, el estadista se ve en la necesidad de tener que desarrollar técnicas para simular de la distribución posterior $\pi(\theta|\mathbf{X})$ sin importar que complejo sea el modelo. Desde principios de los años noventa, se desarrollaron muchos algoritmos y paquetería estadística al aumentar el poder computacional. La gran mayoría de los algoritmos recae en los *métodos Monte Carlo de cadenas de Markov* (MCMC). Estos métodos, como su nombre lo indica, hacen alusión a principios de aleatoriedad, como se daría en un casino. Usando ideas intuitivas de probabilidad y números pseudoaleatorios, se pueden generar muestras prácticamente de cualquier distribución, incluso si su forma funcional es desconocida. La simulación, como tal es un tema que merece un estudio más profundo (**robert2004monte**). Estas poderosas simulaciones, permitieron que los estadistas y experimentadores pudieran hacer el menor número de supuestos posibles sobre los modelos, puesto que ya no se buscan resultados analíticos sino

más bien, se buscaba reflejar la realidad y dejar que los cálculos los hiciera una computadora.

Breve introducción a cadenas de Markov

Al final, la gran mayoría de estos métodos recaen sobre las la teoría de *cadenas de Markov*. Una cadena de Markov, es una secuencia de variables aleatorias: $X^{(1)}$, $X^{(2)}$, ... que cumplen la *propiedad Markoviana*:

$$P(X^{(t+1)}|X^{(t)} = x^{(t)}, X^{(t-1)} = x^{(t-1)}, \ldots, X^{(2)} = x^{(2)}, X^{(1)} = x^{(1)}) = P(X^{(t+1)}|X^{(t)} = x^{(t)}) \quad \forall t$$

con t interpretado como *tiempo*. Por lo tanto, la siguiente variable de la cadena, $X^{(t+1)}$, únicamente depende de *el estado* actual $X^{(t)}$ y no de los anteriores. Usualmente esta propiead es expresada como: el futuro, condicionando al presente, es independiente del pasado. El ejemplo canónico que se presenta es la *caminata aleatoria*: $X^{(t+1)} = X^{(t)} + e^{(t)}$, con $e^{(t)}$ error aleatorio generado de forma independiente. De esta idea se desarrolla toda una rica teoría revisada en cursos de procesos estocásicos (**ross2009introduction**), de donde surgen muchas propiedades aplicables a las cadenas. Una de las ideas más relevantes para lo que concierne este trabajo, es la de *matrices de transición*. Dada una cadena con n posibles estados, es decir, $X^{(t)}$ únicamente puede tomar valores de un subconjunto de cardinalidad n. Se puede construir una matriz cuadrada $P \in \mathbb{R}^{n \times n}$ donde cada entrada $0 \leq p_{i,j} \leq 1$ representa la probabilidad de transicionar del estado i al estado j. Se demuestra, que si una cadena es *ergodica*[2], entonces existe una *distribución límite* que es igual a la *distribución estacionaria*: $\exists\, \pi$ tal que $\pi P = \pi$. Sin entrar en los detalles técnicos, la ergodicidad es la propiedad que asegura que eventualmente se alcanza la convergencia de la cadena sin importar el estado inicial tras repetidas aplicaciones de la matriz de transición P[3]. Esto es, dado un vector de estados inicial $X^{(0)} \in \mathbb{R}^n$ tal que $\mathbf{1}^t X^{(0)} = 1$, se puede encontrar la distribución estacionaria dejando:

$$\pi = \lim_{t \to \infty} X^{(t)} \quad \text{si} \quad X^{(t+1)} = P^t X^{(t)} \tag{5}$$

Esta idea se puede extender a casos más complejos donde se relajan o se cambian supuestos. Incluso, se extiende a casos donde el número de estados es no finito, pero la idea fundamental es la misma.

En el contexto de este trabajo la idea es poder simular *secuencialmente* cadenas de parámetros θ que

2. Aperiódica, irreducible y recurrente positiva. Para efectos de simplicidad en la exposición, la ergodicidad es tratada como una propiedad en si misma. Las definiciones formales, puede ser consultadas en cualquier texto de procesos estocásticos.

3. Esta convergencia es muy diferente a la presentada en el Apéndice **??**. Cuando se cambia de un paradigma frecuentista a uno bayesiano, los teoremas que aseguran la convergencia del modelo son radicalmente diferentes, tanto en forma como en fondo.

estén ligados unos con los otros, que dependan únicamente de el presente y, sobre todo, que una vez simuladas un número arbitrario de estas, converjan a la distribución estacionaria. Precisamente lo que hace un Gibbs sampler.

Gibbs sampler

El Gibbs sampler como tal, es una técnica, para simular variables aleatorias de una *distribución conjunta* sin tener que calcularla directamente, análoga a lo que se vió en más arriba (**gelfand1990sampling**) y (**casella1992explaining**). Usualmente, el muestreo de Gibbs se usa dentro de un contexto bayesiano, aunque también funciona para otras aplicaciones. A primera vista, parece misterioso, pero en realidad, se basa únicamente en las propiedades revisadas (relativamente sencillas) de las cadenas de Markov. Sin perdida de generalidad, se busca simular una muestra de parámetros $\theta = (\theta_1, \ldots, \theta_p)$ que provienen de la distribución conjunta $\pi(\theta|\cdot)$. Esta distribución usualmente no es conocida analíticamente, sin embargo el Gibbs sampler, nos permite dar, no la distribución como tal, pero si una muestra arbitrariamente grande con la que se puede aproximar empíricamente $\hat{\pi}(\theta) \approx \pi(\theta)$. En la practica usualmente más que aproximar la distribución, se busca alguna función de los parámetros como la media o la varianza de la distribución.

Para llevar a cabo el muestreo, se intercambia el difícil cálculo de la distribución conjunta al cálculo de las distribuciones condicionales que usualmente son más fáciles de derivar. Las distribuciones condicionales están dadas por:

$$\begin{aligned} \theta_1 &\sim \pi(\theta_1|\theta_2, \ldots, \theta_p) \\ \theta_2 &\sim \pi(\theta_2|\theta_1, \theta_3, \ldots, \theta_p) \\ &\vdots \\ \theta_p &\sim \pi(\theta_p|\theta_1, \ldots, \theta_{p-1}) \end{aligned} \tag{6}$$

Se comienza con una muestra inicial arbitraria $\theta^{(0)} = (\theta_1^{(0)}, \ldots, \theta_p^{(0)})^t$, donde el superíndice $^{(k)}$ corresponde a la iteración k. Se comienza a simular de las correspondientes distribuciones condicionales, las cuales quedan especificadas para los valores iniciales. En este caso, para $k = 1, 2, 3, \ldots$, se tiene:

$$\begin{aligned} \theta_1^{(k)} &\sim \pi(\theta_1|\theta_2^{(k-1)}, \ldots, \theta_p^{(k-1)}) \\ \theta_2^{(k)} &\sim \pi(\theta_2|\theta_1^{(k)}, \theta_3^{(k-1)}, \ldots, \theta_p^{(k-1)}) \\ &\vdots \\ \theta_p^{(k)} &\sim \pi(\theta_p|\theta_1^{(k)}, \ldots, \theta_{p-1}^{(k)}) \end{aligned} \tag{7}$$

Este proceso se itera hasta tener una muestra de tamaño arbitrario, que haya alcanzado la región de probabilidad donde se encuentra la distribución estacionaria, en este caso la distribución posterior $\pi(\theta|\cdot)$.

La convergencia no es intuitiva, es decir, no es trivial derivar que al muestrear de las distribuciones condicionales, se llegue (eventualmente) a un la distribución conjunta. Sin embargo, la prueba formal, aunque compleja, recae en que se puede formar una matriz de transición con las condicionales de θ_i, análoga a las matrices de las cadenas de Markov. Al dejar que $k \to \infty$, se llega a un resultado equivalente al de la ecuación (5), habiendo muestrado de la distribución posterior. Sin embargo, la ergodicidad, es un supuesto importante que se preserva y es necesario validad. Esto, pues se pueden dar casos donde la distribución posterior no existe.

El tener una muestra de la distribución final $\left\{\theta^{(k)}\right\}_{k=k^*}^{N_{\text{sim}}}$, donde N_{sim} es el número total de simulaciones arbitraria y k^* es el punto a partir del cual se obtiene la convergencia a $\pi(\theta|\cdot)$, tiene muchos beneficios en la practica. Se le pueden calcular momentos a la muestra, medidas de desviación y hacer representaciones gráficas para su análisis y evaluación. En la Figura 1, se tienen dos imágenes que de una muestra Gibbs para una simulación donde: $\theta = \beta \in \mathbb{R}^3$, $N_{\text{sim}} = 1000$ y $k^* = 500$. Para esta figura, se toman los últimos

(a) Trazas (b) Histogramas

Figura 1: Muestro Gibbs para el ejemplo **??**

500 valores de las cadenas. Se observan las *trazas* que se forman al ir simulando los parámetros y se hace un histograma, dando una idea de las distribuciones subyacentes[4].

4. Estas imágenes tienen como propósito ejemplificar el Gibbs sampler. El modelo que se usa es el presentado en este trabajo. Asimismo, en el Capítulo **??** se hará una exploración a fondo de este ejemplo en particular y se verá por que el parámetro β_2 es idénticamente cero, además de la razón por la que los histogramas se ven sospechosamente parecidos a una distribución normal. Asimismo, estas imágenes fueron generadas con la librería *ggplot2*, incorporada a las funcionalidades del paquete *bpwpm* desarrollado para este trabajo.

En la practica, muchos de los pormenores derivados del muestreo Gibbs, pueden ser mejorados. Dado que el valor inicial $\theta^{(0)}$ es dado por el estadista, en ocaciones el método tiene que explorar una región extensa de posibles valores para θ, por lo que podría tardar en converger. Esto deriva en que las primeras observaciones deban ser descartadas pues no son realizaciones de la distribución final buscada. A este periodo se le conoce como *burn-in*. En el ejemplo anterior, se utilizó k^* para designar la iteración a partir de cual se toman los valores simulados. En la practica, usualmente se guarda toda la cadena, se explora, tanto por resúmenes numéricos como con representaciones gráficas y se decide (de forma subjetiva) el corte k^*. Otro método ampliamente usado es el de adelgazamiento o *thinning*. Por las mismas características de la estimación por Gibbs, sobre todo en casos multivariados, los valores de las cadenas tienen correlaciones altas. Si se quieren muestras independientes, la bibliografía recomienda tomar cada k_{thin}-ésimo valor de la cadena generada para reducir la dependencia entre los parámetros. Usualmente se usan valores pequeños para k_{thin}. Estos sencillos pasos pasos para mejorar las cadenas, ya se encuentran implementados en el paquete *bpwpm* para `R` para el análisis rápido de las cadenas.

0.2.1. Algoritmo de Albert y Chibb

El algoritmo particular del Gibbs sampler que se usa en este trabajo, es una versión modificada del presentado en (**albert1993bayesian**). Este método ofrece varias ventajas para el modelo propuesto, pues se desarrolló específicamente para regresiones probit usando la variables latente z. Además es un método muy eficiente pues usa distribuciones conjugadas, por lo que las distribuciones condicionales, se pueden calcular directamente y la parte estocástica depende únicamente de simular distribuciones conocidas. Esto lleva a que los periodos de burn-in sean relativamente pequeños y que el adelgazamiento no sea fundamentalmente necesario.

A su algoritmo, ellos lo llaman *Data Augmentation for Binary Data* y son los pioneros en el uso de variables latentes para unir la respuesta y con las covariables $\mathbf{x}$ como se revisa en la Sección **??**. En su exposición, ellos utlizan una función de proyección lineal $f(\mathbf{x}) = \beta^t\mathbf{x}$, diferente. Por lo mismo, (y por un breve momento) esta se utiliza para explicar la idea fundamental. Usando la misma notación, se introducen n variables latentes $\mathbf{z} = (z_1, \ldots, z_n)^t$, con $z_i \sim N(\beta^t\mathbf{x}_i, 1)$, sobre las que se definien se redefinen las respuestas:

$$y_i = \begin{cases} 1, & \text{si } z_i > 0 \\ 0, & \text{si } z_i \leq 0 \end{cases} \tag{8}$$

La simplificación del modelo, obliga a $\theta = (\mathbf{z}, \beta)$. Por lo tanto, la derivación bayesiana queda:

$$\begin{aligned}
\pi(\mathbf{z}, \beta|\mathbf{y}, \mathbf{X}) &\alpha \quad \pi(\mathbf{y}|\mathbf{X}, \mathbf{z}, \beta)\ \pi(\mathbf{z}, \beta) \\
&\alpha \quad \pi(\mathbf{y}|\mathbf{z})\ \pi(\mathbf{z}|\beta, \mathbf{X})\ \pi(\beta) \\
&\alpha \quad \prod_{i=1}^{n}[I(y_i = 1)I(z_i > 0) + I(y_i = 0)I(z_i \leq 0)] \times \phi(z_i|\beta^t \mathbf{x}_i, 1) \times \pi(\beta) \qquad (9)
\end{aligned}$$

Ahora, bajo los fundamentos del Gibbs Sampler, en lugar de querer encontrar la distribución posterior (9), se busca encontrar las distribuciones condicionales. Para β:

$$\begin{aligned}
\pi(\beta|\mathbf{z}, \mathbf{y}, \mathbf{X}) &= \frac{\pi(\mathbf{z}, \beta|\mathbf{y}, \mathbf{X})}{\pi(\mathbf{z})} \qquad (10) \\
&= \frac{\pi(\mathbf{y}|\mathbf{z})\ \pi(\mathbf{z}|\beta, \mathbf{X})\ \pi(\beta)}{\pi(\mathbf{y}, \mathbf{X})\ \pi(\mathbf{z})} \\
&= \frac{\pi(\mathbf{y}|\mathbf{z})}{\pi(\mathbf{y}, \mathbf{X})\ \pi(\mathbf{z})} \overset{C}{\nearrow} \times \pi(\mathbf{z}|\beta, \mathbf{X})\ \pi(\beta) \qquad (11) \\
&= C\, \pi(\beta) \prod_{i=1}^{n} \phi(z_i|\beta^t \mathbf{x}_i, 1) \qquad (12)
\end{aligned}$$

la densidad condicional es la misma que se derivaría si se tuviera una regresión lineal bayesiana con z de regresor, es decir: $z_i = \beta^t \mathbf{x}_i + e_i$ con $e_i \sim N(0, 1)$. Esta es la utlidad de la variable latente, que convierte una regresión probit a una regresión lineal ttadicional. Asimismo, para estos modelos, dependiendo de la distribucion *a priori* $\pi(\beta)$ se pueden conseguir resultados cerrados. Se hace notar que la ecuación (10) se toma de la definición de probabilidad condicional, y el paso de (11) a (12) se puede hacer ya que, al definir y como en la ecuación (8), sus representaciones son análogas y el cociente se desvanece, dejando únicamente la constante C que sale del término $\pi(\mathbf{y}, \mathbf{X})$. Ahora, únicamente falta definir $\pi(\beta)$. Es común en la práctica usar distribuciones *no informativas* sobre los parámetros, cuando no se tiene experiencia sobre ellos. Sin embargo, para el modelo lineal bayesiano, existe una familia de distribuciones conjugadas (**banerjee2008gory**), que son razonables para la aplicacion que se buscan. En particular, al dejar la varianza fija y eligiendo de distribución $\pi(\beta)$:

$$\beta \sim N_{d+1}(\beta|\mu_\beta, \Sigma_\beta) \qquad (13)$$

donde $\mu_\beta \in \mathbb{R}^{d+1}$ es el hiperparámetro de media y $\Sigma_\beta \in \mathbb{R}^{(d+1)^2}$ la matriz de covarianza, entonces, se

tiene la distribución conjugada:

$$\beta|\mathbf{y},\mathbf{z},\mathbf{X} \sim N_{d+1}(\beta|\mu^*_\beta,\Sigma^*_\beta) \tag{14}$$

donde:

$$\mu^*_\beta = \Sigma^*_\beta \times (\Sigma^{-1}_\beta \mu_\beta + \mathbf{X}^t\mathbf{z})$$
$$\Sigma^*_\beta = (\Sigma^{-1}_\beta + \mathbf{X}^t\mathbf{X})^{-1}$$

Esta distribución es conjugada pues preserva la estructura normal de β. Asimismo, es fácil de simular usando cualquier software estadístico, calculando previamente todos los parámetros y dando un valor (o iteración) para $\mathbf{z}$[5].

Ahora, condicionar sobre $\mathbf{z}$, es más sencillo, pues la derivación es similar, comenzando con la expresión (9) y reordenando términos:

$$\begin{aligned}
\pi(\mathbf{z}|\beta,\mathbf{y},\mathbf{X}) &= \frac{\pi(\mathbf{z},\beta|\mathbf{y},\mathbf{X})}{\pi(\beta)} \\
&= \frac{\pi(\mathbf{y}|\mathbf{z})\ \pi(\mathbf{z}|\beta,\mathbf{X})\ \pi(\beta)}{\pi(\mathbf{y},\mathbf{X})\ \pi(\beta)} \\
&= \cancel{\frac{1}{\pi(\mathbf{y},\mathbf{X})}}^{C} \ \pi(\mathbf{y}|\mathbf{z}) \times \pi(\mathbf{z}|\beta,\mathbf{X}) \\
&= C \prod_{i=1}^{n} [I(y_i=1)I(z_i>0) + I(y_i=0)I(z_i\leq 0)] \times \phi(z_i|\beta^t\mathbf{x}_i,1)
\end{aligned} \tag{15}$$

De donde es claro ver que las z_i son independientes y tienen distribuciones normales truncadas en 0:

$$\begin{aligned}
z_i|y_i=1,\beta &\sim N(z_i|\beta^t\mathbf{x}_i,1) \text{ truncada a la izquierda} \\
z_i|y_i=0,\beta &\sim N(z_i|\beta^t\mathbf{x}_i,1) \text{ truncada a la derecha}
\end{aligned} \tag{16}$$

las cuales también son fáciles de simular usando los algortimos de (**devroye1986non**).

Finalmente, una vez que se tienen las distribuciones condicionales (16) y (14), la simulación de esto dos primeros grupos de parámetros se realiza con un muestreo de Gibbs dado por las ecuaciones (7) de la página 7. En forma de pseudocódigo:

El valor de $\mathbf{z}^{(0)}$ en realidad no se tiene que dar pues se simula dependiendo de y y $\beta^{(0)}$. Este valor inicial

5. Se hace notar, que este estimador, es relativamente parecido al estimador que se da en una regresión cordillera.

```
SampleoGibbs(y, X, N_sim, beta(0), mu_beta, sigma_beta):
    sigma_beta <- ((sigma_beta)^-1 + X'X)^-1
    PARA k = 1 HASTA N_sim:
        z(k) <- SimNormTrunc(y, X, beta(k-1))
        mu_beta(k) <- sigma_beta*((sigma_beta)^-1*mu_beta + X'*z(k))
        beta(k) <- NormMulti(mu_beta(k), sigma_beta(k))
    REGRESAR beta
```

Tabla 1: Algoritmo de Albert y Chibb para modelos probit

$\beta^{(0)}$ es arbitario, pero se sugiere que sea dado por el estimador de máxima verosimilitud o el de mínimos cuadrados para las respuestas binarias $\beta^{(0)} = (\mathbf{X}^t\mathbf{X})^{-1}\mathbf{X}^t y$. Sin embargo, en la práctica el algoritmo, por default, los inicializa en ceros y tardan poco tiempo en converger a la distribución límite.

Como comentario, el trabajo de (**albert1993bayesian**), incluye muchos más métodos Gibbs sampler para diferentes modelos. Entre ellos, multinomiales o modelos tobit, los cuales usan como función liga g a la función de acumulación *t-student* en vez de la distribución normal.

0.3. Algoritmo *bpwpm*

Finalmente, se esta en posición para hablar de la estimación de los parámetros $\mathbf{w}$ en el corazón del modelo. Dado que este modelo se construyó desde cero, no se encontró un algoritmo ya establecido. Sin embargo, usando las intuitivas ideas de los modelos GAM, se implemento un método para simular $\mathbf{w}$ que funciona muy bien en la practica. Una vez establecido eso, se juntan todos los pedazos y se arma el algoritmo final.

0.3.1. Algoritmo de *backfitting* para ajuste de modelos GAM

El *algoritmo de backfitting* fue desarollado para hacer la estimación (frequentista) de las funciones f_j que componen los modelos GAM revisados en la sección **??** (**hastie1986generalized**). Como se mencionó, la idea central recae en que, cada dimensión $j = 1, \ldots, d$, más un término independiente f_0, capture de forma aditiva, toda la información posible del regresor, en este caso $z \in \mathbb{R}$, a través de transformaciones no lineales f_j.

$$z = f_0 + f_1(x_1) + \ldots + f_d(x_d) + e \tag{17}$$

Estas f_j son muy flexibles y se demuestra (bajo ciertos supuestos) que son splines cúbicos. En la práctica, usualmente se dejan indeterminados y son estimadas mediante procedimientos no paramétricos. Usar estas funciones en vez de un proyector lineal está justificado si $\mathbb{E}(z|\mathbf{x})$ no es lineal, lo cual se da muchas veces en la práctica. Sin embargo, la estimación de f_j es diferente a la estimación tradicional de parámetros β pues se enfatiza, que son *arbitrarias* en su escala. Por ejemplo, sea $d = 1$ y $z_i = 1$ para alguna i. Se podría tener $z_i = \hat{f}_0 + \hat{f}_1(x_1)$ con las estimaciones $\hat{f}_0 = 3$ y $\hat{f}_1(x_1) = -2$ ó $\hat{f}_0 = -1$ y $\hat{f}_1(x_1) = 2$, sin importar el valor de x_1 y los dos serían modelos que (al menos para esta observación i) ajustarían perfectamente. Para controlar estas posibles fluctuaciones, el modelo trata de capturar la información de forma secuencial a través de los *residuales parciales*. Es decir, una vez dadas, todas las f_j menos una, se evalúa que tanto le falta al modelo por capturar y posteriormente, se hace un suavizamiento de estos usando una función genérica $S_j(\cdot|x_j)$ que depende de los datos. Más formalmente, los residuales parciales, reformulando la ecuación **??** con la notación de esta sección son, $\forall j^* = 1, \ldots, d$:

$$\hat{r}_{j^*} = z - f_0 - \sum_{\substack{j=1 \\ j \neq j^*}}^{d} f_j(x_j) = z - f_0 - \ldots - f_{j^*-1}(x_{j^*-1}) - f_{j^*+1}(x_{j^*+1}) - \ldots - f_d(x_d) \tag{18}$$

posteriormente, se da un estimado de f_{j^*}:

$$\hat{f}_{j^*}(x_{j^*}) = S(\hat{r}_{j^*}|x_{j^*}) = S[z - f_0 - \sum_{\substack{j=1 \\ j \neq j^*}}^{d} f_j(x_j)|x_{j^*}] \tag{19}$$

Por lo tanto, $\hat{f}_j^*(x_{j^*})$ es un suavizamiento, de esa dimensión. De forma iterativa, una vez dada la estimación para la dimensión j^*, se puede mejorar, a su vez, la estimación de $f_{j^*\pm1}$, dando paso a un algoritmo secuencial hasta que no haya mucha variación en las f_j. Un paso importante que se debe hacer es *identificar* el modelo, esto es, centrar los residuales en 0 tomando el término independiente, como el promedio de las respuestas, es decir: $f_0 = \bar{z}$, de tal forma que todas las f_j queden alrededor de cero.

El algoritmo de backfitting en pseudocódigo y usando una función de suavizamiento arbitraria es:

Aplicando los métodos GAM al modelo

Este principio de suavizamiento iterativo de los residuales parciales, es justamente lo que se buscaba para poder finalizar el algoritmo. Recordando la función de projección (**??**) del modelo, una vez estimada z, se

```
AlgoritmoBackfitting(z, X, f_init):
    f(0) <- Promedio(z)
    f(j) <- f_init(j) para toda j = 1,...,d
    MIENTRAS convergencia:
        PARA j = 1,...,d,1,...,d,...:
            f(j) <- Suavizar(z - f(0) - suma[f(k), para toda k != j], x(j))
```

Tabla 2: Algoritmo de backfitting para modelos GAM

tiene una representación análoga a un GAM.

$$\hat{z} = \beta_0 + \beta_1 f_1(x_1) + \ldots + \beta_d f_d(x_d) + e = \beta^t \mathbf{f}(\mathbf{x}) + e$$

con la adición de las β_j. De donde se pueden obtener, usando la misma idea los residuales parciales, $\forall j^* = 1, \ldots, d$:

$$r_{j^*} = 1/\beta_{j^*}[\hat{z} - \beta_0 - \sum_{\substack{j=1 \\ j \neq j^*}}^{d} \beta_j f_j(x_j)]$$

Estos residuales, calculándose para toda observación $i = 1, \ldots, n$, son vectores. El paso fundamental, para poder hacer la estimación de $\mathbf{w}$ recae sobre el suavizamiento. En la sección anterior, se denotó por la función arbitraria $S_j(\cdot|x_j)$ sin embargo, para este modelo se tiene una representación funcional concreta para S_j, los polinomios por partes flexibles dados por la ecuación (**??**) y (**??**) de donde se hace la igualdad:

$$r_{j^*} = \sum_{l=1}^{N^*} w_{j^*,l}\ \Psi_l(x_{j^*}, \mathcal{P}_{j^*}) = w_{j^*}^t \Psi(x_{j^*}, \mathcal{P}_{j^*}) \tag{20}$$

Lo cual, no es más que un modelo de regresión lineal *hacia abajo*[6], usando a w_{j^*} como coeficientes y a las transformaciones $\Psi(x_{j^*}, \mathcal{P}_{j^*})$ como covariables. Este hecho, se debe a la *dualidad* en expansiones en bases funcionales, de las que tanto se habló en el capítulo pasado. Existe una correspondencia uno a uno, entre β y w_j, así como entre $\mathbf{f}$ y Ψ. Por lo tanto, usando las técnicas que se discutieron en la Sección 0.2, las w_j pueden ser estimadas exactamente de misma forma que β.

Aunque intricado y notacionalmente pesado, el modelo en si, es una serie de regresiones lineales, una para β y d para cada w_j, ligadas a una respuesta binaria a través de la z. Con esta simplificación, se espera que la Figura **??** de la página **??**, haya adquirido una nueva luz. En el siguiente capítulo, se busca esclarecer

6. y no *a lo largo*, como sería para β

con ejemplos toda esta maraña de notación y se probará su efectividad.

0.3.2. Implementacion Final

Una vez conectados todos componentes del modelo, este se puede presentar en su versión final y más completa, aunque definitivamente, menos clara. Se recuerda que existe un compendio notacional en el Apéndice **??**. Juntando las expresiones (**??**) a (**??**), (**??**), (**??**), (13) y agregando las distribuciones a priori para w_j: $\pi(w_j)$; se tiene el *modelo bpwpm*:

$$
\begin{aligned}
y \,|\, z \; &\sim \mathrm{Be}(y \,|\Phi(z)) \\
z \,|\, x \; &\sim \mathrm{N}(z \,|f(\mathbf{x}), 1) \\
f(\mathbf{x}) &\approx \sum_{j=0}^{d} \beta_j f_j(x_j) = \beta^t \mathbf{f}(\mathbf{x}) \\
f_j(x_j) &\approx \sum_{l=1}^{N^*} w_{j,l} \Psi_{j,l}(x_j, \mathcal{P}_j) = w_j^t \Psi(x_j, \mathcal{P}_j) \qquad \forall j = 0, 1, \ldots, d \\
&= \sum_{i=1}^{M} w_{j,i,0}\; x_j^{i-1} + \sum_{i=K}^{M-1} \sum_{k=1}^{J-1} w_{j,i,k}\; (x_j - \tau_k)_+^i \\
\beta &\sim N_{d+1}(\beta|\mu_\beta, \Sigma_\beta) \\
w_j &\sim N_{N^*}(w_j|\mu_{w_j}, \Sigma_{w_j}) \qquad \forall j = 0, 1, \ldots, d
\end{aligned}
\tag{21}
$$

A diferencia de su exposición en el capítulo **??**, el algoritmo final, debe de construir de abajo hacia arriba, pues se necesita tener una estimación puntual de los parámetros para poder calcular las funciones intermedias y que todo quede definido de forma numérica.

Como paso final del algoritmo, se deben definir las particiones $\mathcal{P}_j \quad \forall j = 1, \ldots, d$. Estas particiones, quedan determinadas al estimar las posiciones los $J - 1$ nodos τ_j. De forma intuitiva, se necesitarán más nodos donde se tengan más datos pues son las regiones donde se podrían dar un mayor número de saltos. Por lo tanto, el algoritmo calcula automáticamente las posiciones usando los cuantiles correspondientes para cada dimensión.[7].

Dado que el modelo tiene muchos componentes y pasos intermedios, la Figura 2, hace un resumen gráfico del algoritmo. Una vez más, el supeíndice $^{(k)}$ denota la iteración. $\Psi_N^*(\mathbf{X}, \mathcal{P})$ denota la expansión en

7. Se calculan los cuartiles con saltos de probabilidad $1/J$, donde J es el número de intervalos

bases truncadas para los datos $\mathbf{X}$ y la partición $\mathcal{P}$ (de todas las dimensiones)[8] dependiendo del los tres parámetros M, J y K (resumidos en N^*). Finalmente, $F \in \mathbb{R}^{n\times d}$ denota las matriz de transformaciones no lineales, es decir: $F = [f_1(x_1), \ldots, f_d(x_d)]$ donde cada $f_j(x_j)$ es un vector para todas las observaciones. El ciclo de adentro, se da pues al ir calculando y actualizando cada dimensión, j, cambia la transformación f_j y se deben de actualizar los residuales parciales. Este proceso se repite d veces y una vez calculados todos los vectores w_j, se regresa al bucle principal donde se simula$z$ y β una vez más.

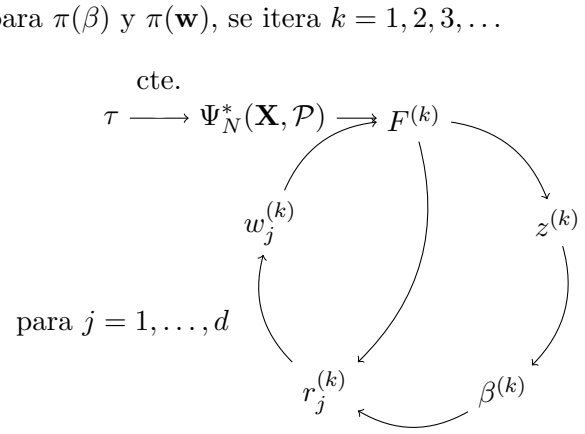

Figura 2: Esquema del algoritmo

Dado que los datos y los nodos son fijos, la expansión en bases de polinomios truncados únicamente se tiene que calcular una vez y es constante. Si se usara el enfoque de (**mallik1998automatic**), esta parte aislada del diagrama, debería de ser incluida en el ciclo.

En forma de pseudocódigo el algoritmo final se encuentra en la Tabla 3 de la página 17. Sin embargo, inclusive en esta forma, el modelo sigue sufriendo de una notación pesada debido a la gran cantidad de índices y variables[9], sin embargo, la implementación del algoritmo final, se separó en una serie de funciones relativamente sencillas y vectorizadas que hacen de el manejo de los objetos una tarea más sencilla. El código que se desarrolló es de dominio publico disponible en `https://github.com/PaoloLuciano/BPWPM`, si se quiere hacer una exploración más profunda. Asimismo, se desarrolló mucha funcionalidad adicional para visualizar e imprimir información útil de los posibles modelos que se puedan hacer. En el Apéndice **??**, se hace un compendio de las funciones y una breve descripción de lo que hacen. La documentación completase encuentra también en el paquete.

8. El objeto Ψ en realidad es un arreglo 3-dimensional pues cada $x_{i,j}$ tiene una expansión de tamaño N^*. Su implementación, se basa en el diagrama **??** de la página **??**

9. Como dato curioso, si se expusieran todas las ecuaciones para cada dato real $x_{i,j}$ y la expansión en bases completa, en vez de su representación vectorial, se tendrían que usar 5 índices.

```
bpwpm(y, X, M, J, K, N_sim, beta(0), mu_beta(0), sigma_beta(0), w(0), mu_w(0), sigma_w(0)):
    tau <- Cuantiles(X, 1/J)
    Phi <- CalculaPhi(X, M, J, K, tau)
    F(0) <- CalculaF(Phi, w(0))

    PARA k = 1 HASTA Nsim:
        z(k) <- SimNormTrunc(y, F, beta(k-1))

        sigma_beta(k) <- ((sigma_beta(0))^-1 + F(k-1)'F(k-1))^-1
        mu_beta(k) <- sigma_beta(k)*((sigma_beta(k))^-1*mu_beta(k-1) + F'*z(k))
        beta(k) <- NormMulti(mu_beta(k), sigma_beta(k))

        PARA j = 1 HASTA d
            r(j) <- ResidualesParciales(z, beta, F(k))

            sigma_w(j,k) <- (sigma_w(j,k-1)^-1 + Phi'Phi)^-1
            mu_w(j,k) <- sigma_w(j,k)*((sigma_w(j,k))^-1*mu_w(j,k-1) + Phi'r(j))
            w(j,k) <- NormMulti(mu_w(j,k), sigma_w(j,k))

            F <- ActualizarDimensionjDeMatrizF(Phi, w(j,k))

    REGRESAR beta, w
```

Tabla 3: Algoritmo *bayesian piecewise polinomial model*