UNIVERSIDAD DE ALMERÍA

FACULTAD DE

CIENCIAS EXPERIMENTALES

Departamento de Estadística
y Matemática Aplicada

MODELOS DE REDES BAYESIANAS CON VARIABLES DISCRETAS Y CONTINUAS

TESIS DOCTORAL

Rafael Rumí Rodríguez

Almería, Junio 2003

MODELOS DE REDES BAYESIANAS CON VARIABLES DISCRETAS Y CONTINUAS

TESIS DOCTORAL

Rafael Rumí Rodríguez

DIRECTORES:

Serafín Moral Callejón

Antonio Salmerón Cerdán

Junio 2003

Agradecimientos

Quisiera dar las gracias desde aquí a todos aquellos que me han prestado su apoyo, bien sea intelectual, o emocional y que con ello han contribuido de forma fundamental a la elaboración de esta memoria.

No resulta sencillo poner por escrito en estas letras todos los sentimientos que a lo largo de la realización de esta tesis me invadieron, por ello no quiero dejarme a nadie en el tintero en estos agradecimientos, pero si así ocurriese, espero me supiese perdonar.

Quiero y debo empezar agradeciendo a mis directores, los doctores Antonio Salmerón Cerdán y Serafín Moral Callejón su predisposición y colaboración total, no sólo a la hora de realizar esta memoria, así como su apoyo, que me brindaron en todo momento, y paciencia, especialmente al comenzar esta ardua tarea.

No puedo olvidarme de mis compañeros del Departamento de Estadística y Matemática Aplicada de la Universidad de Almería, y de los miembros del Grupo de Análisis de Datos de la Universidad de Almería, que me han hecho sentir parte de una familia desde el momento que comencé mi labor en esta Universidad , y cuyas observaciones y sugerencias relacionadas con esta memoria fueron siempre tan acertadas. Para todos ellos mi agradecimiento, en especial a Fernando Reche, por hacer tan llevadera y agradable la convivencia diaria.

También he de agradecer a Ministerio de Ciencia y Tecnología la financiación parcial de esta tesis a través del proyecto TIC-1135-C04-02.

Por último, dedicar un agradecimiento sincero a mi familia, fundamentalmente a mis padres y hermanos, por estar siempre a mi lado, sobre todo en los malos momentos, a todos mis amigos, y especialmente a mi novia Mercedes, por convertir los malos momentos en buenos momentos y ayudarme superar las dificultades del camino. Sin ella no lo habría logrado.

Índice general

Introducción

Desde principios del siglo pasado, el uso de modelos gráficos (Lauritzen 1996) para representar distintos tipos de sistemas es conocido en distintas disciplinas científicas como la Física Estadística, en el estudio de sistemas de partículas donde cada una de ellas interactúa con las que están situadas en su proximidad. Para representar de forma sencilla las relaciones de vecindad entre partículas, se utilizaron grafos no dirigidos. A partir de ahí, el uso de estructuras gráficas para representar problemas científicos se mostró como un medio para llegar a una mejor comprensión y resolución de los mismos. Surgieron entonces trabajos que aplicaban ideas similares en distintas disciplinas, como la genética, con la representación de la herencia de cualidades entre individuos mediante grafos dirigidos. En general, podría decirse que los *modelos gráficos* son formas de representar interacciones entre entidades de un modelo mediante grafos, siendo, por tanto, una herramienta universal ampliamente utilizada.

Estos modelos han sido profundamente estudiados en Estadística desde principios de la década de los 80 del pasado siglo, principalmente como representaciones de modelos probabilísticos factorizables o descomponibles (Darroch, Lauritzen, y Speed 1980; Lauritzen, Speed, y Vijayan 1984; Lauritzen 1996). Pero también dentro del campo del análisis de datos los modelos gráficos son una herramienta que cobra cada vez más importancia desde la publicación del trabajo de Whittaker en 1990.

El empleo de los modelos gráficos para el análisis de datos en lugar de otros métodos multivariantes ha abierto nuevas perspectivas en este campo, entre las que podemos citar las siguientes:

- Análisis de grandes volúmenes de datos con numerosas variables.

- Análisis de datos con semántica fácilmente interpretable para variables cualitativas, en base al concepto de separación direccional o d-separación.
- Posibilidad de analizar modelos en los que aparezcan variables discretas y continuas simultáneamente.
- Posibilidad de incorporar conocimiento cualitativo y cuantitativo sobre el modelo por un experto en el área.

Pero quizás el mayor desarrollo de estas técnicas se ha alcanzado a raíz de su aplicación en el campo de la Inteligencia Artificial, como herramientas para la representación y manipulación del conocimiento en sistemas expertos capaces de emular el comportamiento de un experto humano en un dominio concreto. El uso de la Teoría de la Probabilidad para manejar la incertidumbre en el conocimiento dio origen a los llamados *sistemas expertos probabilísticos*, ampliamente estudiados en la literatura (Castillo, Gutiérrez, y Hadi 1996; Cowell, Dawid, Lauritzen, y Spiegelhalter 1999; Jensen 1996; Jensen 2001; Pearl 1988; Shafer 1996).

Un modelo gráfico, tanto si es el resultado de analizar un conjunto de datos como si es la base de conocimiento de un sistema experto, puede ser utilizado para trazar conclusiones sobre las variables del sistema que representa ante la llegada de nueva información. Los primeros trabajos en este sentido se deben a Kim y Pearl (1983) y Pearl (1986a, 1986b, 1988), que desarrollaron métodos de inferencia en modelos gráficos probabilísticos donde el grafo asociado es un árbol o un poliárbol. Posteriormente, Shafer y Shenoy (1988) y Lauritzen y Spiegelhalter (1988) presentan esquemas de inferencia (propagación) para grafos dirigidos acíclicos en general, basándose en una estructura auxiliar llamada *árbol de cliques*. Estos esquemas fueron mejorados posteriormente por Jensen et al. (Jensen, Olesen, y Andersen 1990; Jensen, Lauritzen, y Olesen 1990) con el esquema implementado en el sistema HUGIN (Andersen, Olesen, Jensen, y Jensen 1989), y más recientemente por Shenoy (1997), Schmidt y Shenoy (1997) y Madsen y Jensen (1999).

Todos los esquemas de inferencia citados anteriormente son *exactos*. Sin embargo, el problema de la propagación exacta es NP-duro (Cooper 1990), por lo que en la práctica,

los métodos exactos no serán aplicables si el tamaño del problema es suficientemente grande. Aunque el problema de la propagación mediante métodos aproximados es también NP-duro (Dagum y Luby 1993), la clase de problemas que se pueden tratar es más amplia.

La mayor parte de los métodos aproximados se basan en técnicas de simulación por Monte Carlo para obtener una estimación de la distribución *a posteriori* (Bouckaert, Castillo, y Gutiérrez 1996; Cano, Hernández, y Moral 1996; Fung y Chang 1990; Henrion 1988; Jensen, Kong, y Kjærulff 1995; Pearl 1987; Salmerón, Cano, y Moral 2000; Salmerón y Moral 2001; Shachter y Peot 1990), o bien en simplificaciones deterministas del problema (Cano y Moral 1997; Cano, Moral, y Salmerón 2000; Cano, Moral, y Salmerón 2002; Cano, Moral, y Salmerón 2003; Kjærulff 1993; Kjærulff 1994; Jensen y Andersen 1990b).

En el caso de aplicaciones prácticas en el ámbito del análisis de datos, es frecuente enfrentarse a problemas en los que aparecen simultáneamente variables discretas y continuas. En este caso, el uso de los métodos citados anteriormente no es, en general, posible de una forma directa. El problema se ha resuelto para casos particulares en los que las variables siguen un modelo probabilístico concreto, como es la distribución condicional gaussiana (Lauritzen y Wermuth 1989; Lauritzen 1992; Olesen 1993), en la que la marginal sobre las variables discretas es multinomial y sobre las continuas condicionada a las variables discretas es normal multivariante. También se han desarrollado aplicaciones con otros modelos, como el uso de redes con variables beta para determinar la resistencia de estructuras de hormigón (Castillo y Gutiérrez 1998). Cuando las variables no se ajustan a los modelos anteriores, la solución más general consiste en discretizar las variables continuas, adoptando a partir de entonces un tratamiento similar a las discretas (Dougherty, Kohavi, y Sahami 1995; Christofides, Tanyi, Whobrey, y Christofides 1999; Kozlov y Koller 1997).

Recientemente han aparecido algunos trabajos en otros modelos (Davies 2002; Davies y Moore 2002).

Objetivos de la memoria

El objetivo de esta memoria se centra en el desarrollo de métodos para la construcción (aprendizaje) de modelos gráficos, en concreto redes bayesianas, para el tratamiento de problemas en los que aparecen simultáneamente variables discretas y continuas, así como algoritmos eficientes para realizar inferencia sobre las mismas. Para alcanzar este objetivo hemos establecido las siguientes etapas:

1. *Estudio de las redes bayesianas discretas.* Abarca la revisión de los modelos especificados mediante una red bayesiana con variables discretas, así como los métodos principales para realizar inferencias sobre ellas. Como herramientas para el análisis de datos, este estudio incluirá el proceso de obtención de las redes a partir de muestras.

2. *Estudio de los métodos para el tratamiento de variables continuas.* Se recopilarán los métodos más relevantes de tratamiento de problemas con variables discretas y continuas que se encuentran en la literatura, centrándonos en la discretización y en los modelos condicionales gaussianos. Este estudio debe detectar las carencias de estos métodos, que servirán de base al desarrollo de nuevos marcos de trabajo.

3. *Definición de un modelo probabilístico para variables multidimensionales mixtas.* El modelo debe ser tal que permita un tratamiento uniforme de las variables discretas y las continuas. También debe de permitir el uso de algoritmos de propagación estándar. Se debe abordar también la definición de las estructuras de datos necesarias para manejar dichos modelos.

4. *Propuesta de métodos para la construcción de los modelos a partir de datos.* Esta tarea de aprendizaje la divideremos en dos:

   - Aprendizaje de distribuciones marginales.
   - Estimación de distribuciones condicionadas.

Organización por capítulos

En el primer capítulo haremos un breve repaso de las cuestiones básicas de las redes bayesianas discretas, como primera toma de contacto con el tema. Empezaremos por las definiciones más básicas, para centrarnos posteriormente en algunos de los algoritmos de propagación que podemos aplicar a las redes bayesianas discretas, con la intención de adaptarlos posteriormente a nuestro esquema particular mixto. Con ese fin hablaremos de los algoritmos basados en cálculos locales, explicando las estructuras que son necesarias para poder llevar a cabos dichos algoritmos. Veremos el algoritmo HUGIN y el algoritmo Shenoy-Shafer, dentro de los exactos, y dentro de los aproximados introduciremos de entre los métodos de simulación el MCMC. Acabaremos este primer capítulo con una breve reseña acerca de cuestiones de aprendizaje en redes discretas, tanto estructural como paramétrico.

En los capítulos dos y tres hacemos una recopilación de las distintas alternativas de las que disponemos cuando nos enfrentamos a variables continuas en el marco de las redes bayesianas. Concretamente, dedicamos el capítulo dos de la memoria a la discretización, centrándonos en el método de discretización no uniforme propuesto por Kozlov y Koller (1997), que trata de discretizar las variables continuas pero no de forma individual tal y como se venía haciendo de forma casi sistemática, sino que plantea dividir el dominio de las variables continuas teniendo en cuenta a la vez a todas ellas, permitiendo por tanto un mejor ajuste a los datos. Proponen a su vez un método de discretización dinámico que tenga en cuenta la probabilidad de la evidencia que se introduce en la red.

El tercer capítulo trata del modelo condicional Gaussiano, en el que cohabitan variables continuas y discretas dentro de una red bayesiana (Lauritzen 1992; Cowell, Dawid, Lauritzen, y Spiegelhalter 1999). Explicamos su definición de potenciales y las operaciones permitidas sobre ellos, lo que nos lleva entonces a los conceptos de marginalizaciones fuertes y débiles, y a partir de ahí a la forma de propagar probabilidades, comentando los problemas que surgen, fundamentalmente debido a la restricción de que variables continuas no pueden tener hijos discretos.

Es a partir del cuarto capítulo cuando presentamos nuestro modelo propio para variables mixtas, el *modelo MTE*. Definimos los potenciales asociados, las operaciones permitidas y una estructura adecuada para su representación: los árboles de probabilidad mixtos.

Los capítulos cinco y seis abordan la cuestión del aprendizaje a partir de datos de los potenciales previamente definidos. Para ello, en el quinto capítulo resolvemos el problema de estimar una distribución MTE univariante, fijándonos en tres cuestiones principalmente; partición del dominio, determinación de la forma funcional de la distribución y estimación de los parámetros necesarios.

En el sexto capítulo retomamos el problema original, y nos planteamos aprender distribuciones condicionadas. Aplicando métodos propios de árboles de clasificación y utilizando la estimación de distribuciones univariantes del capítulo anterior obtenemos un método de estimación de distribuciones condicionadas, que mejora los métodos clásicos de discretización.

En el capítulo siete adaptamos a nuestro modelo los algoritmos que estudiamos en el primer capítulo, probando que podemos realizar las mismas operaciones de propagación de probabilidades sobre redes bayesianas cuyos potenciales sean MTE, y estén representados por árboles de probabilidad mixtos. Concretamente, para poder aplicar el algoritmo *Penniless* definimos una nueva operación sobre los potenciales MTE, la *poda* de árboles. De igual modo, para aplicar de forma correcta el método MCMC, damos un método para la simulación de variables MTE, basado en el método de inversión.

Concluimos la memoria con un capítulo de conclusiones y líneas abiertas, donde resumimos los objetivos alcanzados en la realización de esta tesis, y comentamos algunas de los problemas abiertos relacionados con el modelo MTE sobre los que seguir trabajando.

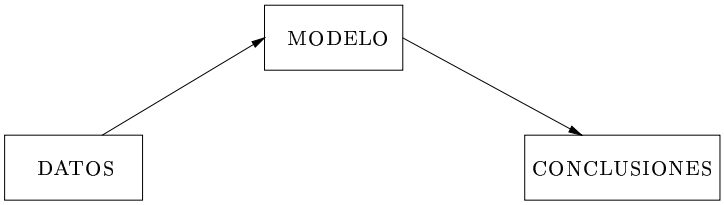
Antes de presentar la bibliografía consultada para la confección de esta memoria, incluimos un Anexo en el que detallamos el proceso de implementación de los algoritmos que hemos llevado a cabo, para el cual hemos utlizado las librerías del programa *Elvira*, aplicación construida en lenguaje JAVA para el tratamiento de modelos gráficos

probabilísticos.

Capítulo 1

Redes Bayesianas Discretas

El planteamiento general en todo análisis de datos puede expresarse por la siguiente figura :



A partir de unos datos empíricos construimos modelo con la que identificamos las variables que intervienen en el sistema objeto de análisis, así como las relaciones que hay entre ellas, y usamos ese modelo para obtener ciertas conclusiones acerca del problema original planteado.

Durante toda la memoria nos centraremos en el caso de los *modelos gráficos*, que constan de una parte cualitativa donde se expresan las relaciones entre las variables mediante enlaces entres ellas, y una parte cuantitativa donde se cuantifica la fuerza de esas relaciones. En concreto estudiaremos las *redes bayesianas*, donde las relaciones de las variables se cuantifican en términos de probabilidades.

¿Qué beneficios proporciona usar una red bayesiana en lugar de alguna otra herramienta clásica de análisis multivariante? Las ventajas son dos fundamentalmente; primero que podemos trabajar con un gran número de variables al mismo tiempo, cosa que puede resultar bastante complicada con las técnicas clásicas, pero sobre todo la representación gráfica que obtenemos con la red bayesiana, en la que se observa claramente y a primera vista las relaciones de dependencia/independencia de las variables en cuestión.

Pasemos pues a definir los conceptos fundamentales que iremos manejando a lo largo de la memoria.

1.1. Semántica del modelo gráfico

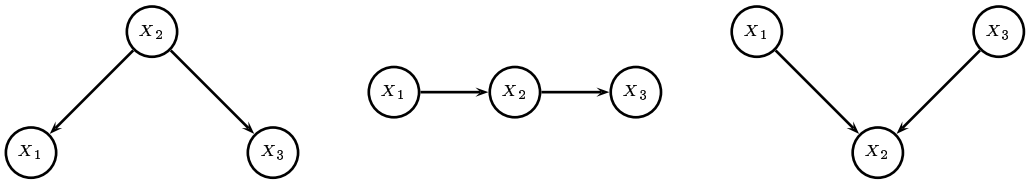
La forma de representar las relaciones entre las variables es mediante los llamados *grafos*.

Notaremos a las variables por letras mayúsculas, como X, Y y Z, mientras que las letras mayúsculas en negrita las usaremos para variables multidimensionales. Si $\mathbf{X}$ es una variable, $\mathbf{x}$ denotará un valor de esa variable.

Definición 1.1 *Un **grafo** es un par de conjuntos $\mathcal{G} = (\mathbf{X}, L)$, donde $\mathbf{X} = (X_1, \ldots, X_n)$ es un conjunto finito de elementos (nodos) y L es un conjunto de aristas, es decir, un subconjunto de pares ordenados de elementos distintos de X.*

El concepto de grafo puede definirse de forma más general: puede permitirse que dos nodos estén conectados por más de una arista, o que un nodo esté conectado consigo mismo. Sin embargo, en nuestro caso, los grafos se utilizan para representar un conjunto de variables (nodos) y unas relaciones de dependencia entre ellas (aristas), por lo que nos basta con la definición dada.Veamos algunas consideraciones más que necesitaremos sobre los grafos:

Definición 1.2 *Dado un grafo $\mathcal{G} = (\mathbf{X}, L)$, L_{ij} representa la arista entre X_i y X_j, si $L_{ij} \in L$ y $L_{ji} \notin L$, la arista L_{ij} se denomina **dirigida** y se denota mediante $X_i \rightarrow X_j$.*



a) Cola con cola **b)** Cabeza con cola **c)** Cabeza con cabeza

Figura 1.1: Tipos de enlaces en un grafo dirigido acíliclo

Definición 1.3 *Un* ***camino*** *del nodo X_i al nodo X_j es una sucesión de nodos $(X_{i_1}, \ldots, X_{i_r})$, comenzando en $X_{i_1} = X_i$ y finalizando en $X_{i_r} = X_j$, de forma que existe una arista del nodo X_{i_k} al nodo $X_{i_{k+1}}$, $k = 1, \ldots, r-1$.*

La *longitud* del camino, $(r-1)$, se define como el número de aristas que contiene, y un camino se dirá *cerrado* si el nodo inicial coincide con el final, esto es, $X_i = X_j$.

Definición 1.4 *Dado un grafo $\mathcal{G} = (\mathbf{X}, L)$ se llama* ***subgrafo*** *de $\mathcal{G}$ a $\mathcal{G}_A = (A, L_A)$ si $A \subseteq X$ y $L_A \subseteq E \cap (A \times A)$.*

Definición 1.5 *Llamamos* **grafo dirigido acíclico** *a todo grafo cuyas aristas sean todas dirigidas y ninguno de sus caminos cerrado.*

Proposición 1.1 *Dado un grafo dirigido acíclico, todas las conexiones entre nodos que nos podemos encontrar son: cola con cola, cabeza con cola y cabeza con cabeza (ver figura 1.1).*

Definición 1.6 *Sea $\mathcal{G} = (\mathbf{X}, L)$ un grafo dirigido acíclico. Sea $\mathbf{X}_K \subseteq \mathbf{X}$, $X_i, X_j \in \mathbf{X} \setminus \mathbf{X}_K$ y α un camino entre X_i y X_j. Se dice que el camino está bloqueado por $\mathbf{X}_K$ si se cumple alguna de estas dos condiciones:*

1. *Existe algún vértice $X_k \in \mathbf{X}_K$ del camino α tal que los arcos incidentes en X_k se encuentran cola con cola o cabeza con cola.*

2. *Hay algún vértice $X_k \in \mathbf{X}$ del camino α tal que los arcos incidentes en X_k se encuentran cabeza con cabeza, y es tal que ni X_k ni ninguno de sus descendientes están en $\mathbf{X}_K$.*

Las relaciones de independencia entre variables se expresan en un grafo mediante el concepto de *d-separación* (Pearl 1987).

Definición 1.7 *Sea $\mathcal{G} = (\mathbf{X}, L)$ un grafo dirigido acíclico.*

1. *Sean $\mathbf{X}_K \subsetneq \mathbf{X}$ y $X_i, X_j \in \mathbf{X} \setminus \mathbf{X}_K$. Se dice que X_i y X_j están* **d-separados** *por $\mathbf{X}_K$ si y sólo si cualquier camino entre X_i y X_j está bloqueado por $\mathbf{X}_K$.*

2. *Sean $\mathbf{X}_{K_1}, \mathbf{X}_{K_2}, \mathbf{X}_K \subsetneq \mathbf{X}$ y disjuntos. Se dice que X_{K_1} y X_{K_2} están* **d-separados** *por $\mathbf{X}_K$ si cualquier camino entre $X_i \in \mathbf{X}_{K_1}$ y $X_j \in \mathbf{X}_{K_2}$ está bloqueado por $\mathbf{X}_K$.*

La d-separación entre nodos del grafo se corresponde en los modelos gráficos probabilísticos con independencia condicional de las correspondientes variables aleatorias.

Por tanto, los tres tipos de enlaces que aparecen en un grafo dirigido acíclico se corresponden a las siguientes independencias:

- Figura 1.1, **a)** y **b)**: las variables X_1 y X_3 serán independientes dado la variable X_2.

- Figura 1.1, **c)**: las variables X_1 y X_3 serán independientes si no conocemos la variable X_2 ni ninguno de sus descendientes.

1.2. Redes bayesianas

Pero un modelo gráfico probabilístico no es sólo la representación cualitativa, sino también cuantitativa, de las relaciones de las variables, y lo representaremos asignando

números a cada uno de los arcos de los ejemplos anteriores. Si A es el padre de B, es natural que $P(B|A)$ sea el número que represente la fuerza del enlace entre A y B, pero si C es también padre de B, entonces las dos probabilidades, $P(B|A)$ y $P(B|C)$, por sí solas no nos dicen nada de cómo se relacionan entre sí las tres variables, ya que pueden interaccionar de diferentes formas, por lo que necesitamos especificar $P(B|A,C)$.
Para fijarlo claramente, introducimos la definición de *red bayesiana*.

Definición 1.8 *Una* **red bayesiana** *es un par $(\mathcal{G}, P)$ donde $\mathcal{G}$ es un grafo dirigido acíclico en el que cada vértice es una variable aleatoria, $P = \{p(x_1|\boldsymbol{\pi}_1), \ldots, p(x_n|\boldsymbol{\pi}_n)\}$ es un conjunto de n funciones de probabilidad condicionada, una para cada variable, y $\boldsymbol{\pi}_i$ es el conjunto de padres del nodo X_i en $\mathcal{G}$. El conjunto P define una función de probabilidad asociada mediante la factorización*

$$p(x) = \prod_{i=1}^{n} p(x_i|\boldsymbol{\pi}_i) \quad x \in \Omega_X \tag{1.1}$$

dónde Ω_X es el conjunto de posibles valores de la variable X.

Las distribuciones de probabilidad y los resultados de operar con ellas las expresaremos como *potenciales* (Lauritzen y Spiegelhalter 1988):

Definición 1.9 *Dado un conjunto de variables $\mathbf{X}_I$, llamaremos potencial a cualquier función $f : \Omega_{\mathbf{X}_I} \rightarrow R_0^+$.*

De forma intuitiva, podemos decir que los potenciales representan información probabilística sobre un conjunto de variables.

A continuación definiremos una serie de operaciones primitivas para manipular dicha información.[1]:

[1]Estas operaciones son genéricas, dependiendo de la forma de los potenciales, las definiremos más precisamente cuando sea necesario.

Definición 1.10 *Dado un potencial f definido sobre un conjunto de variables $\mathbf{X}_I$, y dado $J \subseteq I$, se define la* ***marginalización*** *de f sobre las variables $\mathbf{X}_J$ (o la eliminación de las variables con índices en $I \setminus J$) como un nuevo potencial $f^{\downarrow J}$ definido sobre el conjunto de variables con índices en J dado por la expresión*

$$f^{\downarrow J}(\mathbf{x}) = \sum_{\mathbf{y} \in \Omega_{\mathbf{X}_I},\ \mathbf{y}^{\downarrow J} = \mathbf{x}} f(\mathbf{y}), \qquad \forall \mathbf{x} \in \Omega_{\mathbf{X}_J}$$

Definición 1.11 *Dados r potenciales $f_1, \ldots, f_r$, cada uno de ellos definido sobre los conjuntos $\Omega_{\mathbf{X}_{I_1}}, \ldots, \Omega_{\mathbf{X}_{I_r}}$, se define su* ***combinación o producto*** *como un potencial definido sobre el conjunto de variables con índices en $I = \bigcup_{i=1}^{r} I_i$ y dado por la expresión*

$$f(\mathbf{x}) = \prod_{i=1}^{r} f_i(\mathbf{x}^{\downarrow I_i}), \quad \forall \mathbf{x} \in \Omega_{\mathbf{X}_I}.$$

Definición 1.12 *Sea f un potencial definido para el conjunto de variables con índices en I, sea un conjunto de variables $\mathbf{X}_J$, $J \subset I$ y un elemento fijo $\mathbf{x}_0 \in \Omega_{X_J}$. La* ***restricción*** *de f a los valores $\mathbf{x}_0$ es un nuevo potencial $f^{R(\mathbf{X}_J = \mathbf{x}_0)}$ definido sobre las variables con índices en $I \setminus J$ de acuerdo con la siguiente expresión:*

$$f^{R(\mathbf{X}_J = \mathbf{x}_0)}(\mathbf{x}) = f(\mathbf{y}), \quad \forall \mathbf{x} \in \Omega_{\mathbf{X}_{I \setminus J}}$$

tal que $\mathbf{y} \in \Omega_{\mathbf{X}_I}$ es tal que $\mathbf{y}^{\downarrow I \setminus J} = \mathbf{x}$ e $\mathbf{y}^{\downarrow J} = \mathbf{x}_0$.

En nuestro caso, al tratar en este capítulo exclusivamente con variables discretas y finitas, los potenciales serán tablas, con tantas entradas como posibles configuraciones haya de las variables involucradas.

La factorización de la distribución de probabilidad de una red bayesiana se obtiene de forma sencilla a partir del grafo dirigido considerando un conjunto de funciones de probabilidad condicionada que involucran a cada nodo con sus padres, como vimos en (1.1).

1.3. Propagación de probabilidades. Algoritmos exactos

Una de las opciones más importantes de una red bayesiana consiste en obtener conclusiones a medida que se va conociendo más información o *evidencia*. Por ejemplo, en el área médica puede ser obtener un diagnóstico para un determinado paciente que presenta ciertos síntomas (evidencia). El mecanismo para obtener conclusiones a partir de la evidencia se conoce como *propagación de la evidencia* o, simplemente, *propagación*. Esta tarea consiste en actualizar las probabilidades de las variables en función de las observaciones. En el caso del diagnóstico médico, se trata de conocer las probabilidades de cada una de las enfermedades, dados los síntomas observados en el paciente.

Supóngase un conjunto de variables discretas $\mathbf{X} = \{X_1, \ldots, X_n\}$ y una función de probabilidad $p(\mathbf{x})$ en $\mathbf{X}$. Cuando no se dispone de ninguna información, es decir, cuando no existe evidencia, el proceso de propagación consiste en calcular las probabilidades marginales $p(X_i = x_i)$, también denotadas por $p(x_i)$, para cada $X_i \in \mathbf{X}$. Estas probabilidades proporcionan información *a priori* sobre los distintos valores que pueden tomar las variables.

Cuando se dispone de cierta evidencia, es decir, cuando se conoce un conjunto de variables $\mathbf{E} \subset \mathbf{X}$ que tienen asociadas los valores $X_i = e_i$, para $X_i \in \mathbf{E}$, el proceso de propagación debe tener en cuenta estos valores para calcular las nuevas probabilidades de las variables.

Así, la propagación de la evidencia consiste en calcular las funciones de probabilidad condicionada $p(x_i|\mathbf{e})$ para cada variable $X_i \notin \mathbf{E}$, dada la evidencia $\mathbf{E} = \mathbf{e}$. Estas funciones de probabilidad condicionada miden el efecto producido por la evidencia en cada variable. Cuando no se dispone de evidencia, $(\mathbf{E} = \emptyset)$, las funciones condicionadas son simplemente las funciones de probabilidad marginal $p(x_i)$.

Al tratarse de funciones de probabilidad condicionada, la forma usual de calcular estas funciones es

$$p(x_i|\mathbf{e}) = \frac{p(x_i, \mathbf{e})}{p(\mathbf{e})} \propto p(x_i, \mathbf{e})$$

Por tanto, se puede obtener $p(x_i|\mathbf{e})$, calculando y normalizando las probabilidades marginales $p(x_i, \mathbf{e})$; de esta forma se tiene

$$p(x_i, \mathbf{e}) = \sum_{\mathbf{x}\setminus\{x_i,\mathbf{e}\}} p_{\mathbf{e}}(x_i, \ldots, x_n),$$

donde $p_{\mathbf{e}}(x_1, \ldots, x_n)$ es la función de probabilidad obtenida al sustituir en $p(x_1, \ldots, x_n)$ las variables con evidencia, $\mathbf{E}$, por sus valores, $\mathbf{e}$. Por tanto, para calcular $p(x_i, \mathbf{e})$ ha de sumarse $p_{\mathbf{e}}(x_1, \ldots, x_n)$ para todas las combinaciones posibles de valores de las variables que no estén contenidas en $\mathbf{E}$, excepto la variable X_i. Simplemente lo que hacemos es marginalizar a partir de la distribución conjunta, una vez incorporada la evidencia, mediante la operación de restricción.

Debido al alto número de combinaciones de valores que conlleva este método resulta altamente ineficiente y también presenta problemas de almacenamiento; por ejemplo, si todas las variables de la red son binarias, sería necesario almacenar 2^n valores valores de probabilidad para una red de n variables. Así esto no es viable para redes medianamente grandes.

El problema de hacer la propagación de esta forma es que no se tiene en cuenta la estructura de independencia contenida en la función de probabilidad $p(\mathbf{x})$. El número de cálculos necesarios en el proceso de propagación puede ser reducido de forma considerable, teniendo en cuenta las relaciones de independencia que hay entre las variables de la función de probabilidad $p(\mathbf{x})$. De esta forma puede obtenerse la distribución a posteriori mediante cálculos locales, acelerando el proceso de inferencia y sin necesidad de trabajar con la distribución conjunta.

Basados en esta idea de realizar los cálculos localmente hay muchos algoritmos, como puede ser el *algoritmo para inferencia en poliárboles* de Pearl (1986b, 1988), los basados en las *técnica de eliminación de nodos* (Shachter y Peot 1990; Zhang y Poole 1996) y los llamados *métodos de agrupamiento* (Jensen, Lauritzen, y Olesen 1990; Jensen, Olesen, y Andersen 1990; Lauritzen y Spiegelhalter 1988). Estos métodos de

agrupamiento se basan en la construcción de subconjuntos de nodos que capturen las estructuras locales del modelo probabilístico asociado al grafo. De esta forma, el proceso de propagación de evidencia puede ser realizado calculando probabilidades locales (que depende de un número reducido de variables), evitando así calcular probabilidades globales (que dependen de todas las variables y que son las difíciles de calcular cuando el número de variables es elevado).

1.3.1. Árbol de cliques

Como hemos mencionado antes, estos algoritmos no trabajan directamente sobre la red bayesiana, sino sobre lo que se llama un *árbol de cliques*, que es un tipo especial de *árbol de grupos*.

Definición 1.13 *Un* ***árbol*** *es un grafo no dirigido acíclico.*

Definición 1.14 *Un grafo no dirigido se denomina* **completo** *si contiene una arista entre cada par de nodos.*

Definición 1.15 *Dado* $\mathbf{X}$ *el conjunto de nodos de un grafo, un* ***árbol de grupos*** *sobre* $\mathbf{X}$ *es un árbol cuyos nodos son subconjuntos de* $\mathbf{X}$*, donde para cada par de nodos* U*,* V*, todos los nodos en el camino entre* U *y* V *contienen a la intersección* $U \cap V$*. La unión de todos los nodos es* $\mathbf{X}$*.*

Definición 1.16 *Un subgrafo completo se dice* ***maximal*** *si no es un subgrafo propio de otro subgrafo completo.*

Definición 1.17 *Un subgrafo completo se denomina* **clique** *si es maximal.*

Definición 1.18 *Un árbol de uniones se llama* ***árbol de cliques*** *si todos los nodos son cliques.*

Este árbol de cliques surge del *grafo moral*, que es un grafo especial que se obtiene a partir de la red bayesiana.

Definición 1.19 *Un **grafo moral** asociado a una red bayesiana es un grafo no dirigido en el que todos los padres de un mismo hijo están unidos entre sí.*

Por tanto, dada una red bayesiana cualquiera, podemos construirnos su grafo moral asociado simplemente quitando las direcciones de las aristas y conectando todas las variables de la red que tengan hijos en común. Ahora que sabemos como obtener el grafo moral, veamos cómo obtener el árbol de cliques. Para ello necesitaremos las siguientes definiciones:

Definición 1.20 *La **eliminación** de un vértice de un grafo no dirigido es el proceso por el cual primero se añaden las aristas necesarias para que ese vértice y sus nodos adyacentes formen un subgrafo completo y después se elimina del grafo el vértice con las aristas incidentes en él.*

Definición 1.21 *Una **cuerda** es una arista que une dos nodos de un ciclo y que no pertenece al ciclo.*

Definición 1.22 ***Triangular** un grafo es añadir cuerdas que dividan los ciclos de longitud mayor que tres. De esta forma un grafo triangulado tiene ciclos como mucho de longitud tres.*

La siguiente proposición, (Cowell, Dawid, Lauritzen, y Spiegelhalter 1999; Jensen 1996), nos dice como triangular un grafo moral, paso intermedio para la obtención del árbol de cliques.

Proposición 1.2 *Dado un grafo moral, si establecemos un orden de eliminación de las variables y las eliminamos en ese orden, el grafo formado por el grafo original y las nuevas aristas añadidas en el proceso de eliminación de las variables resulta ser triangulado.*

Veamos pues como podemos construirnos un árbol de cliques a partir de un grafo dirigido acíclico:

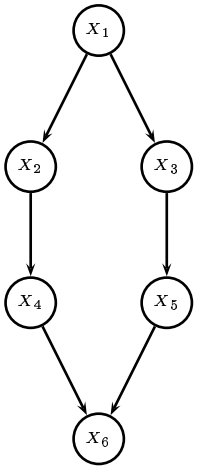
ARBOL_CLIQUES($\mathcal{G}$)

ENTRADA: una red bayesiana $\mathcal{G}$.
SALIDA: un árbol de cliques $\mathcal{T}$ asociado a la red $\mathcal{G}$.

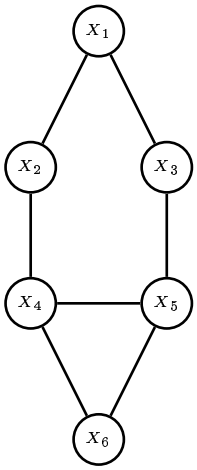
1. Construir el grafo moral asociado a la red $\mathcal{G}$.
2. Definir un orden de eliminación de las variables.
3. Triangular el grafo moral, mediante el proceso de eliminación de variables.
4. Obtener los cliques que se han ido generando en el proceso de eliminación de variables al triangular el grafo moral.
5. Para cada clique C_i generado, empezando por el generado en último lugar:
   a) Añadir el clique C_i al árbol.
   b) Añadir una arista entre el clique C_i y aquel clique C_j del árbol con el que tenga más variables en común, siempre que no formen ciclo.
6. Para cada par de cliques adyacentes del árbol, C_i y C_j, si $C_i \subseteq C_j$, eliminar C_i y la correspondiente arista[2]
7. DEVUELVE $\mathcal{T}$.

Ejemplo 1.1 *Supongamos el siguiente grafo dirigído acíclico :*

[2]De este modo nos aseguramos que los nodos del árbol son cliques maximales del grafo triangulado.



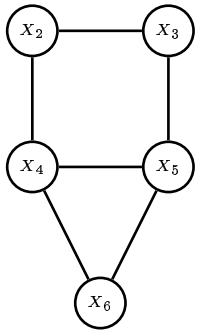
Para construir un árbol de cliques asociado a él, primero se moraliza el grafo, esto es, se quita la dirección a las aristas y se unen aquellas variables con hijos en común, en nuestro caso X_4 y X_5 ya que tienen en común a X_6 como hijo. Así queda el siguiente grafo moral:



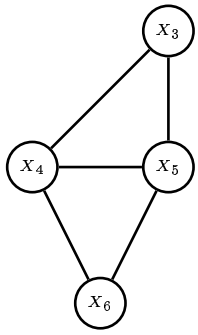
Ahora escogemos una secuencia de eliminación de variables, por ejemplo la secuencia $X_1, X_2, X_3, X_4, X_5, X_6$. Eliminemos paso a paso cada una de estas variables :

Eliminar X_1. *Para que X_1 y sus nodos adyacentes formen un subgrafo completo se*

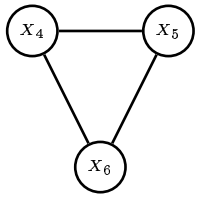
añade una arista entre X_2 y X_3. Así el subgrafo completo que se obtiene es el formado por $\{X_1, X_2, X_3\}$. Al eliminar X_1 :



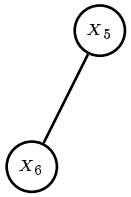
Eliminar X_2. *Se añade una arista entre X_4 y X_3, con lo que queda el subgrafo completo $\{X_2, X_3, X_4\}$. Al eliminar X_2 queda :*



Eliminar X_3. *Ahora $\{X_3, X_4, X_5\}$ forma ya de por si un subgrafo completo, por lo que no hay que insertar ninguna arista adicional. Tras eliminar X_3 queda*



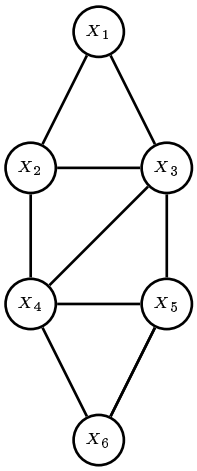
Eliminar X_4**.** *De nuevo no es necesario añadir ninguna arista, ya que el subgrafo* $\{X_4,X_5,X_6\}$ *es completo. Simplemente se elimina* X_4*:*



Eliminar X_5**.** *No se insertan aristas, ya que* $\{X_5,X_6\}$ *es completo obviamente. Al eliminar* X_5 *queda un grafo con un sólo nodo y lógicamente ninguna arista:*

Eliminar X_6**.** *Es este caso el subgrafo completo esta formado únicamente por* $\{X_6\}$*, y al eliminarlo no queda nada.*

Una vez que eliminadas todas las variables, por la proposición 1.1, el grafo

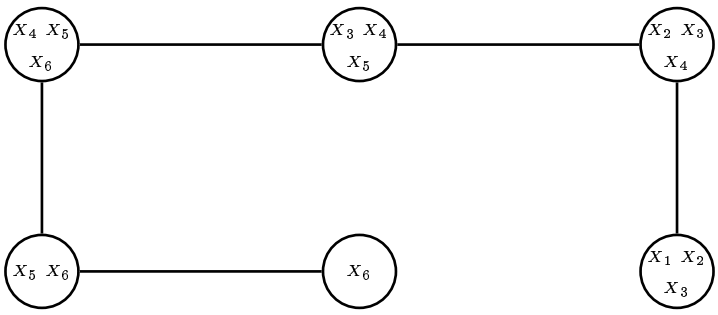


es triangulado.

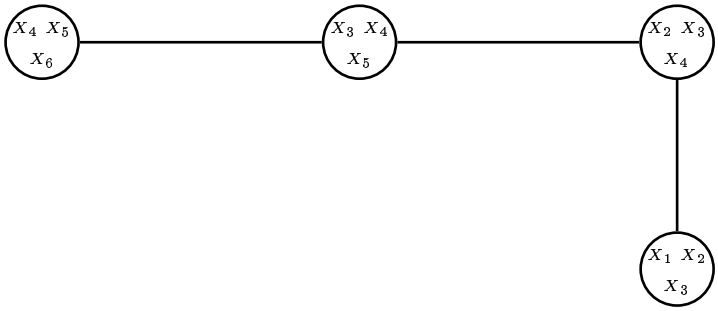
Así los nodos del árbol de cliques son los subgrafos completos maximales a los que se ha hecho referencia en cada uno de los pasos de eliminación. Se insertan los cliques en orden inverso a como se han ido obteniendo, esto es, el primer clique sería $\{X_6\}$*, el siguiente, el que se obtuvo al eliminar* X_5*, y así con todos :*

$$\begin{aligned} C_1 &= \{X_6\} \\ C_2 &= \{X_5, X_6\} \\ C_3 &= \{X_4, X_5, X_6\} \\ C_4 &= \{X_3, X_4, X_5\} \\ C_5 &= \{X_2, X_3, X_4\} \\ C_6 &= \{X_1, X_2, X_3\} \end{aligned}$$

Al insertar C_2 *se une con* C_1*. Así lo vamos haciendo con todos los cliques; cada clique que insertamos lo unimos con el clique que ya esté en el árbol con el que tenga mayor número de variables en común:*



Ahora bien, $C_1 \subseteq C_2$*, lo que quiere decir que* C_1 *y* C_2 *no aportan información adicional a la que ya tengamos en* C_3*, luego lo que hacemos es embeberlos en su vecino, esto es,* C_1 *en* C_2*, y* C_2 *en* C_3 *y por tanto nos queda*



y éste sería el árbol de cliques asociado al grafo dirigido acíclico del principio.

Pero una red bayesiana no es sólo un grafo dirigido acíclico, sino que también tiene una parte cuantitativa representada en forma de distribuciones de probabilidad. Por esto, cada clique tiene también un potencial asociado a él, es decir, una tabla real valuada sobre todas las posibles configuraciones de su conjunto de variables. La forma de definir estos potenciales es la siguiente: Primero asignamos a todos los cliques potenciales unidad, es decir, tablas de unos solamente. Posteriormente para cada variable X_i de la red escogemos un clique que contenga el conjunto $\mathbf{\Pi}_i \cup \{X_i\}$ y multiplicamos el potencial del clique por la función de probabilidad condicionada $P(X_i|\mathbf{\Pi}_i)$, que también viene expresada en forma de tabla. Así, notando por ϕ_{C_i} al potencial del clique C_i tenemos que

$$\phi_{C_i} = \prod_{x_j \in \mathbf{A}_i} p(x_j|\boldsymbol{\pi}_j),$$

con $\mathbf{A}_i$ el conjunto de nodos asociado al clique C_i

A continuación explicamos más en detalle dos de los métodos de agrupamiento más utilizados, el *HUGIN* (Andersen, Olesen, Jensen, y Jensen 1989; Jensen, Lauritzen, y Olesen 1990; Jensen, Olesen, y Andersen 1990), y el Shenoy-Shaffer (Shenoy y Shafer 1990).

1.3.2. Algoritmo HUGIN

Una vez obtenida esta estructura (árbol y potenciales) podemos ya realizar la propagación.

Para ello se sigue un esquema de paso de *mensajes* entre nodos adyacentes, de modo que al finalizar el algoritmo tenemos en cada nodo la distribución *a posteriori* de las variables asocidas a éstos, luego para conocer la distribución *a posteriori* de una variable sólo tenemos que marginalizar aquella distribución que esté asociada a algún clique al que pertenezca la variable.

El envio de un mensaje de un clique C_i a otro C_j se puede considerar como el paso de la información contenida en C_i a C_j, información que el clique C_j incorporará a su potencial, y que a su vez enviará en forma de mensaje a sus otros cliques vecinos.

Definición 1.23 *Dado un árbol de cliques, se define el* **separador** S *de una arista entre los cliques* C_i *y* C_j *como el conjunto de variables que tienen en común los dos cliques*

$$S = C_i \cap C_j$$

Proposición 1.3 *(Jensen 1996) Sea un árbol de cliques derivado de una red bayesiana definida entonces la distribución de probabilidad conjunta de la red se puede obtener como el producto de todos los potenciales de los cliques dividido por el producto de los potenciales de los separadores.*

Para poder realizar la propagación en el árbol de cliques es necesario definir una nueva operación, la *absorción.*

Definición 1.24 *Dados dos cliques adyacentes* C_i *y* C_j *con separador* S*, se dice que* C_i *absorbe de* C_j *si los nuevos potenciales de los cliques asociados pasan a ser :*

$$\begin{aligned}
\phi^*_S(\mathbf{x}_S) &= \phi^{\downarrow S}_{C_j}(\mathbf{x}_{C_j}), \\
\phi^*_{C_i}(\mathbf{x}_{C_i}) &= \phi_{C_i}(\mathbf{x}_{C_i})\frac{\phi^*_S(\mathbf{x}_S)}{\phi_S(\mathbf{x}_S)}, \\
\phi^*_{C_j}(\mathbf{x}_{C_j}) &= \phi_{C_j}(\mathbf{x}_{C_j}).
\end{aligned}$$

La idea que hay detrás de la absorción es que la información que C_i y C_j tienen en común es lo que hay en el separador S, y eso es lo que C_i recibe de C_j. Si la operación de absorción entre dos cliques no cambia los potenciales de éstos, diremos que el enlace que los une es *consistente*.

El paso de mensajes se realiza en dos fases: RECOLECCIÓN Y DISTRIBUCIÓN, que podemos describir como sigue:

1. **Petición de recolección**. Supongamos que un clique C_i recibe un mensaje de recolección de otro clique C_j; entonces C_i manda un mensaje de recolección a todos sus vecinos excepto a C_j. Cuando un clique termina la operación de recolección, el clique que la solicitó absorbe de éste.

2. **Petición de distribución**. Si un clique C_i recibe un mensaje de distribución desde un clique C_j, entonces C_i absorbe de C_j y manda un mensaje de distribución a todos sus vecinos excepto C_j.

Como dijimos anteriormente la propagación consiste en calcular las probabilidades marginales de las variables de interés dada la evidencia, es decir, en algún momento tenemos que incorporar al proceso de propagación la información que tengamos acerca de una o varias variables, la evidencia.

Definición 1.25 *Toda observación $X_i = e_i$ tiene asociada una función de Dirac definida sobre Ω_i como sigue:*

$$\delta_i(x_i; e_i) = \begin{cases} 1 & si\ e_i = x_i, \\ 0 & si\ e_i \neq x_i. \end{cases}$$

Esta función de Dirac será el potencial asociado a la evidencia e_i.

Así pues, cada evidencia se incorpora a un sólo clique que contenga a la variable observada multiplicando su potencial por el potencial asociado a la evidencia. Lo

incorporamos sólo a un clique ya que al hacer la propagación, la información proporcionada por la observación se repartirá por todo el árbol alcanzando a todos los cliques.

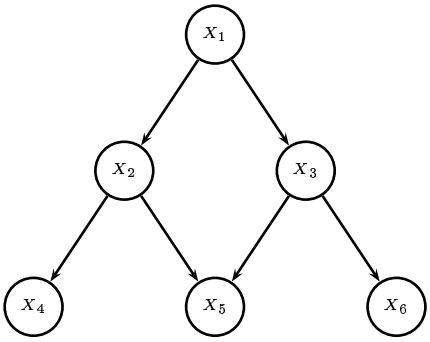
Ahora ya podemos enunciar el **algoritmo HUGIN de propagación**.

HUGIN($\mathcal{G}$)

ENTRADA: una red bayesiana $\mathcal{G}$.
SALIDA: el árbol de cliques $\mathcal{T}$ asociado a la red $\mathcal{G}$ con los potenciales actualizados.

1. Construir el árbol de cliques $\mathcal{T}$ asociado a la red $\mathcal{G}$.
2. Para cada arista del árbol, crear el correspondiente separador.
3. Calcular los potenciales asociados a cada clique del árbol, y asignar potenciales unidad a los separadores.
4. Incorporar las evidencias.
5. Seleccionar un clique R como raíz.
6. Enviar un mensaje de recolección a partir de R.
7. Enviar un mensaje de distribución desde R.
8. DEVUELVE $\mathcal{T}$.

Ejemplo 1.2 *(Castillo, Gutiérrez, y Hadi 1996) Considérese el siguiente grafo dirigido:*



Este grafo implica la siguiente factorización de la función de probabilidad :

$$p(x_1, x_2, x_3, x_4, x_5, x_6) = p(x_1)p(x_2|x_1)p(x_3|x_1)p(x_4|x_2)p(x_5|x_2, x_3)p(x_6|x_3).$$

Los valores numéricos asociados a esas probabilidades condicionadas se muestran en las tablas siguientes:

x_1	$p(x_1)$
0	*0.3*
1	*0.7*

x_1	x_3	$p(x_3\|x_1)$
0	*0*	*0.2*
0	*1*	*0.8*
1	*0*	*0.5*
1	*1*	*0.5*

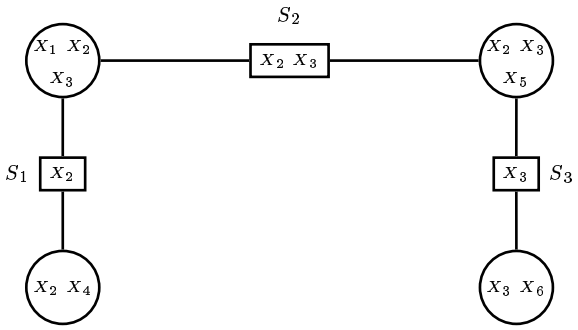
x_1	x_2	$p(x_2\|x_1)$
0	*0*	*0.4*
0	*1*	*0.6*
1	*0*	*0.1*
1	*1*	*0.9*

x_2	x_4	$p(x_4\|x_2)$
0	*0*	*0.3*
0	*1*	*0.7*
1	*0*	*0.2*
1	*1*	*0.8*

x_3	x_6	$p(x_6\|x_3)$
0	*0*	*0.1*
0	*1*	*0.9*
1	*0*	*0.4*
1	*1*	*0.6*

x_2	x_3	x_5	$p(x_5\|x_2,x_3)$
0	*0*	*0*	*0.4*
0	*0*	*1*	*0.6*
0	*1*	*0*	*0.5*
0	*1*	*1*	*0.5*
1	*0*	*0*	*0.7*
1	*0*	*1*	*0.3*
1	*1*	*0*	*0.2*
1	*1*	*1*	*0.8*

Aplicando el algoritmo de propagación expuesto arriba, lo primero que hay que hacer es construir el árbol de cliques y sus separadores



El siguiente paso es asignar familias a los nodos; por ejemplo:

$$X_1, X_2, X_3 \to C_1 \ ; \ X_5 \to C_2 \ ; \ X_4 \to C_3 \ ; \ X_6 \to C_4.$$

Así,

$$\begin{aligned}
\phi_{C_1}(x_1,x_2,x_3) &= p(x_1)p(x_2|x_1)p(x_3|x_1), \\
\phi_{C_2}(x_2,x_3,x_5) &= p(x_5|x_2,x_3), \\
\phi_{C_3}(x_2,x_4) &= p(x_4|x_2), \\
\phi_{C_4}(x_3,x_6) &= p(x_6|x_3),
\end{aligned}$$

es decir, los potenciales serán

x_1	x_2	x_3	$\phi_{C_1}(x_1, x_2, x_3)$
0	*0*	*0*	*0.024*
0	*0*	*1*	*0.096*
0	*1*	*0*	*0.036*
0	*1*	*1*	*0.144*
1	*0*	*0*	*0.035*
1	*0*	*1*	*0.035*
1	*1*	*0*	*0.315*
1	*1*	*1*	*0.315*

x_2	x_3	x_5	$\phi_{C_2}(x_2, x_3, x_5)$
0	*0*	*0*	*0.40*
0	*0*	*1*	*0.60*
0	*1*	*0*	*0.50*
0	*1*	*1*	*0.50*
1	*0*	*0*	*0.70*
1	*0*	*1*	*0.30*
1	*1*	*0*	*0.20*
1	*1*	*1*	*0.80*

x_2	x_4	$\phi_{C_3}(x_2, x_4)$
0	*0*	*0.30*
0	*1*	*0.70*
1	*0*	*0.20*
1	*1*	*0.80*

x_3	x_6	$\phi_{C_4}(x_3, x_6)$
0	*0*	*0.10*
0	*1*	*0.90*
1	*0*	*0.40*
1	*1*	*0.60*

Supongamos ahora que sabemos el valor que toman las variables $\mathbf{E} = \{X_1, X_3, X_5\}$: $X_1 = 0, X_3 = 1, X_5 = 0$. *Así las funciones de Dirac asociadas a estas evidencias serían:*

x_1	$\delta_1(x_1; 0)$
0	*1*
1	*0*

x_3	$\delta_3(x_3; 1)$
0	*0*
1	*1*

x_5	$\delta_5(x_5; 0)$
0	*1*
1	*0*

Se incorpora cada evidencia al potencial de un clique multiplicando dicho potencial por la función de Dirac de la variable observada siempre que la variable esté en el potencial:

$$\begin{aligned}
\phi^*_{C_1}(x_1, x_2, x_3) &= \phi_{C_1}(x_1, x_2, x_3)\delta_1(x_1; 0), \\
\phi^*_{C_2}(x_2, x_3, x_5) &= \phi_{C_2}(x_2, x_3, x_5)\delta_5(x_5; 0), \\
\phi^*_{C_3}(x_2, x_4) &= \phi_{C_3}(x_2, x_4), \\
\phi^*_{C_4}(x_3, x_6) &= \phi_{C_4}(x_3, x_6)\delta_3(x_3; 1).
\end{aligned}$$

Por tanto los potenciales quedan :

x_1	x_2	x_3	$\phi_{C_1}(x_1, x_2, x_3)$
0	*0*	*0*	*0.024*
0	*0*	*1*	*0.096*
0	*1*	*0*	*0.036*
0	*1*	*1*	*0.144*
1	*0*	*0*	*0*
1	*0*	*1*	*0*
1	*1*	*0*	*0*
1	*1*	*1*	*0*

x_2	x_3	x_5	$\phi_{C_2}(x_2, x_3, x_5)$
0	*0*	*0*	*0.4*
0	*0*	*1*	*0*
0	*1*	*0*	*0.5*
0	*1*	*1*	*0*
1	*0*	*0*	*0.7*
1	*0*	*1*	*0*
1	*1*	*0*	*0.2*
1	*1*	*1*	*0*

x_2	x_4	$\phi_{C_3}(x_2, x_4)$
0	*0*	*0.30*
0	*1*	*0.70*
1	*0*	*0.20*
1	*1*	*0.80*

x_3	x_6	$\phi_{C_4}(x_3, x_6)$
0	*0*	*0*
0	*1*	*0*
1	*0*	*0.40*
1	*1*	*0.60*

Ahora, se selecciona el clique C_1 como clique raíz y a partir de él se envía un mensaje de recolección. La secuencia sería:

1. *C_1 envía un mensaje de recoleción a C_2.*
2. *Al llegarle el mensaje a C_2, éste se lo envía a todos sus vecinos excepto a C_1, es decir a C_4.*

3. *Como C_4 no tiene vecinos a los que mandar el mensaje de recolección, termina y por tanto el clique que le envió el mensaje absorbe de él, es decir, C_2 absorbe de C_4.*

4. *Ahora ya como C_2 ha terminado, C_1 absorbe de él.*

5. *C_1 le envía un mensaje de recolección a su otro vecino, C_3.*

6. *Como C_3 no tiene vecinos a los que enviar el mensaje, C_1 absorbe de él.*

Así terminaría la etapa de recolección. En ella hay tres absorciones, C_2 de C_4 , C_1 de C_2 y C_1 de C_3, en este orden:

C_2 absorbe de C_4 : *el nuevo potencial de S_3, ϕ_{S_3} será :*

$$\phi^*_{S_3}(x_3) = \sum_{x_6} \phi_{C_4}(x_3, x_6),$$

que da como resultado la función de Dirac para la variable X_3. Ahora

$$\phi^*_{C_2}(x_2, x_3, x_5) = \phi_{C_2}(x_2, x_3, x_5)\frac{\phi^*_{S_3}}{\phi_{S_3}}$$

como ϕ_{S_3} es 1, el nuevo potencial de C_2 es el producto de $\phi^*_{S_3}$ por el potencial de C_2, es decir, se incorporará a C_2 el conocimiento que tenemos de la variable X_3 y que estaba en el clique C_4:

x_2	x_3	x_5	$\phi_{C_2}(x_2, x_3, x_5)$
0	0	0	$0.4 \times 0 = 0$
0	0	1	$0 \times 0 = 0$
0	1	0	$0.5 \times 1 = 0.5$
0	1	1	$0 \times 1 = 0$
1	0	0	$0.7 \times 0 = 0$
1	0	1	$0 \times 0 = 0$
1	1	0	$0.2 \times 1 = 0.2$
1	1	1	$0 \times 1 = 0$

C_1 absorbe de C_2 : *el nuevo potencial de S_1 será:*

$$\phi^*_{S_1}(x_2, x_3) = \sum_{x_5} \phi_{C_2}(x_2, x_3, x_5) \equiv$$

x_2	x_3	$\phi^*_{S_1}(x_2, x_3)$
0	*0*	*0 + 0 = 0*
0	*1*	*0.5 + 0 = 0.5*
1	*0*	*0 + 0 = 0*
1	*1*	*0.2 + 0 = 0.2*

Aquí la información que tienen en común C_1 y C_2 es de las variables X_2 y X_3; X_3 sí es conocida, pero X_2 no, por eso el potencial del separador se modifica. Actualizando el potencial de C_1:

$$\phi^*_{C_1}(x_1, x_2, x_3) = \phi_{C_1}(x_1, x_2, x_3) \frac{\phi^*_{S_1}(x_2, x_3)}{\phi_{S_1}(x_2, x_3)}$$

x_1	x_2	x_3	$\phi_{C_1}(x_1, x_2, x_3)$
0	*0*	*0*	*0.024 × 0 = 0*
0	*0*	*1*	*0.096 × 0.5 = 0.048*
0	*1*	*0*	*0.036 × 0 = 0*
0	*1*	*1*	*0.144 × 0.2 = 0.0288*
1	*0*	*0*	*0 × 0 = 0*
1	*0*	*1*	*0 × 0.5 = 0*
1	*1*	*0*	*0 × 0 = 0*
1	*1*	*1*	*0 × 0.2 = 0*

Como se puede ver, el mensaje ha influido en el potencial del clique, no sólo incorporando evidencia, sino que también modificando los valores del potencial, pues el mensaje que recibe C_1 lleva información.

C_1 absorbe de C_3 : *el potencial del separador es*

$$\phi^*_{S_2}(x_2) = \sum_{x_4} \phi_{C_3}(x_2, x_4) \equiv 1$$

El paso de este mensaje no aporta ninguna información nueva.

Una vez terminado el proceso de recolección pasamos a la distribución, que seguiría el siguiente orden:

1. *C_1 le envía un mensaje de distribución a C_2.*

2. *C_2 al recibir el mensaje absorbe de C_1 y después le envía un mensaje de distribución a su único vecino, aparte de C_1, C_4.*

3. *C_4 recibe el mensaje de C_2 y absorbe de éste, y no envía mensajes a ningún nodo pues no tiene más vecino que C_2.*

4. *C_1 le envía un mensaje de distribución a C_3, pues ya ha terminado el mensaje que le envió a C_2.*

5. *C_3 al recibir el mensaje de C_1 absorbe de él y al no tener vecinos excepto C_1 no envía el mensaje.*

Hay tres absorciones :

C_2 absorbe de C_1: *el nuevo potencial del separador S_1 será*

$\phi^*_{S_1}(x_2, x_3) = \sum_{x_1} \phi_{C_1}(x_1, x_2, x_3) \equiv$

x_2	x_3	$\phi^*_{S_1}(x_2, x_3)$
0	0	0
0	1	0.048
1	0	0
1	1	0.0288

Por tanto $\phi^*_{C_2}(x_2, x_3, x_5) = \phi_{C_2}(x_2, x_3, x_5)\frac{\phi^*_{S_1}(x_2,x_3)}{\phi_{S_1}(x_2,x_3)}$, que resulta:

$\phi^*_{C_2}(x_2, x_3, x_5) \equiv$

x_2	x_3	x_5	$\phi^*_{C_2}(x_2, x_3, x_5)$
0	*0*	*0*	*0*
0	*0*	*1*	*0*
0	*1*	*0*	*0.5 × 0.096 = 0.048*
0	*1*	*1*	*0 × 0.096 = 0*
1	*0*	*0*	*0*
1	*0*	*1*	*0*
1	*1*	*0*	*0.2 × 0.144 = 0.0288*
1	*1*	*1*	*0 × 0.144 = 0*

C_4 absorbe de C_2 : *el potencial del separador S_3 es:*

$\phi^*_{S_3}(x_3) = \sum_{x_2, x_5} \phi_{C_2}(x_2, x_3, x_5) \equiv$

x_3	$\phi^*_{S_3}(x_3)$
0	*0*
1	*0.048 + 0.0288 = 0.0768*

Por tanto

$\phi^*_{C_4}(x_3, x_6) = \phi_{C_4}(x_3, x_6) \frac{\phi^*_{S_3}(x_3)}{\phi_{S_3}(x_3)} \equiv$

x_3	x_6	$\phi^*_{C_4}(x_3, x_6)$
0	*0*	*0*
0	*1*	*0*
1	*0*	*0.4 × 0.0768 = 0.03072*
1	*1*	*0.6 × 0.0768 = 0.04608*

C_3 absorbe de C_1 : *ahora el potencial que se actualiza es el de C_3 de la siguiente forma :*

$\phi^*_{S_2}(x_2) = \sum_{x_1, x_3} \phi_{C_1}(x_1, x_2, x_3) \equiv$

x_2	$\phi^*_{S_2}(x_2)$
0	*0.048*
1	*0.0288*

dado que $\phi_{S_2} \equiv 1$, calculamos

$\phi_{C_3}(x_2, x_4)\phi^*_{S_2}(x_2) \equiv$

x_2	x_4	$\phi^*_{C_3}(x_2, x_4)$
0	*0*	*0.3 × 0.048 = 0.0144*
0	*1*	*0.7 × 0.048 = 0.0336*
1	*0*	*0.2 × 0.0288 = 0.00576*
1	*1*	*0.8 × 0.0288 = 0.02304*

Así termina la etapa de distribución. Los potenciales están actualizados de forma que sólo hay que marginalizar sobre un potencial para conocer la distribución a posteriori de cualquier variable de la red.
Por ejemplo, para saber la distribución a posteriori de la variable X_4 sólo hay que marginalizar ϕ_{C_3}, esto es:

$$P[X_4 = 0] = \sum_{x_2} \phi_{C_3}(x_2, 0) = 0.0144 + 0.00576 = 0.02016$$
$$P[X_4 = 1] = \sum_{x_2} \phi_{C_3}(x_2, 1) = 0.0336 + 0.02304 = 0.05664$$

que normalizando queda

$$P[X_4 = 0] = \frac{0.02016}{0.02016 + 0.05664} = 0.2625$$
$$P[X_4 = 1] = \frac{0.05664}{0.02016 + 0.05664} = 0.7375$$

1.3.3. Algoritmo Shenoy-Shafer

Este algoritmo actua también sobre el árbol de cliques construido a partir de la red original, pero el paso de mensajes y la iniciación son diferentes (Shenoy y Shafer 1990).

En cada arista que une dos nodos, C_i y C_j tenemos un "buzón" para los mensajes que salen de C_i y llegan a C_j y otro para los que van en sentido contrario. Los mensajes que se almacenan en ambos buzones son potenciales definidos sobre el separador $S = C_i \cap C_j$. Empezamos con todos los buzones vacíos y una vez que un mensaje se haya almacenado en uno de ellos, se dice que está lleno.

A un nodo C_i se le permite enviar un mensaje a su nodo vecino C_j si y solo si todos

los buzones de los mensajes que llegan a C_i están llenos excepto el que viene de C_j a C_i.

La propagación se organiza en dos etapas; en la primera los mensajes se envían de las hojas a un nodo raíz seleccionado previamente, y en la segunda los mensajes se envían de la raíz a las hojas.

El mensaje que va de C_i a C_j se calcula como :

$$\phi_{C_i \to C_j} = \left\{ \phi_{C_i} \cdot \left(\prod_{C_k \in ne(C_i) \setminus \{C_j\}} \phi_{C_k \to C_i} \right) \right\}^{\downarrow S} \tag{1.2}$$

donde ϕ_{C_i} es el potencial original definido sobre C_i, $\phi_{C_k \to C_i}$ los mensajes en lo buzones de C_k a C_i y $ne(C_i)$ los nodos vecinos de C_i, S el separador correspondiente a esos dos cliques.

Nótese que un mensaje contiene la información que viene de una parte del árbol y se envía a la otra parte del árbol. Se puede demostrar (Shenoy y Shafer 1990) que siempre hay al menos un nodo que puede enviar un mensaje hasta que todos los buzones están llenos. Cuando el proceso de paso de mensajes termina, para cada nodo C_i en el árbol se tiene que la marginal de la distribución conjunta en las variables del nodo C_i es

$$p(\mathbf{x}_{C_i}, \mathbf{e}) = \phi_{C_i}(\mathbf{x}_{C_i}, \mathbf{e}) \cdot \left(\prod_{C_k \in ne(C_i)} \phi_{C_k \to C_i}(\mathbf{x}_{S_{ki}}, \mathbf{e}) \right) \tag{1.3}$$

donde S_{ki} es el separador correspondiente a los cliques C_k y C_i. Esta probabilidad es proporcional a la distribución condicional de las variables del nodo C_i dadas las observaciones $\mathbf{e}$. $p(x_i|\mathbf{e})$ se calcula marginalizando $p(\mathbf{x}_{C_i}, \mathbf{e})$ sobre X_i y normalizando.

Shenoy-Shafer($\mathcal{T}$)

ENTRADA: un árbol de cliques $\mathcal{T}$.
SALIDA: el árbol de cliques $\mathcal{T}$ tras la propagación.

1. Seleccionar un nodo raíz R.
2. Inicializamos los potenciales de los cliques.
3. Para cada $C \in ne(R)$,
   - **PropagarArriba(**R,C**)**
4. Para cada $C \in ne(R)$,
   - Calcular mensaje $\phi_{R \to C} = \left\{ \phi_R \cdot \left(\prod_{C_k \in ne(R) \setminus C} \phi_{C_k \to R} \right) \right\}^{\downarrow R \cap C}$.
   - **PropagarAbajo(**R,C**)**
5. DEVOLVER $\mathcal{T}$.

---

donde **PropagarArriba** es un algoritmo que envía mensajes de las hojas a la raíz, y **PropagarAbajo** envía mensajes de la raíz a las hojas. Ambos se detallan a continuación.

PropagarArriba(C_1,C_2**)**

---

1. Para cada $C \in ne(C_2) \setminus C_1$,
   - **PropagarArriba(**C_2,C**)**
2. Calcular mensaje $\phi_{C_2 \to C_1} = \left\{ \phi_{C_2} \cdot \left(\prod_{C_k \in ne(C_2) \setminus C_1} \phi_{C_k \to C_2} \right) \right\}^{\downarrow C_1 \cap C_2}$.

PropagarAbajo(C_1,C_2)

---

1. Para cada $C \in ne(C_2) \backslash C_1$,

   - Calcular mensaje $\phi_{C_2 \to C} = \left\{ \phi_{C_2} \cdot \left(\prod_{C_k \in ne(C_2) \backslash C_1} \phi_{C_k \to C_2} \right) \right\}^{\downarrow C \cap C_2}$.
   - **PropagarAbajo(C_2,C)**

Ejemplo 1.3 *Consideremos la misma red del ejemplo 1.2, con las mismas evidencias. Si a esta red le aplicamos el algortimo Shenoy-Shafer se obtiene:*

1. *Se selecciona como nodo raíz el* C_1.

2. *El conjunto de vecinos de* C_1 *es* $ne(C_1) = \{C_2, C_3\}$.

   - *Si se escoge primero a* C_2*: llamar a* **PropagarArriba(C_1,C_2)**:

     a) $ne(C_2) \setminus C_1 = C_4$, *luego se llama a* **PropagarArriba(C_2,C_4)**:

        1) $ne(C_4) \setminus C_2 = \{\emptyset\}$

        2) *Calcular el mensaje de* C_4 *a* C_2 : $\phi_{C_4 \to C_2}$

     b) *Calcula el mensaje de* C_2 *a* C_1 : $\phi_{C_2 \to C_1}$

   - *Siguiendo con el otro vecino de* C_1, C_3*: llamar a* **PropagarArriba(C_1,C_3)**:

     a) $ne(C_3) \setminus C_1 = \{\emptyset\}$.

     b) *Calcular el mensaje de* C_3 *a* C_1 : $\phi_{C_3 \to C_1}$

   Hay tres mensajes que calcular, que son:

Mensaje de C_4 a C_2:

$$\phi_{C_4 \to C_2}(\mathbf{x}_{S_3}) = \phi_{C_4 \to C_2}(x_3) = \sum_{x_6} \phi_{C_4}(x_3, x_6) = \delta_3(x_3, 1)$$

ya que el único mensaje que llega al clique C_4 es el que viene de C_2.

Mensaje de C_2 a C_1:

$$\phi_{C_2 \to C_1}(\mathbf{x}_{S_1}) = \phi_{C_2 \to C_1}(x_2, x_3) = \sum_{x_5} \phi_{C_2}(x_2, x_3, x_5)\phi_{C_4 \to C_2}(x_3)$$

pues el único mensaje distinto de la unidad que llega al clique C_2 es el que proviene de C_4 y que se acaba de calcular.

x_2	x_3	$\phi_{C_2 \to C_1}(x_2, x_3)$
0	*0*	*0 + 0 = 0*
0	*1*	*0.5 + 0 = 0.5*
1	*0*	*0 + 0 = 0*
1	*1*	*0.2 + 0 = 0.2*

Mensaje de C_3 a C_1:

$$\phi_{C_3 \to C_1}(\mathbf{x}_{S_2}) = \phi_{C_3 \to C_1}(x_2) = \sum_{x_4} \phi_{C_3}(x_2, x_4) \equiv 1$$

ya que el único mensaje que llega al clique C_3 es el que viene de C_1.

Así termina la etapa en la que se envían los mensajes de las hojas a la raíz. Se pasa pues a la etapa en la que se envían los mensajes de la raíz a las hojas:

3. *El conjunto de vecinos de C_1 es $ne(C_1) = \{C_2, C_3\}$.*

   - *Escogiendo C_2:*

     a) *Calcular el mensaje de C_1 a C_2.*

     b) *Llamar a* **PropagarAbajo(**C_1,C_2**)**: *como $ne(C_2) \setminus C_1 = \{C_4\}$*

        1) *Calcular el mensaje de C_2 a C_4*

        2) *Llamar a* **PropagarAbajo(**C_2,C_4**)**: *como $ne(C_4) \setminus C_2 = \{\emptyset\}$ se para ahí.*

- *Siguiendo con el otro vecino de C_1, C_3:*

  a) *Calcular el mensaje de C_1 a C_3.*

  b) *Llamar a* **PropagarAbajo(C_1,C_3)**: *como $ne(C_3) \setminus C_1 = \{\emptyset\}$ se para.*

Hay tres mensajes que calcular, que son:

Mensaje de C_1 a C_2:

$$\phi_{C_1 \to C_2}(\mathbf{x}_{S_1}) = \phi_{C_1 \to C_2}(x_2, x_3) = \sum_{x_1} \phi_{C_1}(x_1, x_2, x_3)\phi_{C_3 \to C_1}(x_2) =$$

$$= \sum_{x_1} \phi_{C_1}(x_1, x_2, x_3)$$

ya que el único mensaje que llega a C_1 que no sea desde C_2 es desde C_3, pero como se acaba de calcular es exactamente 1.

x_2	x_3	$\phi_{C_1 \to C_2}(x_2, x_3)$
0	*0*	*0*
0	*1*	*0.048*
1	*0*	*0*
1	*1*	*0.0288*

Mensaje de C_2 a C_4:

$$\phi_{C_2 \to C_4}(\mathbf{x}_{S_3}) = \phi_{C_2 \to C_4}(x_3) = \sum_{x_2, x_5} \phi_{C_2}(x_2, x_3, x_5)\phi_{C_1 \to C_2}(x_2, x_3)$$

al ser C_1 el único nodo distinto de C_4 que envía un mensaje a C_2.

$\phi_{C_2 \to C_4}(\mathbf{x}_{S_3}) \equiv$

x_3	$\phi_{C_2 \to C_4}(x_3)$
0	*0 + 0 + 0 + 0 = 0*
1	*0 + 0 + 0.024 + 0.00576= 0.02976*

Mensaje de C_1 a C_3:

$$\phi_{C_1 \to C_3}(\mathbf{x}_{S_2}) = \phi_{C_1 \to C_3}(x_2) = \sum_{x_1, x_3} \phi_{C_1}(x_1, x_2, x_3)\phi_{C_2 \to C_1}(x_2, x_3)$$

$$\phi_{C_1 \to C_3}(\mathbf{x}_{S_2}) \equiv$$

x_2	$\phi_{C_1 \to C_3}(x_2)$
0	*0 + 0.048 + 0 + 0 = 0.048*
1	*0 + 0.0288 + 0 + 0 = 0.0288*

Si se quiere calcular alguna distribución marginal sólo hay que aplicar la fórmula (1.3).

Por ejemplo, para calcular la marginal de la variable X_4:

$$P(X_4) = \sum_{x_2} \phi_{C_3}(x_2, x_4)\phi_{C_1 \to C_3}(x_2)$$

x_2	x_4	$\phi_{C_3}\phi_{C_1 \to C_3}$
0	*0*	*0.3 × 0.048 = 0.0144*
0	*1*	*0.7 × 0.048 = 0.0336*
1	*0*	*0.2 × 0.0288 = 0.00576*
1	*1*	*0.8 × 0.0288 = 0.02304*

luego

$$P[X_4 = 0] = 0.0144 + 0.00576 = 0.02016,$$
$$P[X_4 = 1] = 0.0336 + 0.02304 = 0.05664.$$

que normalizando queda

$$P[X_4 = 0] = \frac{0.02016}{0.02016 + 0.05664} = 0.2625,$$
$$P[X_4 = 1] = \frac{0.05664}{0.02016 + 0.05664} = 0.7375.$$

que como vemos es exactamente lo que salía con el algoritmo HUGIN.

1.4. Propagación de probabilidades. Algoritmos aproximados

Los algoritmos mencionados en las secciones anteriores son exactos, es decir, obtienen la aunténtica distribución a posteriori de las variables de interés. Sin embargo

en redes suficientemente complicadas, el empleo de estos algoritmos no es factible, debido a que las tablas de probabilidad que representan los potenciales que surgen en la propagación pueden tener un tamaño muy elevado. Para solucionar este problema aparecen los denominados *métodos aproximados*, que no calculan la distibución a posteriori exacta si no una aproximación a ella. Estos métodos se pueden agrupar en dos categorías: métodos de simulación y deterministas:

- Simulación: están basados en la obtención de una muestra de las variables de la red, fundamentalmente por métodos de Monte Carlo, que después se usará para estimar las distribuciones marginales (Dagum y Luby 1993; Fung y Chang 1990; Hernández, Moral, y Salmerón 1996; Salmerón, Cano, y Moral 2000; Shachter y Peot 1990).
- Deterministas: se han desarrollado diversas ideas, como son reemplazar valores bajos de probabilidad por ceros para reducir costes de almacenamiento de los datos (Jensen y Andersen 1990b); simplificar la estructura del modelo eliminando dependencias débiles (Kjærulff 1994); y enumerar las configuraciones de la variables con más alta probabilidad con el fin de obtener aproximaciones de la distribución marginal a posteriori (Poole 1993; Santos y Shimony 1994).

A continuación destacaremos el algoritmo presentado en (Pearl 1987), basado en Monte Carlo por Cadenas de Markov (MCMC) (Gilks, Richardson, y Spiegelhalter 1996), que será utilizado en capítulos posteriores.

1.4.1. Algoritmo MCMC de Pearl

Este método se basa en la obtención de una muestra representativa de las variables de la red en la que los individuos que la forman no son independientes, sino que cumplen la propiedad de Markov.

Comienza con una configuración inicial de las variables de la red, y a partir de ella vamos simulando una por una todas las variables no observadas. Cada variable X_i se simula con la distribución condicionada de ésta dada la configuración actual

que, por la propiedad de d-separación (definición 1.7) es proporcional a la distribución condicionada a su envolvente de Markov:

Definición 1.26 *La **envolvente de Markov** de una variable X_i en una red bayesiana es el conjunto formado por los padres, los hijos, y los padres de los hijos de X_i, excepto X_i. La denotaremos por $\mathbf{W}_{X_i}$.*

La distribución condicionada de X_i a su envolvente de Markov, $P(X_i|\mathbf{W}_{X_i})$, es

$$P(X_i = x_i|\mathbf{W}_{X_i} = \mathbf{w}_{x_i}) \propto p(x_i|\boldsymbol{\pi}_i) \prod_{k \in H_i} p_k(x_k|\boldsymbol{\pi}_k) \quad \forall x_i \in \Omega_{X_i}$$

Para simular un valor a partir de esta distribución, se restringe a la configuración actual de las variables de su dominio excepto X_i, obteniendo un potencial definido únicamente sobre X_i. Los valores se simulan usando el método de la *transformada inversa* (Rubinstein 1981).

Una vez obtenida la muestra, se estiman las probabilidades $p(x_i|\mathbf{e})$, mediante la proporción de elementos de la muestra en los que $X_i = x_i$.

El algoritmo quedaría:

MCMC($\mathcal{G}$)

ENTRADA: una red bayesiana $\mathcal{G}$.
SALIDA: las distribuciones marginales de cada variable.

1. Seleccionar una primera configuración, $x^{(0)}$ de las variables, con probabilidad mayor que cero.
2. Desde $j = 1$ hasta m
   - Para cada variable X_i no observada:

a) Calcular $P(X_i|\mathbf{W}_{X_i} = \mathbf{w}_{x_i}^{(j-1)})$, donde $\mathbf{w}_{x_i}^{(j-1)}$ es la proyección de $x^{(j-1)}$ sobre la envolvente de Markov de la variable X_i.

b) Simulamos un valor $x_i^{(j)}$ para X_i a partir de ella.

3. Obtener las distribuciones marginales $p(x_i|\mathbf{e})$.

4. DEVOLVER $\{p(X_i|\mathbf{E}) \quad \forall i\}$.

---

donde m es el tamaño de la muestra.

Ejemplo 1.4 *Siguiendo con la misma red de los ejemplos 1.2 y 1.3, apliquemos este algoritmo para obtener una aproximación de la distribución marginal de la variable X_4, considerando las mismas evidencias, $X_1 = 0, X_3 = 1, X_5 = 0$.*

Supongamos que tenemos la siguiente configuración inicial:

$$\mathbf{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, x_3^{(0)}, x_4^{(0)}, x_5^{(0)}, x_6^{(0)}) = (0, 1, 1, 0, 0, 0)$$

Las distribuciones condicionadas a su manto de Markov de las variables no observadas, X_2, X_4 y X_6, son las siguientes:

$X_2|\mathbf{W}_{X_2}$ *: $P(x_2|\mathbf{W}_{x_2}) = p(x_2|x_1)p(x_4|x_2)p(x_5|x_2, x_3)$ este potencial está definido para todas las variables excepto X_6, es decir, su tabla de probabilidad tendría $2^5 = 32$ valores, pero debido a las observaciones, se puede considerar definido únicamente sobre las variables X_2 y X_4:*

$$P(x_2|\mathbf{W}_{x_2}) = p(x_2|x_1 = 0)p(x_4|x_2)p(x_5 = 0|x_2, x_3 = 1)$$

x_2	x_4	$P(x_2\|\mathbf{W}_{x_2})$
0	*0*	*0.4 × 0.3 × 0.5 = 0.06*
0	*1*	*0.4 × 0.7 × 0.5 = 0.14*
1	*0*	*0.6 × 0.2 × 0.2 = 0.024*
1	*1*	*0.6 × 0.8 × 0.2 = 0.096*

$X_4|\mathbf{W}_{X_4}$: $P(x_4|\mathbf{W}_{x_4}) = p(x_4|x_2)$

x_2	x_4	$P(x_4|\mathbf{W}_{x_4})$
0	*0*	*0.3*
0	*1*	*0.7*
1	*0*	*0.2*
1	*1*	*0.8*

$X_6|\mathbf{W}_{X_6}$: $P(x_6|\mathbf{W}_{x_6}) = p(x_6|x_3)$

como X_3 es una variable observada realmente el potencial se puede considerar definido únicamente sobre X_6:

X_6	$P(x_6|\mathbf{W}_{x_6})$
0	*0.4*
1	*0.6*

A continuación se obtiene una muestra. Para ello se simulan las variables en el siguiente orden: $\{X_2, X_4, X_6\}$

Simular X_2: *el potencial que se utiliza para simular X_2 está definido sobre X_2 y X_4, luego se usa el valor que toma la variable X_4 en la última configuración simulada, en este caso la primera: $X_4 = 0$. Así el potencial $P(x_2|\mathbf{W}_{x_2})$ se convierte en*

$$P[X_2 = 0] = 0.06$$

$$P[X_2 = 1] = 0.024$$

pero para poder simular es necesario que esté normalizado:

$$P[X_2 = 0] = 0.7143$$

$$P[X_2 = 1] = 0.2857$$

Usando el método de inversión para simular un valor de la variable: supongamos que tenemos un número aleatorio 0.81449*, entonces el valor simulado sería* $X_2 = 1$.

Simular X_4: *ahora el valor que se usa para simular es el de la variable X_2, $X_2 = 1$, y por tanto el potencial con el que se simula es:*

$$P[X_4 = 0] = 0.2$$

$$P[X_4 = 1] = 0.8$$

Este potencial ya está normalizado, luego, si el número aleatorio es 0.84969, *entonces el valor simulado sería $X_4 = 1$*

Simular X_6: *el potencial que se usa para simular X_6 sólo depende de X_6, así pues, si el número aleatorio es* 0.58476, *entonces el valor simulado para la variable es $X_6 = 1$.*

Por tanto, el primer valor de la muestra será: $\mathbf{x}^{(1)} = (0, 1, 1, 1, 1, 0, 1)$.

Repitiendo estos pasos se obtiene la siguiente muestra de tamaño 10*:*

u_1, u_2, u_3	$\mathbf{x}^{(i)}$	$(x_1, x_2, x_3, x_4, x_5, x_6)$
$0.8145; 0.8497; 0.5848$	$\mathbf{x}^{(1)}$	$(0, 1, 1, 1, 0, 1)$
$0.4124; 0.6462; 0.7747$	$\mathbf{x}^{(2)}$	$(0, 0, 1, 1, 0, 1)$
$0.0577; 0.1107; 0.9709$	$\mathbf{x}^{(3)}$	$(0, 0, 1, 0, 0, 1)$
$0.3360; 0.8228; 0.6190$	$\mathbf{x}^{(4)}$	$(0, 0, 1, 1, 0, 1)$
$0.3841; 0.5193; 0.8624$	$\mathbf{x}^{(5)}$	$(0, 0, 1, 1, 0, 1)$
$0.2363; 0.8561; 0.2762$	$\mathbf{x}^{(6)}$	$(0, 0, 1, 1, 0, 0)$
$0.8621; 0.1340; 0.3334$	$\mathbf{x}^{(7)}$	$(0, 1, 1, 0, 0, 0)$
$0.3319; 0.9122; 0.8559$	$\mathbf{x}^{(8)}$	$(0, 0, 1, 1, 0, 1)$
$0.5453; 0.9519; 0.1929$	$\mathbf{x}^{(9)}$	$(0, 0, 1, 1, 0, 0)$
$0.6692; 0.9865; 0.0153$	$\mathbf{x}^{(10)}$	$(0, 1, 1, 1, 0, 0)$

donde $u_1; u_2; u_3$ son los números aleatorios usados para generar X_2, X_4 y X_6 respectivamente.

Para X_4 se obtiene la siguiente aproximación de la distribución a posteriori:

$$P[X_4 = 0] = \frac{2}{10} = 0.2,$$
$$P[X_4 = 1] = \frac{8}{10} = 0.8.$$

1.5. Aprendizaje

Una vez que ya sabemos operar con la estructura de una red bayesiana, surge la necesidad de obtener dicha estructura a partir de una base de datos.

En este aprendizaje se pueden diferenciar dos cuestiones, obtención de la estructura y obtención de los parámetros, que no son tareas independientes, pues para estimar los parámetros de las probabilidades necesitamos conocer los padres de cada variable, es decir la estructura, y viceversa. Podemos encontrar una descripción mas detallada de este proceso general de aprendizaje en (de Campos 1998; Castillo, Gutiérrez, y Hadi 1996).

1.5.1. Aprendizaje paramétrico

Suponiendo que conocemos la estructura de la red, el correspondiente grafo acíclico $\mathcal{G}$, y que disponemos de una base de datos de las variables asociadas a los nodos de $\mathcal{G}$, el **aprendizaje paramétrico** consiste en obtener los parámetros de todas las distribuciones condicionadas $P(X_i|\mathbf{\Pi}_i)$ necesarias para definir la distribución de probabilidad conjunta de las variables en la red.

La forma más habitual de estimar dichos parámetros es mediante sus correspondientes estimadores máximo verosímiles: sean $n(\boldsymbol{\pi}_i)$ y $n(x_i, \boldsymbol{\pi}_i)$ el número de veces de entre los datos en que $\mathbf{\Pi}_i = \boldsymbol{\pi}_i$ y el número de veces en que $X_i = x_i$ y $\mathbf{\Pi}_i = \boldsymbol{\pi}_i$ simultáneamente, respectivamente.

La probabilidad a calcular se obtiene como

$$P(x_i|\boldsymbol{\pi}_i) = \frac{n(x_i, \boldsymbol{\pi}_i)}{n(\boldsymbol{\pi}_i)} \ , \tag{1.4}$$

Esta forma de obtener las probabilidades presenta dos inconvenientes fundamentales:

- Se necesita una muestra de tamaño grande, ya que si un valor no aparece entre los datos observados, su probabilidad será estimada como cero, o incluso para algunas configuraciones de los padres, la fórmula (1.4) puede no estar definida.

- Este estimador tiende a sobreajustar los datos, dando lugar en el caso de muestras de poco tamaño a predicciones pobres.

Para paliar estos problemas, surgen otros estimadores. Uno de ellos está basado en:

Ley de sucesión de Laplace: (Good 1965) *Si en una muestra de N casos encontramos k casos que verifican una determinada propiedad Q, entonces la probabilidad de que el siguiente caso que observemos muestre la misma propiedad es* $\frac{k+1}{N+|Q|}$, *donde* $|Q|$ *representa el número de posibles alternativas de la propiedad Q.*

En este caso, si Q representa que X_i tome el valor x_i, el estimador sería:

$$P(x_i|\boldsymbol{\pi}_i) = \frac{n(x_i, \boldsymbol{\pi}_i) + 1}{n(\boldsymbol{\pi}_i) + |\Omega_{X_i}|} \ , \tag{1.5}$$

donde $|\Omega_{X_i}|$ es el número de posibles valores de X_i.

La fórmula (1.5) siempre está definida, y presenta las siguientes características:

1. Si la muestra es pequeña, da como resultado una distribución parecida a la uniforme, más parecida cuanto más pequeña sea la muestra. En el caso concreto de que una configuración de los padres no tenga ninguna observación en la muestra es exactamente la distribución uniforme.

2. La ecuación (1.5) da como resultado un estimador que tiende al estimador máximo verosímil cuando la muestra es lo suficientemente grande, dado que entonces los valores 1 y $|\Omega_{X_i}|$ es insignificante comparado con las frecuencias.

1.5.2. Aprendizaje estructural

Planteamos aquí la necesidad de aprender la estructura de la red. Los distintos algoritmos pueden clasificarse en dos grandes grupos fundamentalmente, de acuerdo con su enfoque:

- Los que tratan de obtener las relaciones de las variables mediante tests de independencia.
- Los que realizan una búsqueda en el espacio de todas las redes posibles, de acuerdo con una medida de *calidad* o *métrica*.

1.5.2.1. Algoritmos basados en tests de independencias

Un algoritmo clásico dentro de los de este tipo es el **algoritmo PC** (Spirtes, Glymour, y Scheines 1991; Spirtes, Glymour, y Scheines 1993). Este algoritmo comienza con el grafo completo no dirigido correspondiente a todas las variables de la base de datos, para luego ir reduciéndolo. Primero se eliminan las aristas que unen nodos que verifican un independencia condicional de orden cero, independencias simples, luego las aristas que unen nodos que verifican una independencia condicional de orden uno, independencias de dos variables dada una tercera, y así sucesivamente hasta un valor m prefijado. Dadas dos variables, X_i y X_j, el conjunto de variables a las que condicionar se selecciona de entre los conjuntos de nodos adyacentes a X_i o a X_j.

1.5.2.2. Algoritmos de búsqueda

El objetivo de este tipo de algoritmos es obtener un grafo dirigido acíclico que represente la base de datos, a partir de:

- Una medida de la calidad, f que permita seleccionar el mejor grafo entre varios.
- Una heurística de búsqueda en el espacio de todos los grafos dirigidos acíclicos que encuentre el mejor grafo basándose para ello en la medida de calidad f.

Se han definido muchas medidas a la hora de medir la calidad de un grafo que representa una base de datos (Castillo, Gutiérrez, y Hadi 1996), que se basan en la filosofía de la estadística bayesiana, de las cuales podemos destacar:

- Métrica **K2** (Cooper y Herskovits 1992): la calidad de un grafo $\mathcal{G}$ con respecto a una base de datos D se define como

$$f(\mathcal{G}:D) = logP(\mathcal{G}) + \sum_{i=1}^{n}\left[\sum_{k=1}^{s_i}\left[log\frac{\Gamma(r_i)}{\Gamma(N_{ij}+r_i)} + \sum_{j=1}^{r_i} log\Gamma(N_{ijk}+1)\right]\right] \ ,$$

  donde:

  - r_i es el número de casos de la variable X_i.
  - x_{ik} es el k-ésimo valor de X_i.
  - $\boldsymbol{\pi}_{ij}$ es el j-ésimo valor de $\boldsymbol{\Pi}_i$.
  - $s_i = \prod_{x_j \in \boldsymbol{\Pi}_i} r_j$, el número de casos de $\boldsymbol{\Pi}_i$.
  - N_{ijk} es el número de casos de la base de datos tales que $X_i = x_{ik}$ y $\boldsymbol{\Pi}_i = \boldsymbol{\pi}_{ij}$.
  - $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$.

- Métrica **BIC** (Heckerman, Geiger, y Chickering 1995):

$$f(\mathcal{G}:D) = logP(\mathcal{G}) + \sum_{i=1}^{n}\sum_{j=1}^{r_i}\sum_{k=1}^{s_i} N_{ijk} log\frac{N_{ijk}}{N_{ik}} - \frac{1}{2}C(\mathcal{G})logN \ ,$$

  donde ahora $C(\mathcal{G}) = \sum_{i=1}^{n}(r_i - 1)s_i$ es una medida de complejidad de la red y N el número de registros de la base de datos.

Una propiedad deseable de cualquier medida de calidad, es que verifique la siguiente descomposición:

$$f(\mathcal{G}:D) = \sum_{i=1}^{n} f(x_i|\boldsymbol{\pi}_i : N_{x_i,\boldsymbol{\pi}_i}). \tag{1.6}$$

Sea han propuesto a su vez diferentes heurísticas de búsqueda, cuyo objetivo es encontrar el grafo $\mathcal{G}$ tal que maximice la correspondiente medida de calidad. Entre todas ellas podemos destacar el **algoritmo K2** (Cooper y Herskovits 1992). Este algoritmo usa la métrica K2, y aprovecha que verifica (1.6) para trabajar de forma local con cada variable X_i, y, partiendo de un conjunto vacío de padres de la variable, va añadiendo padres para X_i conforme mejora la calidad de la red obtenida. El orden en que se escogen las variables influye en el resultado, por tanto es necesario establecer un orden previo.

Capítulo 2

Discretización

2.1. Introducción

Los algoritmos descritos en el capítulo anterior generalmente operan sobre variables discretas con un número finito de posibles valores. La razón es que para este tipo de variables los potenciales probabilísticos pueden representarse mediante estructuras de datos (por ejemplo las tablas de probabilidad) sobre las que las operaciones básicas para la propagación (restricción, marginalización y combinación) están perfectamente definidas. Sin embargo, en realidad los algoritmos están bien definidos para cualquier tipo de variables cuyos potenciales asociados permitan las tres operaciones básicas (en el caso de MCMC una cuarta operación es nececsaria: la simulación de un valor a partir de un potencial).

En el caso de variables continuas, las distribuciones se representan en términos de funciones de densidad sobre las que las operaciones básicas están definidas. Sin embargo, implementar los algoritmos anteriores para variables continuas puede presentar algunos problemas, que principalmente proceden de la dificultad de encontrar una estructura de datos común para distintos tipos posibles de densidades o incluso para manejar variables discretas y continuas simultáneamente. Por otro lado, la operación de marginalización, que se corresponde con la integración en el caso continuo, puede presentar problemas de cálculo.

Soluciones satisfactorias se han obtenido en casos concretos donde existe un modelo probabilístico para todas las variables de la red. Entre los trabajos más destacados en este sentido podemos citar los basados en el modelo condicional gaussiano (Lauritzen 1992; Lauritzen y Wermuth 1989; Olesen 1993) y en variables tipo Beta (Castillo y Gutiérrez 1998).

El enfoque más general, que permite el empleo de los algoritmos de propagación con cualquier tipo de variable consiste en discretizar las continuas y tratarlas como si fueran discretas. El problema de la discretización ha sido ampliamente tratado en la literatura, tanto desde un punto de vista general (Dougherty, Kohavi, y Sahami 1995; Christofides, Tanyi, Whobrey, y Christofides 1999) como orientado a inferencia en redes bayesianas *híbridas*, que se definen como sigue:

Definición 2.1 *Una red bayesiana se dice* ***híbrida*** *si contiene tanto variables aleatorias discretas como continuas.*

Después del proceso de discretización, una red bayesiana híbrida se trata igual que una red bayesiana desde el punto de vista de la propagación.

Por discretización se entiende una partición del rango de una variable continua. Considerando, sin pérdida de generalidad, el rango de las variables igual al hipercubo unidad $\Omega = [0,1]^n$, se puede definir una discretización como sigue:

Definición 2.2 *Una* ***discretización*** *D de un hipercubo $\Omega = [0,1]^n$ es una función constante a trozos $i_D(x_1, \ldots, x_n)$ de Ω en un conjunto finito de enteros de 1 a m. La función define un conjunto de subregiones $\{W_i : i = 1, \ldots, m\}$ que forman una partición de Ω.*

Si una función de densidad $f_D(x_1, \ldots, x_n)$ es constante en cada una de las subregiones W_i, la llamamos una *función discretizada* sobre D. El empleo de la función discretizada en lugar de la original lleva asociado un error que puede medirse por la divergencia de Kullback-Leibler (Kullback 1978).

Definición 2.3 *Se define la entropía relativa o divergencia de Kullback-Leibler (KL) entre dos funciones de densidad $f(x)$ y $g(x)$ como*

$$D(f||g) = \int_S f(x)\ log\frac{f(x)}{g(x)}dx\ .$$

A la hora de discretizar las variables de una red, una primera alternativa podría ser discretizar cada variable por separado, de manera que si tenemos una función definida sobre varias variables, su dominio sería el producto cartesiano de las variables discretizadas sobre las que la función está definida. El problema de este enfoque es que no tiene en cuenta la función, de manera que puede ser que divida innecesariamente zonas donde la función es constante y, por el contrario, no tome suficientes divisiones donde la función es más irregular. Cuando la discretización se realiza teniendo en cuenta todas las variables para las que está definida la función simultáneamente, se habla de *discretización no uniforme*. En el resto del capítulo vamos a describir el método de discretización no uniforme de Kozlov y Koller (1997).

2.2. Discretización no uniforme

El objetivo del método propuesto en (Kozlov y Koller 1997) es sólo discretizar una función manteniendo el número de subregiones de discretización bajo tratando a la vez de preservar la mayor parte de la información contenida en la función original.

Antes de discretizar el dominio de una función multidimensional, es necesario definir cuánto valdrá la función discretizada en cada una de las subregiones que se obtengan en la discretización. El valor óptimo lo proporciona el siguiente teorema, cuya demostración se puede encontrar en (Cover y Thomas 1991).

Teorema 2.1 *Dada una discretización D del espacio Ω, la distancia KL mínima entre una función de densidad $f(x_1, \ldots, x_n)$, y una función discretizada $f_D(x_1, \ldots, x_n)$ se alcanza con aquella función discretizada que es constante en cada una de las subregiones W_i e igual a la media de la de la función $f(x_1, \ldots, x_n)$ en la correspondiente subregión W_i.*

Para la partición del dominio, el criterio de optimalidad es escoger aquella discretización que minimice el error en términos de entropía relativa.

De acuerdo con este criterio de optimalidad, para dividir un hipercubo Ω en m subregiones, una forma de hacerlo podría ser obtener todas las posibles particiones de Ω en m subregiones y quedarnos con aquélla que minimice la distancia KL, asignando de forma óptima los valores dentro de cada subregión. Este proceso, en el caso multidimensional, es muy costoso.

Kozlov y Koller (1997) proponen una técnica divide y vencerás para obtener la partición del dominio. Para ello introducen el concepto de *árbol BSP* (Binary Split Partition), que es una estructura de datos recursiva que representa la descomposición binaria de una función multidimensional. Cada nodo del árbol representa una región del espacio, y puede tener dos hijos, cada uno de los cuales representa la mitad del espacio del padre. La partición va desde la raíz del árbol, que representa todo el dominio de la función, hasta las hojas, que son las que contienen los valores de la función en cada subregión. Todas las divisiones del espacio se consideran binarias, es decir, en dos mitades iguales definidas por un plano ortogonal a uno de los ejes.

Ejemplo 2.1 *A continuación mostramos un árbol BSP que representa a la siguiente función discretizada con dominio* $[0,1]^2$*:*

$$f(x,y) = \begin{cases} a & si\ (x,y) \in [0,\frac{1}{2}] \times (\frac{1}{2},1] \\ b & si\ (x,y) \in [0,\frac{1}{2}] \times [0,\frac{1}{2}] \\ c & si\ (x,y) \in (\frac{1}{2},\frac{3}{4}] \times [0,\frac{1}{2}] \\ d & si\ (x,y) \in (\frac{3}{4},1] \times [0,\frac{1}{4}] \\ e & si\ (x,y) \in (\frac{3}{4},1] \times (\frac{1}{4},\frac{1}{2}] \\ f & si\ (x,y) \in (\frac{1}{2},\frac{3}{4}] \times (\frac{1}{2},1] \\ g & si\ (x,y) \in (\frac{3}{4},\frac{7}{8}] \times (\frac{1}{2},1] \\ h & si\ (x,y) \in (\frac{7}{8},1] \times (\frac{1}{2},1] \end{cases} \tag{2.1}$$

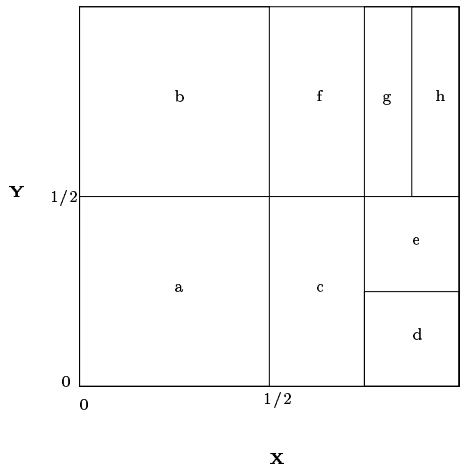


Figura 2.1: Discretización del dominio de la función (2.1).

En este caso los valores a, b, c, d, e, f, g y h son los valores óptimos que tomaría la función discretizada en esas subregiones, es decir, la media de la función en la correspondiente subregión.

La discretización del dominio de la función (2.1) puede verse gráficamente en la figura 2.1, y su representación mediante árbol BSP en la figura 2.2.

Podemos observar que cada X y cada Y corresponden con una partición de la subregión correspondiente perpendicular a X o a Y respectivamente. El hijo izquierdo de cada nodo representa a la primera mitad de la subregión y el hijo derecho a la segunda. En este caso, si el padre es Y, el primer hijo corresponde a la parte inferior de la subregión y la segunda a la superior. Si el nodo es X, el primer hijo será la parte izquierda de la subregión y el segundo la parte derecha.

En este ejemplo en concreto, empezamos con un nodo X, eso quiere decir que partimos el espacio inicial en dos mitades por un eje perpendicular a X. Ahora, el hijo izquierdo se corresponde con la mitad izquierda del rango de esta variable, y el segundo con la mitad derecha. Como vemos la parte izquierda se divide por el eje Y, y eso nos da dos mitades, en cada una de las cuales se toma un valor, a en la inferior y b en la superior. La parte derecha se divide a su vez otra vez por el eje X, y cada una de estas dos mitades por el eje Y, y así sucesivamente.

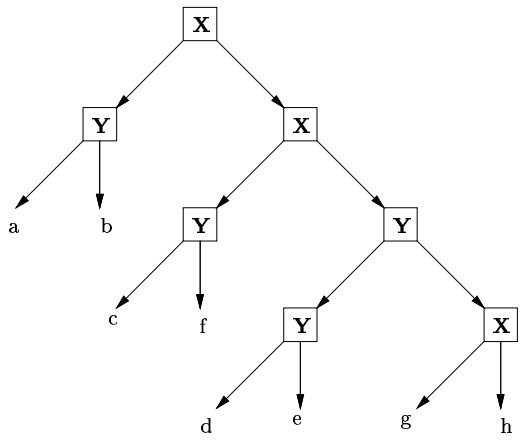


Figura 2.2: Árbol BSP correspondiente a la discretización de la figura 2.1.

Como se puede observar, la discretización es más fina en la parte derecha del plano XY, donde no nos paramos en la primera partición y seguimos dividiendo el espacio. Esto puede ser útil para representar una función con mayores irregularidades en la parte derecha.

Sobre el árbol BSP se va construyendo recursivamente la discretización. Para ello se necesita alguna regla que indique qué hoja partir y en qué dirección. La regla usada por Kozlov y Koller (1997) está basada en la distancia KL entre la función f y su discretización f_D, que en cada subregión W_i se corresponde con la función media $\bar{f}$. En realidad, debido a la complejidad del cálculo de la distancia KL en el caso general, los autores proponen usar una cota superior de la misma, dada por:

$$\int_{W_i} f(x) \log \frac{f(x)}{\bar{f}(x)} d\Omega \leq \left[\frac{f_{max} - \bar{f}}{f_{max} - f_{min}} f_{min} \log \frac{f_{min}}{\bar{f}} + \frac{\bar{f} - f_{min}}{f_{max} - f_{min}} f_{max} \log \frac{f_{max}}{\bar{f}} \right] |W_i| \ ,$$

donde $|W_i|$ es el volumen de la subregión W_i, $\bar{f}, f_{min}$ *y* f_{max} son la media, el máximo y el mínimo de la función en W_i, que pueden ser calculados mediante simulación por Monte Carlo.

Cada nodo de un árbol BSP representa una subregión del espacio, a la que o bien se divide en dos o bien se le asigna un valor, que será la media de la función en esa

subregión. Las hojas se van guardando en una cola, junto con su estimación del error. Aquella hoja que tenga mayor error será la siguiente en ser dividida en otras dos hojas (mitades), que a su vez serán añadidas a la cola. Este proceso empieza con una sola hoja, que representará todo el dominio de la función, y a partir de ella mediante divisiones binarias se va creando el árbol BSP.

Para controlar la precisión de nuestra discretización, se guarda en todo momento la suma de todas las estimaciones de los errores cometidos por todas las hojas de la cola. El proceso de discretización para cuando ésta suma de errores esté por debajo de algún valor predeterminado o bien cuando se alcance un número máximo de hojas támbien predeterminado.

Respecto a la decisión sobre la dirección en la que dividir el espacio, de nuevo la solución óptima sería obtener el descenso en la entropía relativa debido a las divisiones en todas las direcciones posibles y elegir la dirección de mayor descenso. Sin embargo, hacer esto de forma exacta requeriría el cálculo integrales multidimensionales, que no siempre sería posible resolver de forma exacta. La solución adoptada en (Kozlov y Koller 1997) consiste en muestrear varios puntos de la subregión y escoger la dirección en la que la función cambia más, es decir, el eje de coordenadas en el que la razón f_{max}/f_{min} sea mayor.

Después de definir el proceso de discretización, (Kozlov y Koller) adaptan el algoritmo HUGIN para la obtención de la marginal sobre una variable objetivo cuando intervienen variables continuas en la red. En este caso, sólo se lleva a cabo la fase de recolección de información, y el nodo raíz ha de ser tal que contenga a la variable objetivo. Después de realizar el envío de mensajes, la marginal sobre la variable objetivo se obtiene marginalizando el potencial del clique raíz.

La discretización se introduce modificando ligeramente la definición de absorción del algoritmo HUGIN:

Definición 2.4 *Dados dos cliques adyacentes C_i y C_j con separador S, se dice que C_i*

absorbe de C_j si las nuevas funciones de los cliques asociados pasan a ser :

$$\begin{aligned}
\phi^*_S(x_S) &= \phi^{\downarrow S}_{C_j}(x_{C_j}) \\
\phi^*_{C_i}(x_{C_i}) &= \phi_{C_i}(x_{C_i})\frac{\phi^*_S(x_S)}{\phi_S(x_S)} \\
\phi^*_{C_i}(x_{C_i}) &= \textbf{Discretizar}(\phi^*_{C_i}(x_{C_i}))
\end{aligned}$$

Así, el algoritmo de propagación se puede enunciar como:

PROPAGACIóN($\mathcal{G}$,X_q)

ENTRADA: una red bayesiana $\mathcal{G}$ y la variable objetivo X_q.
SALIDA: la distribución marginal a posteriori de la variable objetivo dada la evidencia $\phi_q(x_q)$.

1. Construir el árbol de cliques $\mathcal{T}$ asociado a la red $\mathcal{G}$.
2. Calcular las funciones asociadas a cada clique del árbol.
3. Incorporar las evidencias.
4. Seleccionar un clique R como raíz, que contenga a la variable objetivo X_q.
5. Enviar un mensaje de recolección a partir de R, siendo la recolección exactamente la misma que en el capítulo anterior.
6. Marginalizar el clique raíz R sobre la variable objetivo, para obtener su distribución marginal a posteriori $\phi_q(x_q)$.
7. DEVUELVE $\phi_q(x_q)$.

En realidad, la redefinición de la absorción hace que cada clique antes de enviar un mensaje, discretice la función resultante, con lo que se obtiene una función que se puede representar mediante una tabla de probabilidad. Nótese que primero se discretiza y luego se marginaliza sobre el separador, evitándo tener que integrar la función continua.

Tal y como está enunciado el algoritmo, la evidencia se incorpora, al principio, y ya no se tiene más en cuenta. En la siguiente sección veremos como podemos sacar partido de ella para mejorar nuestro proceso de discretización. Sin embargo, Kozlov y Koller (1997) muestran que ésta puede influir en el proceso de discretización, tal y como se describe en la siguiente sección.

2.3. Discretización dinámica

Para medir el impacto de la probabilidad de la evidencia en la discretización, se mide la distancia KL entre la función de distribución conjunta de todas las variables no observadas y de la evidencia, $p(\mathbf{x}, \mathbf{e})$ y la discretización de dicha distribución de probabilidad conjunta, $f_D(\mathbf{x}, \mathbf{e})$:

$$D(p(\mathbf{x}, \mathbf{e})||f_D(\mathbf{x}, \mathbf{e})) = \iint p(\mathbf{x}, \mathbf{e}) \log \frac{p(\mathbf{x}, \mathbf{e})}{f_D(\mathbf{x}, \mathbf{e})} d\mathbf{x} d\mathbf{e}$$

y como $p(\mathbf{x}, \mathbf{e}) = p(\mathbf{x}|\mathbf{e})p(\mathbf{e})$, al igual que con $f_D(\mathbf{x}, \mathbf{e})$, se puede escribir

$$D(p(\mathbf{x}, \mathbf{e})||f_D(\mathbf{x}, \mathbf{e})) = \iint p(\mathbf{e})p(\mathbf{x}|\mathbf{e}) \log \frac{p(\mathbf{e})p(\mathbf{x}|\mathbf{e})}{f_D(\mathbf{e})f_D(\mathbf{x}, \mathbf{e})} d\mathbf{x} d\mathbf{e}$$

$$= \iint p(\mathbf{e})p(\mathbf{x}|\mathbf{e}) \log \frac{p(\mathbf{e})}{f_D(\mathbf{e})} d\mathbf{x} d\mathbf{e} + \iint p(\mathbf{e})p(\mathbf{x}|\mathbf{e}) \log \frac{p(\mathbf{x}|\mathbf{e})}{f_D(\mathbf{x}|\mathbf{e})} d\mathbf{x} d\mathbf{e}$$

$$= \int p(\mathbf{e}) \left(\int p(\mathbf{x}|\mathbf{e}) \log \frac{p(\mathbf{e})}{f_D(\mathbf{e})} d\mathbf{x} \right) d\mathbf{e} + \int p(\mathbf{e}) \int p(\mathbf{x}|\mathbf{e}) \log \frac{p(\mathbf{x}|\mathbf{e})}{f_D(\mathbf{x}|\mathbf{e})} d\mathbf{x} d\mathbf{e}$$

$$= \int p(\mathbf{e}) \log \frac{p(\mathbf{e})}{f_D(\mathbf{e})} \left(\int p(\mathbf{x}|\mathbf{e}) d\mathbf{x} \right) d\mathbf{e} + D(p(\mathbf{x}|\mathbf{e})||f_D(\mathbf{x}|\mathbf{e}))$$

$= D(p(\mathbf{e})||f_D(\mathbf{e})) + D(p(\mathbf{x}|\mathbf{e})||f_D(\mathbf{x}|\mathbf{e}))$ ya que $\int p(\mathbf{x}|\mathbf{e})d\mathbf{x} = 1$.

donde $D(p(\mathbf{x}|\mathbf{e})||f_D(\mathbf{x}|\mathbf{e}))$ es la *entropía relativa condicional* o *distancia condicional de Kullback-Leibler*, definida como:

$$D(p(\mathbf{x}|\mathbf{e})||f_D(\mathbf{x}|\mathbf{e})) = \int p(\mathbf{e}) \int p(\mathbf{x}|\mathbf{e}) \log \frac{p(\mathbf{x}|\mathbf{e})}{f_D(\mathbf{x}|\mathbf{e})} d\mathbf{x} d\mathbf{e} \ .$$

Como vemos, la descomposición de esa distancia viene afectada claramente por la probabilidad de la evidencia, el primer sumando es la distancia entre la probabilidad de la evidencia y la probabilidad de la evidencia en la red discretizada. El segundo sumando es la distancia entre la respuesta "exacta", $p(\mathbf{x}|\mathbf{e})$ y la obtenida, $f_D(\mathbf{x}|\mathbf{e})$. Dado que este sumando viene multiplicado por la probabilidad de la evidencia, aunque se acote el error de la red entera por ϵ, la cota del error de la respuesta sería $\frac{\epsilon}{p(\mathbf{e})}$.

En caso de orientar la propagación a una variable objetivo, en vez de minimizar la distancia de toda la probabilidad conjunta interesa minimizar la distancia del nodo objetivo X_q condicionado a la evidencia:

$$\int p(x_q|\mathbf{e}) \log \frac{p(x_q|\mathbf{e})}{f_D(x_q|\mathbf{e})} dx_q$$

Por tanto puede que convenga cambiar la discretización y rediscretizar más finamente aquellas regiones que hayan cambiado a la luz de la evidencia. Para ello Kozlov y Koller (1997) definen otra métrica más general que permite discretizar unas regiones más finamente que otras:

Definición 2.5 *Se define la* ***distancia ponderada de entropía relativa*** *(WKL) $W(f(x)||g(x); w(x))$ entre las funciones de densidad $f(x)$ y $g(x)$ con un peso estrictamente positivo $w(x)$ como*

$$W(f(x)||g(x); w(x)) = \int w(x) f(x) \log \frac{f(x)}{g(x)} dx$$

suponiendo que dicha integral exista.

Si el peso es $w(x) = 1$, entonces la distancia WKL coincide con la distancia KL. Además, si el peso $w(x)$ y la función $f(x)$ siguen la misma discretización D, entonces se cumple que :

$$D(f(x)|f_D(x))min_x w_D(x) \leq W(f(x)||f_D(x); w_D(x)) \leq D(f(x)|f_D(x))max_x w_D(x) \ ,$$

dado que, suponiendo que la discretización D consta de m regiones $(S_1, \ldots, S_m)$, entonces ocurre que

$$W(f(x)||f_D(x); w_D(x)) = \sum_{i=1}^{m} \int_{S_i} w_i(x) f_i(x) \log \frac{f(x)}{f_i(x)} \leq$$

$$\leq \sum_{i=1}^{m} maxw_i(x) \int_{S_i} f_i(x) \log \frac{f(x)}{f_i(x)} \leq max_x w_D(x) \sum_{i=1}^{m} \int_{S_i} f_i(x) \log \frac{f(x)}{f_i(x)} =$$

$$= D(f(x)|f_D(x))max_x w_D(x) \ ,$$

y lo mismo pasa con el mínimo.

Veamos ahora como podemos cambiar la discretización de los cliques según sea la evidencia.

En la versión de HUGIN descrita en la sección anterior, en cada clique C_i se discretiza una función resultado de la combinación de los mensajes que le llegan al clique y las propias funciones asociadas al clique. La cuestión es que al discretizar esta función sólo se tiene en cuenta la evidencia que se encuentra por debajo del clique y que ha llegado a éste a través de los mensajes, pero puede que la evidencia que se encuentre por encima de este clique aconseje cambiar la discretización.

Este problema no lo tiene el clique raíz, ya que su discretización se hace teniendo en cuenta toda la evidencia de la red. Por tanto, un peso constantemente uno en el clique raíz es el adecuado. En un clique intermedio sí hay 'pérdida de información', ya que recibe mensajes de todos los cliques excepto del clique padre, que es el que lo conecta con la raíz. Si este mensaje le llegara, el producto de todos los mensajes y las funciones asociadas al clique serían la mejor aproximación de la distribución a posteriori, y no haría falta ningún peso. Pero como no tenemos este mensaje del padre, se usa para la discretización una estimación incompleta de la distribución a posteriori.

Para intentar aproximar esta función lo más posible a la original, se determina un peso, w, que aproxime lo mejor posible este mensaje faltante del clique padre.

La asignación de pesos a los distintos cliques del árbol, se hace de forma que se minimice la distancia WKL entre la función original y la discretizada dentro de cada clique y así minimizar el error de la probabilidad del nodo objetivo q dada la evidencia $\mathbf{e}$. Los pesos en sentido inverso al de la propagación, de arriba a abajo. Para cada clique se supone conocido el peso del clique padre y a partir de él se escoge el peso del clique en cuestión, de forma que minimice la distancia WKL resultante con el padre.

Como hemos dicho, se asignan los pesos desde la raíz a las hojas; por tanto, el primer peso a asignar es el del clique raíz, pero ya sabemos que éste es constantemente 1. Veamos ahora cómo se obtiene el peso para un clique a partir del peso del clique padre.

Sean dos cliques $C_1 = \{X, Y\}$ y $C_2 = \{Y, Z\}$, C_1 padre de C_2 y conocemos $w(x, y)$, peso de C_1. Las funciones $r(x, y)$ y $r(y, z)$ representan el producto de todas las funciones asociadas a los cliques C_1 y C_2 respectivamente y $s(y)$ representa el mensaje que envia C_2 a C_1. Se pretende encontrar el mejor peso $w(y, z)$ sobre C_2 de forma que el mensaje $s(y)$ minimice la entropía relativa entre el verdadero potencial $f(x, y) = r(x, y)s(y)$ y su discretización $f_D(x, y) = r_D(x, y)s_D(y)$. Veamos cómo hacerlo:

$$W(f(x,y)||f_D(x,y); w(x,y)) =$$

$$= \int w(x,y) f(x,y) \log \frac{f(x,y)}{f_D(x,y)} dxdy =$$

$$= \int w(x,y) r(x,y) s(y) \log \frac{r(x,y)s(y)}{r_D(x,y)s_D(y)} dxdy =$$

$$= \int w(x,y) r(x,y) s(y) \log \frac{r(x,y)}{r_D(x,y)} dxdy +$$

$$+ \int w(x,y) r(x,y) s(y) \log \frac{s(y)}{s_D(y)} dxdy =$$

$$= W(r(x,y)||r_D(x,y); w(x,y)s(y)) + W(s(y)||s_D(y); \int w(x,y) r(x,y) dx)$$

Nótese como se integra el peso de la última distancia, ya que al depender $s(y)$ y $s_D(y)$ sólo de la variable y, el peso ha de depender a su vez sólo de y.

De esta forma se descompone, pues, la distancia entre f y f_D en la suma de otras dos distancias. Ya que $r_D(x,y) = \frac{f_D(x,y)}{s_D(x,y)}$, la discretización del potencial del clique C_1 es la responsable de la minimización del primer término de la suma de distancias. El segundo término se puede escribir también como :

$$W\left(s(y)||s_D(y); \int w(x,y)r(x,y)dx\right) = W\left(s(y)||s_D(y); \frac{\int w(x,y)f(x,y)dx}{s(y)}\right) \quad (2.2)$$

y su minimización se hace implícitamente al discretizar C_2, ya que entonces discretizamos $f(y,z)$ y a partir de ahí obtenemos sumando $s_D(y)$. Por tanto, si minimizamos esta última distancia, minimizaremos la distancia del potencial de C_1 y su discretización, que es la que usaremos en la propagación.

Pero el objetivo real es buscar un peso $w(y,z)$:

$$W(f(y,z)||f_D(y,z); w(y,z)) = \int w(y,z)f(y,z)\log\frac{f(y,z)}{f_D(y,z)}dydz$$

$$= \int w(y,z)f(y,z)\log\frac{s(y)}{s_D(y)}dydz + \int w(y,z)f(y,z)\log\frac{\frac{f(y,z)}{s(y)}}{\frac{f_D(y,z)}{s_D(y)}}dydz$$

$$= \int \frac{s(y)w(y,z)f(y,z)}{s(y)}\log\frac{s(y)}{s_D(y)}dydz + \int w(y,z)f(y,z)\log\frac{\frac{f(y,z)}{s(y)}}{\frac{f_D(y,z)}{s_D(y)}}dydz \ .$$

Por tanto, si se reduce la distancia $W(f(y,z)||f_D(y,z); w(y,z))$ se reduce también la distancia

$$W\left(s(y)||s_D(y); \frac{\int w(y,z)f(y,z)dz}{s(y)}\right) \quad (2.3)$$

Fijándonos en las ecuaciones (2.2) y (2.3), se ve que los pesos de los cliques vecinos han de cumplir la siguiente condición :

$$\frac{\int w(x,y)f(x,y)dx}{s(y)} = \frac{\int w(y,z)f(y,z)dz}{s(y)} \quad (2.4)$$

Como el peso del clique C_1, $w(x, y)$, es conocido, con esta ecuacón se puede conocer el peso del clique C_2, $w(y, z)$, que es precisamente el objetivo buscado.

La discretización dinámica se integra en el proceso de propagación como sigue. Inicialmente, para poder discretizar cada clique según esta discretización dinámica es necesario conocer los pesos de cada clique, que no conocemos, además, para poder propagar necesitamos tener una discretización, que no tenemos ya que no tenemos los cliques. Para solucionar esto, lo que se hace es asignar inicialmente a todos los cliques pesos uniformes 1, y luego propagar. Más concretamente el algoritmo es como sigue:

1. Asignar pesos $w = 1$ a todos los cliques.
2. Hasta que la probabilidad a posteriori converja repetir:
3. *a*) Realizar el algoritmo de propagación.
   b) Propagar pesos hacia abajo en el árbol.

En primer lugar se lleva a cabo una propagación de probabilidades, con lo que cambian los potenciales de los cliques, con lo cual, aplicando la fórmula (2.4) y teniendo en cuenta que el peso del clique raíz es siempre 1, se obtienen actualizaciones de los pesos. Al cambiar los pesos, cambian las discretizaciones, y por tanto la propagación también, y así sucesivamente hasta que la probabilidad a posteriori converja.

Para asegurar que lo pesos tengan la misma discretización que sus correspondientes potenciales se guardan en el mismo árbol BSP. Ahora bien, siguiendo este proceso, en realidad los cambios que se hacen en las discretizaciones están restringidos, es decir, sólo se pueden discretizar más finamente algunas regiones, pero no hacer cambios más drásticos, que puede que sean los que sugieren las evidencias. Para solucionarlo se poda el árbol en cada iteración, eliminando aquellas hojas que aporten al error total menos que una hoja 'media'. El error en las hojas se estima por métodos de integración de Monte Carlo durante el proceso de discretización. El potencial del clique será discretizado de nuevo en la siguiente iteración, y en ella aquellas regiones con los pesos mayores serán seguramente rediscretizadas más finamente.

2.4. Conclusiones

En este capítulo hemos revisado un método para el tratamiento de variables continuas que se basa en la discretización. El enfoque descrito tiene unas importantes características algunas de las cuales tendrán una influencia notable en nuestro trabajo posterior. Entre ellas, podemos destacar:

- No se considera cada variable de forma aislada, sino que una variable se discretiza en función de sus relaciones con otras variables.
- Una misma variable puede tener distintas discretizaciones en distintos potenciales.
- La discretización depende de las observaciones que tengamos en un momento dado.
- La discretación se obtiene mediante un proceso de mejora iterativo que conlleva varias etapas de propagación en lugar de las dos (recolección y distribución de la evidencia) de los algoritmos con variables discretas.

Entre los posibles inconvenientes de este método, podríamos citar el enorme tamaño que pueden llegar a tener los árboles BSP, aun para representar distribuciones de probabilidad simples. Estos árboles van aumentando de complejidad a medida que se van realizando operaciones de combinación. Esto también hace que la aproximación de estos árboles sea un problema difícil y consuma un tiempo considerable.

Capítulo 3

Distribuciones Condicionales Gaussianas

3.1. Introducción

En el primer capítulo revisamos algunos procedimientos de inferencia para redes bayesianas con variables aleatorias discretas. Pero, frecuentemente, en problemas reales aparecen simultáneamente variables discretas y continuas. Para abordar esta situación, en el segundo capítulo se estudió el uso de la discretización, con el fin de convertir las variables continuas en variables discretas. Pero el tránsito de variable continua a discreta es una aproximación, cuya precisión depende de la discretización empleada, y aún con métodos sofisticados como la *discretización dinámica* puede que no se obtengan buenos resultados, o que para obtenerlos el precio que se pague sea demasiado elevado, en términos de memoria y tiempo de cálculo.

Ahora bien, se discretiza ante la imposibilidad de realizar los cálculos de forma exacta para las variables continuas, pero, como ya se dijo en la introducción del capítulo anterior, hay ciertas clases de variables continuas en la que los cálculos sí se pueden realizar de forma exacta, y una de ellas la estudiaremos aquí, la clase de *distribuciones Gaussianas multivariantes* (Castillo, Gutiérrez, y Hadi 1996).

En realidad, lo que vamos a estudiar es una generalización de ese modelo, la *distribución condicional Gaussiana* (Cowell, Dawid, Lauritzen, y Spiegelhalter 1999; Lauritzen 1992; Olesen 1993), en la que conviven variables discretas y continuas en una red híbrida. Este modelo mixto verifica que la distribución de condicional de las variables continuas dadas las discretas sea una distribución Gaussiana multivariante, de ahí que sea una extensión del anterior (Lauritzen y Wermuth 1989). Con el modelo condicional Gaussiano, completamos el estudio de los métodos existentes en la literatura para el tratamiento de redes bayesianas con variables discretas y continuas.

3.2. Definiciones

En este apartado daremos una breves nociones de notación y algunas definiciones y propiedades básicas. Nos centraremos en los resultados fundamentales, no en las demostraciones ni en los resultados previos, que pueden encontrarse en el trabajo de Lauritzen y Wermuth (1989).

Al tratarse de un modelo mixto tendremos variables discretas y continuas. Por tanto, el conjunto de variables $\mathbf{X}$ estará dividido en dos, $\mathbf{X} = \mathbf{Y} \cup \mathbf{Z}$, donde $\mathbf{Y}$ serán las variables discretas y $\mathbf{Z}$ las continuas, siendo $|\mathbf{Y}| = d$ y $|\mathbf{Z}| = c$ el número de variables discretas y continuas respectivamente que hay en el modelo. Por tanto, un elemento del espacio de estados conjunto será de la forma :

$$\mathbf{x} = (\mathbf{y}, \mathbf{z}) = (y_1, \ldots, y_d, z_1, \ldots, z_c)$$

donde y_i para $i = 1, \ldots, d$ son valores cualitativos y z_j para $j = 1, \ldots, c$ son números reales.

Definición 3.1 *Una variable aleatoria mixta* $\mathbf{X} = (\mathbf{Y}, \mathbf{Z})$ *se dice que sigue una distribución* ***condicional Gaussiana*** *(distribución CG) si la distribución conjunta de las variables del modelo (de todas las variables, discretas y continuas) tiene una densidad*

$$f(\mathbf{x}) = f(\mathbf{y}, \mathbf{z}) = \chi(\mathbf{y}) \exp\{g(\mathbf{y}) + h(\mathbf{y})^T \mathbf{z} - \mathbf{z}^T K(\mathbf{y}) \mathbf{z}/2\} \tag{3.1}$$

donde $\chi(\mathbf{y}) \in \{0, 1\}$ *indica si* f *es positiva en* $\mathbf{y}$ *o no,* g *es una función que devuelve un número real,* h *una función que devuelve un vector de tamaño* c*,* K *una función que devuelve una matrix* $c \times c$*, y* $\mathbf{z}^T$ *es la traspuesta de* $\mathbf{z}$*.*

Como comentamos en la introducción, al decir que $\mathbf{X}$ sigue una distribución CG, queremos decir que las variables continuas siguen una distribución Gaussiana multivariante dadas las discretas, esto es :

$$\mathbf{Z}|\mathbf{Y} = \mathbf{y} \rightarrow \mathcal{N}_c(\xi(\mathbf{y}), \Sigma(\mathbf{y})) \quad \text{siempre que } p(\mathbf{y}) = p(\mathbf{Y} = \mathbf{y}) > 0$$

siendo

$$\xi(\mathbf{y}) = K(\mathbf{y})^{-1}h(\mathbf{y}) \quad , \quad \Sigma(\mathbf{y}) = K(\mathbf{y})^{-1} \tag{3.2}$$

con $\Sigma(\mathbf{y})$ definida positiva, al igual que ocurre en la distribución normal multivariante.

Si conocemos las funciones g, h, K, tenemos la distribución de $\mathbf{X}$, de ahí que a la tripleta (g, h, K) se la denomine *caracteristicas canónicas de la distribución.* Obviamente, estas características canónicas de la distribución están bien definidas sólo si $\chi(\mathbf{y}) > 0$. Pero la distribución la podemos poner también en términos de p, ξ, y Σ, luego la tripleta (p, ξ, Σ) también vale para definir la distribución y se conoce como *momentos característicos de la distribución.* Si sólo tenemos variables discretas, las características canónicas de la distribución serán $(g, 0, 0)$, y si sólo hay variables continuas, las características serán $(0, h, K)$, indicando los ceros componentes no definidos.

Tal y como vimos en el primer capítulo, las distribuciones de probabilidad, así como los resultados de las operaciones entre ellas se almacenan como potenciales. Veamos pues en qué consiste un potencial en este caso:

Definición 3.2 *Una función* ϕ *se dice que es un* ***potencial CG*** *si es de la forma*

$$\phi(\mathbf{x}) = \phi(\mathbf{y}, \mathbf{z}) = \chi(\mathbf{y}) \exp\left\{g(\mathbf{y}) + h(\mathbf{y})^T\mathbf{z} - \mathbf{z}^T K(\mathbf{y})\mathbf{z}/2\right\} \ ,$$

donde K *sólo ha de verificar ser una matriz simétrica. Por tanto,* ϕ *no ha de ser una densidad necesariamente. Se llama características canónicas del potencial a la tripleta* (g, h, K)*.*

Una diferencia básica es que los momentos carcterísticos de un potencial CG sólo están bien definidos cuando K es definida positiva para todos aquellos $\mathbf{y}$ tales que $\chi(\mathbf{y}) > 0$. Entonces, para pasar de las características canónicas a los momentos canónicos sólo falta obtener p, ya que Σ y ξ los conocemos de (3.2). Para obtener p sólo hay que integrar $\mathbf{z}$ de la distribución conjunta (3.1), de lo que obtenemos[1]:

$$p(\mathbf{y}) \propto \{det\Sigma(\mathbf{y})\}^{\frac{1}{2}} \exp\{g(\mathbf{y}) + h(\mathbf{y})^T \Sigma(\mathbf{y}) h(\mathbf{y})/2\} \tag{3.3}$$

También podemos calcular las características canónicas a partir de los momentos:

$$\begin{aligned} K(\mathbf{y}) &= \Sigma(\mathbf{y})^{-1} \\ h(\mathbf{y}) &= K(\mathbf{y})\xi(\mathbf{y}) \\ g(\mathbf{y}) &= \log p(\mathbf{y}) + \{\log detK(\mathbf{y}) - c\log 2\pi - \xi(\mathbf{y})^T K(\mathbf{y})\xi(\mathbf{y})\}/2 \end{aligned}$$

Ya que tenemos los potenciales, el siguiente paso natural es definir las operaciones a realizar sobre ellos y que se usarán en la propagación.

Definición 3.3 *Sea ϕ un potencial definido sobre $U = (Y_1 \times Z_1)$. La* ***extensión*** *de dicho potencial a $\bar{\phi}$ definido sobre $W = (Y_1 \times Y_2) \times (Z_1 \times Z_2)$ se define como $\bar{\phi}(y_1, y_2, z_1, z_2) = \phi(y_1, z_1)$.*

Si (g, h, K) son las características canónicas del potencial, la actualización lo único que hace es insertar ceros a las características hasta obtener la dimensión deseada, esto es, las nuevas características canónicas serían :

$$\bar{g}(y_1, y_2) = g(y) \;\; ; \;\; \bar{h}(y_1, y_2) = \begin{pmatrix} h(y_1) \\ 0 \end{pmatrix} \;\; ; \;\; \bar{K}(y_1, y_2) = \begin{pmatrix} K(y_1) & 0 \\ 0 & 0 \end{pmatrix}$$

Siempre que quede claro no haremos distinciones entre un potencial y su extensión.

[1] Si $\Sigma(\mathbf{y})$ no es definida positiva, como dijimos anteriormente que tenía que ser, entonces (3.3) no es una probabilidad.

Definición 3.4 *Sea ϕ un potencial CG definido sobre $\mathbf{X} = (\mathbf{Y}_1 \times \mathbf{Z}_1) \times (\mathbf{Y}_2 \times \mathbf{Z}_2)$, y sea $(\mathbf{z}_1 \times \mathbf{z}_2) \in (\mathbf{Z}_1 \times \mathbf{Z}_2)$ fijos. La **restricción** $\phi^{R(\mathbf{Z}_1=\mathbf{z}_1,\mathbf{Z}_2=\mathbf{z}_2)}$ de ϕ a $\mathbf{Y}_1 \times \mathbf{Y}_2$ se define como*

$$\phi^{R(\mathbf{Z}_1=\mathbf{z}_1,\mathbf{Z}_2=\mathbf{z}_2)}(\mathbf{y}_1, \mathbf{y}_2) = \phi(\mathbf{y}_1, \mathbf{z}_1, \mathbf{y}_2, \mathbf{z}_2)$$

En términos de características canónicas, esta operación consiste simplemente en quedarse con las partes relevantes.

La combinación se define de la forma usual :

Definición 3.5 *Dados dos potenciales CG, ϕ y ψ definidos sobre $\mathbf{X}$ y $\mathbf{X}'$ respectivamente, se define su **combinación,** $\phi\psi$, sobre $\mathbf{X} \cup \mathbf{X}'$ como*

$$(\phi\psi)(\mathbf{x}) = \phi(\mathbf{x})\psi(\mathbf{x})$$

Si lo expresamos en términos de las características canónicas, la multiplicación se convierte en simple adición de las componentes:

$$(g_1, h_1, K_1) \times (g_2, h_2, K_2) = (g_1 + g_2, h_1 + h_2, K_1 + K_2)$$

La división se define de igual forma que la multiplicación, pero observando el caso particular de dividir por cero.

Definición 3.6 *Dados dos potenciales CG, ϕ y ψ definidos sobre $\mathbf{X}$ y $\mathbf{X}'$ respectivamente, se define su **división**, ϕ/ψ, para un $\mathbf{x} \in \mathbf{X} \cup \mathbf{X}'$ como*

$$(\phi/\psi)(\mathbf{x}) = \begin{cases} \frac{\phi(\mathbf{x})}{\psi(\mathbf{x})} & \text{si} \quad \psi(\mathbf{x}) \neq 0 \\ 0 & \text{en otro caso} \end{cases}$$

Obviamente, si lo ponemos en términos de las características canónicas esta división se convierte en sustracción de las componentes

$$(g_1, h_1, H_1)/(g_2, h_2, K_2) = (g_1 - g_2, h_1 - h_2, K_1 - K_2)$$

El problema surge ahora con la marginalización. Esto es porque cuando sumamos potenciales CG no obtenemos otro potencial CG, sino una mixtura de potenciales CG. Por tanto, definir una marginalización sobre los potenciales CG requiere algo más de atención que la otras operaciones.

Primero veamos cómo se eliminan mediante marginalización las variables continuas. Supongamos que tenemos un potencial ϕ con

$$\mathbf{z} = \begin{pmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \end{pmatrix} \quad h = \begin{pmatrix} h_1 \\ h_2 \end{pmatrix} \quad K = \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{pmatrix}$$

siendo la dimensión de $\mathbf{y}_1$ igual a p. Entonces la siguiente proposición dice cuál sería la marginal de ϕ. La demostración de dicha proposición se puede encontrar en (Cowell, Dawid, Lauritzen, y Spiegelhalter 1999).

Proposición 3.1 *La integral* $\int \phi(\mathbf{y}, \mathbf{z}_1, \mathbf{z}_2) d\mathbf{z}_1$ *es finita si y sólo si* K_{11} *es definida positiva. Entonces dicha integral, esto es, la marginal de* ϕ *sobre* $\mathbf{z}_1$ *es un potencial CG* $\widetilde{\phi}$ *con características canónicas:*

$$\begin{aligned} \widetilde{g}(\mathbf{y}) &= g(\mathbf{y}) + \{p \log(2\pi) - \log det K_{11}(\mathbf{y}) + h_1(\mathbf{y})^T K_{11}(\mathbf{y})^{-1} h_1(\mathbf{y})\}/2 \\ \widetilde{h}(\mathbf{y}) &= h_2(\mathbf{y}) - K_{21}(\mathbf{y}) K_{11}(\mathbf{y})^{-1} h_1(\mathbf{y}) \\ \widetilde{K}(\mathbf{y}) &= K_{22}(\mathbf{y}) - K_{21}(\mathbf{y}) K_{11}(\mathbf{y})^{-1} K_{12}(\mathbf{y}) \end{aligned}$$

Para la marginalización con variables discretas, se distinguen dos casos :

Caso 1: **Si** h **y** K **no dependen de** $\mathbf{Y}'$. Esto es, $h(\mathbf{y}, \mathbf{y}') \equiv h(\mathbf{y})$ y $K(\mathbf{y}, \mathbf{y}') \equiv K(\mathbf{y})$. Bajo estas condiciones se define la marginal $\widetilde{\phi}$ de ϕ sobre $\mathbf{y}'$ de la forma usual:

$$\widetilde{\phi}(\mathbf{y}, \mathbf{z}) = \sum_{\mathbf{y}'} \phi(\mathbf{y}, \mathbf{y}', \mathbf{z}) =$$

$$= \sum_{\mathbf{y}'} \chi(\mathbf{y},\mathbf{y}') \exp\{g(\mathbf{y},\mathbf{y}') + h(\mathbf{y})^T\mathbf{z} - \mathbf{z}^T K(\mathbf{y})\mathbf{z}/2\} =$$

$$= \exp\{h(\mathbf{y})^T\mathbf{z} - \mathbf{z}^T K(\mathbf{y})\mathbf{z}/2\} \sum_{\mathbf{y}'} \chi(\mathbf{y},\mathbf{y}') \exp g(\mathbf{y},\mathbf{y}')$$

Esto equivale a las siguientes características canónicas:

$$\begin{aligned} \widetilde{g}(\mathbf{y}) &= \log \sum_{\mathbf{y}':\chi(\mathbf{y},\mathbf{y}')=1} \exp g(\mathbf{y},\mathbf{y}') \\ \widetilde{h}(\mathbf{y}) &= h(\mathbf{y}) \\ \widetilde{K}(\mathbf{y}) &= K(\mathbf{y}) \end{aligned}$$

Caso 2: h **o** K **dependen de** $\mathbf{Y}'$. En este caso, la simple adición de potenciales CG no es un potencial CG. El método que sigue sólo está bien definido si $K(\mathbf{y},\mathbf{y}')$ es definida positiva. Lo describiremos en términos de los momentos característicos $\{p,\xi,\Sigma\}$. La marginal $\widetilde{\phi}$ de ϕ es el potencial CG cuyos momentos característicos $\{\widetilde{p},\widetilde{\xi},\widetilde{\Sigma}\}$ son :

$$\widetilde{p}(\mathbf{y}) = \sum_{\mathbf{y}'} p(\mathbf{y},\mathbf{y}'), \qquad \widetilde{\xi}(\mathbf{y}) = \sum_{\mathbf{y}'} \xi(\mathbf{y},\mathbf{y}')p(\mathbf{y},\mathbf{y}')/\widetilde{p}(\mathbf{y})$$

$$\widetilde{\Sigma}(\mathbf{y}) = \sum_{\mathbf{y}'} \Sigma(\mathbf{y},\mathbf{y}')p(\mathbf{y},\mathbf{y}')/\widetilde{p}(\mathbf{y}) + \sum_{\mathbf{y}'} (\xi(\mathbf{y},\mathbf{y}') - \widetilde{\xi}(\mathbf{y}))^T (\xi(\mathbf{y},\mathbf{y}') - \widetilde{\xi}(\mathbf{y}))p(\mathbf{y},\mathbf{y}')/\widetilde{p}(\mathbf{y})$$

Teniendo en cuenta lo anterior se puede definir la operación de *marginalización* como sigue.

Definición 3.7 *La* ***marginalización*** *sobre variables continuas y discretas se realiza marginalizando primero sobre las variables continuas y luego sobre las discretas. Si en esta segunda marginalización se da el caso 1, esto es,* (h,K) *independientes de* $\mathbf{y}'$*, entonces se dice que la marginalización es* ***fuerte****. Si por el contrario se da el caso 2, se dice que es* ***débil*** *y se usa el proceso de marginalización visto arriba.*

La marginalización (tanto débil como fuerte) se denota por $\sum_{W\setminus U}\phi_W$, esto es, $W\setminus U$ es el conjunto de variables sobre el que marginalizamos, las que se eliminan, y U el conjunto de variables al que marginalizamos, es decir, que se "mantienen".

Es fácil demostrar que la marginalización débil verifica la regla de composición: Si tenemos $U \subset V \subset W$, entonces

$$\sum_{V\setminus U}\Big(\sum_{W\setminus V}\phi_W\Big) = \sum_{W\setminus U}\phi_W.$$

Sin embargo, sólo las marginalizaciones fuertes se comportan bien si hay combinaciones de potenciales involucrados, esto es, en general tenemos que

$$\sum_{W\setminus V}(\phi_W\phi_V) \neq \phi_V\Big(\sum_{W\setminus V}\phi_W\Big). \tag{3.4}$$

Debido a este problema, los algoritmos de propagación, en concreto el HUGIN, ha de ser ampliado para poder aplicarlo a este caso adecuadamente.

Si la marginalización es fuerte, entonces la igualdad en (3.4) sí se alcanza y entonces se puede aplicar sin más problemas la propagación HUGIN.

3.3. Grafos marcados y su árbol de cliques

Dado que las distribuciones CG no son cerradas para la marginalización, a la hora de aplicarlas a las redes bayesianas han de imponerse ciertas restricciones en el tipo de modelos a considerar. Como en los capítulos anteriores, trabajamos con un grafo, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ donde V es un conjunto finito de vértices, y E el conjunto de aristas. Ahora bien, necesitamos trabajar con grafos donde los vértices estén *marcados*, en el sentido de que están divididos en dos grupos, Δ y Γ. Un grafo de este tipo se denomina *grafo marcado*. Los vértices en el conjunto Δ serán variables discretas y los del conjunto Γ serán variables continuas.

Como vimos en el capítulo 1, los métodos propagación locales es dividen el grafo en

distintas componentes independientes, que son los cliques. Sin embargo, dada la clara asimetría que hay entre las variables discretas y continuas, también hay que tener en cuenta las *marcas* del grafo. Para un estudio más detallado de estas cuestiones teóricas de los grafos, ver (Leimer 1989).

Definición 3.8 *Una tripleta (A, B, C) de conjuntos disjuntos del conjunto V de vértices de un grafo marcado y no dirigido $\mathcal{G}$ se dice que forma una* ***descomposición fuerte*** *de $\mathcal{G}$ si $V = A \cup B \cup C$ y se verifican además las tres siguientes condiciones:*

1. *C separa a A de B.*
2. *C es un subconjunto completo de V.*
3. *$C \subseteq \Delta$ o $B \subseteq \Gamma$.*

Cuando se verifican las tres condiciones se dice que (A, B, C) *descomponen* a $\mathcal{G}$ en las *componentes* $\mathcal{G}_{A\cup C}$ y $\mathcal{G}_{B\cup C}$.

Si sólo se verifican las dos primeras condiciones de la definición, entonces se dice que (A, B, C) forma una **descomposición débil** de $\mathcal{G}$. Por tanto, la diferencia entre una descomposición débil y una fuerte es que la débil no tiene en consideración las marcas del grafo, que es justo lo que andábamos buscando.

Un grafo descomponible es aquel que se puede llegar a descomponer en sus cliques. Veamos, mediante una definición recursiva, cómo hacerlo:

Definición 3.9 *Un grafo marcado no dirigido $\mathcal{G}$ se dice que es* ***descomponible*** *si es completo, o si existe una descomposición (A, B, C) de $\mathcal{G}$ con A y B no vacíos en componentes $\mathcal{G}_{A\cup C}$ y $\mathcal{G}_{B\cup C}$ que sean a su vez subgrafos descomponibles.*

En grafos sin marcas, sólo se necesita que el grafo sea triangulado para conseguir que sea descomponible, esto es, obtener su árbol de cliques. En grafos marcados se necesita que éste carezca de cierto tipo de caminos (Leimer 1989):

Proposición 3.2 *Un grafo marcado no dirigido $\mathcal{G}$ es descomponible si y sólo si es triangulado y no contiene ningún camino entre dos vértices discretos no vecinos que esté únicamente compuesto de vértices continuos.*

En el capítulo primero vimos que la estructura sobre la que se realiza la propagación es el árbol de cliques, y cómo se construye a partir de la red original. El proceso ahora es similar, pero teniendo en cuenta las siguientes peculiaridades.

La primera parte del proceso permanece intacta, esto es, a partir del grafo no dirigido acíclico se obtiene el grafo moral simplemente quitando las direcciones de las aristas, y añadiendo nuevas aristas entre padres que tengan hijos en común y no estén ya conectados. Ahora lo que se busca es un grafo descomponible. Esto se hace añadiendo aristas de forma que se eviten los caminos prohibidos que se señalan en la proposición 3.2. La forma de hacerlo es, al igual que con redes discretas, mediante un proceso de eliminación de variables, pero en este caso se han de eliminar siempre primero las variables continuas (Cowell, Dawid, Lauritzen, y Spiegelhalter 1999).

Ya que tenemos el grafo descomponible, el paso siguiente es construir el árbol de cliques (recuérdese que ha de verificarse la condición expresada en la definición 1.15).

Diversos algoritmos detallados y explicados para obtener de forma sistemática tanto el grafo descomponible como el árbol de cliques a partir de éste se pueden encontrar en (Cowell, Dawid, Lauritzen, y Spiegelhalter 1999; Leimer 1989).

Ahora bien, llegados a este punto, entra en juego de nuevo la asimetría entre variables continuas y discretas, y es por ello que se requiere una condición más para hacer que el esquema de propagación funcione correctamente en este tipo de estructura.

Definición 3.10 *Un nodo R de un árbol de cliques es una* ***raíz fuerte*** *si cualquier par de nodos vecinos del árbol A, B con A más cercano a R que B se tiene que*

$$(B \setminus A) \subseteq \Gamma \quad o \quad (B \cap A) \subseteq \Delta. \tag{3.5}$$

Para un árbol de cliques, la condición (3.5) equivale a que $(A \setminus B, B \setminus A, A \cap B)$ forme una descomposicón fuerte de $\mathcal{G}_{A \cup B}$. Esto quiere decir que si un separador entre dos

cliques vecinos no es totalmente discreto, es decir, no todas sus variables son discretas, entonces el clique más lejano a la raíz sólo tiene vértices continuos fuera del separador.

Leimer (1989) prueba que los cliques de un grafo descomponible siempre se pueden agrupar en un árbol de cliques de forma que al menos uno de ellos sea una raíz fuerte. Por tanto, a partir de ahora suponemos que el árbol de cliques construido tiene, al menos, una raíz fuerte.

3.4. Propagación de probabilidades

El grafo original especifica claramente las relaciones de dependencia/independencia que hay entre las variables, y verifica, la condición (1.1) expresada en la definición 1.8 del primer capítulo, es decir, la densidad conjunta de todas las variables de la red se puede expresar como el producto de las densidades condicionadas de las variables dados sus padres en el grafo. Pero en el modelo CG surge la restricción añadida de que **ningún nodo continuo puede tener hijos discretos**. Si no se cumple esta condición, se pueden usar varias alternativas, como las expuestas en (Koller, Lerner, y Anguelov 1999; Lauritzen 1992). El motivo de imponer esta restricción es que, si no es así, la distribución condicionada de una variable continua a una discreta descendiente sería un cociente de mixturas de Gaussianas y, por tanto, no perteneciente a la familia CG.

Una vez comprobado que se verifica esta condición, lo primero que se hace es fijar estas densidades condicionadas. Para cada variable discreta Y_i, hay que especificar la distribución dados los estados de los padre, que siempre serán discretos, por lo visto en el párrafo superior. Será por tanto una tabla de valores reales como las del capítulo 1. Para cada variable continua Z_i, la distribución condicionada dados sus padres (continuos o discretos) es de la forma

$$Z_i|\pi_i \rightarrow \mathcal{N}(\alpha(\mathbf{y}) + \beta(\mathbf{y})^T\mathbf{z}, \gamma(\mathbf{y})) \ , \tag{3.6}$$

donde $(\mathbf{y}, \mathbf{z})$ son los estados de los padres de Z_i, $\mathbf{y}$ los estados de los padres discretos y

z los estados de los padres continuos, $\gamma(\mathbf{y}) > 0$, $\alpha(\mathbf{y})$ un número real, y $\beta(\mathbf{y})$ un vector de la misma dimensión que la parte continua de los padres.

Por tanto, esta densidad condicional se corresponde con un potencial CG ϕ_{Z_i} definido sobre la combinación $(\mathbf{y}, \mathbf{z}, z_i)$, donde $(\mathbf{y}, \mathbf{z})$ son los estados de los padres y z_i el valor de la variable Z_i en cuestión. Las características canónicas $(g_{Z_i}, h_{Z_i}, K_{Z_i})$ son

$$\begin{aligned} g_{Z_i}(\mathbf{y}) &= -\frac{\alpha(\mathbf{y})^2}{2\gamma(\mathbf{y})} - [\log\{2\pi\gamma(\mathbf{y})\}]/2 \\ h_{Z_i}(\mathbf{y}) &= \frac{\alpha(\mathbf{y})}{\gamma(\mathbf{y})} \begin{pmatrix} 1 \\ -\beta(\mathbf{y}) \end{pmatrix} \\ K_{Z_i}(\mathbf{y}) &= \frac{1}{\gamma(\mathbf{y})} \begin{pmatrix} 1 & -\beta(\mathbf{y})^T \\ -\beta(\mathbf{y}) & \beta(\mathbf{y})\beta(\mathbf{y})^T \end{pmatrix} \end{aligned} \tag{3.7}$$

El paso siguiente es asociar potenciales a los distintos cliques del árbol. Recordemos que en el árbol de cliques teníamos cliques, $\mathcal{C}$, y separadores, $\mathcal{S}$, que eran las intersecciones de los cliques vecinos, y que teníamos potenciales asociados, tanto a los cliques como a los separadores, de tal forma que se verificaba que el potencial ϕ_U calculado como

$$\phi_U = \frac{\prod_{V \in \mathcal{C}} \phi_V}{\prod_{S \in \mathcal{S}} \phi_S} \tag{3.8}$$

es proporcional a la densidad conjunta de todas las variables. Dado que todos los potenciales involucrados son potenciales CG, la densidad conjunta, y ese potencial ϕ_U también serán potenciales CG.

Se supone además que se cumple que si tenemos un nodo V con separador S, entonces si

$$\phi_S(\mathbf{x}) = 0 \Rightarrow \phi_V(\mathbf{x}) = 0 \tag{3.9}$$

Esto permite manejar adecuadamente aquellas situaciones en las que se modeliza la imposibilidad de que ocurra determinada situación (que se dé cierto estado) asignándole

potencial cero. Ahora el siguiente paso es inicializar el árbol de cliques (que tiene una raíz fuerte) de forma que se verifique (3.9) y que el potencial total (3.8) sea la densidad conjunta especificada por el modelo. Primero se asigna cada vértice (variable) X_i del grafo original a un clique V del árbol. Al igual que con redes discretas, esto se ha de hacer de forma que se cumpla $\{X_i\} \cup \pi_i \subseteq V$. Así, a cada clique V del árbol se le puede asignar un potencial ϕ_V que sea el producto de todos los potenciales ϕ_{X_i} de las variables asignadas al clique. A aquellos cliques a los que no se les haya asignado ninguna variable y a los separadores, se les asigna inicialmente potenciales $\phi \equiv 1$, es decir, potenciales CG con características canónicas $(0, 0, 0)$.

Con esto se completa la estructura sobre la que desarrollar la propagación. Los cliques son objetos que contienen información en forma de potenciales y los separadores los podemos considerar como canales de comunicación a través de los que fluye la información.

Antes de empezar a propagar, se incorpora la evidencia. En este caso, puede que conozcamos los valores que toman variables discretas y continuas, y por tanto las trataremos por separado. Las discretas exactamente igual que en el capítulo 1, mediante las funciones de Dirac. Veamos cómo se tratan las evidencias continuas. Supongamos que sabemos que la variable continua Z_i toma un determinado valor z_i^*. Entonces se introduce esta evidencia en todos aquellos cliques que contengan a la variable Z_i, modificando sus potenciales de forma que la variable Z_i se fije al valor z_i^*. Si el potencial ϕ tiene características canónicas (g, h, K) con

$$h(\mathbf{y}) = \begin{pmatrix} h_1(\mathbf{y}) \\ h_{Z_i}(\mathbf{y}) \end{pmatrix} \qquad K(\mathbf{y}) = \begin{pmatrix} K_{11}(\mathbf{y}) & K_{1Z_i}(\mathbf{y}) \\ K_{Z_i1}(\mathbf{y}) & K_{Z_iZ_i}(\mathbf{y}) \end{pmatrix}$$

entonces el nuevo potencial ϕ^* tendrá sus correspondientes características canónicas (g^*, h^*, K^*) dadas por

$$\begin{aligned} K^*(\mathbf{y}) &= K_{11}(\mathbf{y}) \\ h^*(\mathbf{y}) &= h_1(\mathbf{y}) - z_i^* K_{Z_i1}(\mathbf{y}) \\ g^*(\mathbf{y}) &= g(\mathbf{y}) + h_{Z_i}(\mathbf{y}) z_i^* - K_{Z_iZ_i}(\mathbf{y})(z_i^*)^2/2 \end{aligned} \tag{3.10}$$

Una vez que se ha introducido la evidencia, se verifica que el potencial (3.8) es

proporcional a la densidad condicional dadas las variables observadas.

En estas condiciones, la propagación se puede llevar a cabo usando el algoritmo HUGIN. Por supuesto las definiciones de *absorción, recolección y distribución* son las mismas, pero con algunas diferencias puntuales que comentaremos a continuación.

La operación básica que se realizaba cuando propagabamos era la de *absorción*. La propagación consistía en el paso de mensajes, mediante la absorción, de un nodo a otro, de forma que se repartiese la información que contenían esos nodos. Veamos si eso ocurre ahora en el caso CG:

Proposición 3.3 *Si ϕ_S es la marginal fuerte de ϕ_W, entonces U y W son consistentes tras absorber W de U, esto es*

$$\sum_{U \setminus W} \phi_U^* = \phi_S^* = \sum_{W \setminus U} \phi_W^* \tag{3.11}$$

Demostración: El separador S, esta compuesto de la intersección entre U y W, luego se verifica que

$$\sum_{W \setminus U} \phi_W^* = \sum_{W \setminus U} \phi_W (\frac{\phi_S^*}{\phi_S}) = (\frac{\phi_S^*}{\phi_S}) \sum_{W \setminus U} \phi_W = (\frac{\phi_S^*}{\phi_S}) \phi_S = \phi_S^*$$

La otra igualdad es exactamente igual. Además, este resultado es falso si ϕ_S es la marginal débil, ya que entonces, no tiene por qué ser cierto que $\sum_{W \setminus U} \phi_W (\frac{\phi_S^*}{\phi_S}) = (\frac{\phi_S^*}{\phi_S}) \sum_{W \setminus U} \phi_W$, como se indica en (3.4).

■

En este punto, sólo queda describir los esquemas de *recolección* y de *distribución*. La idea es enviar una petición de recolecćon desde una raíz fuerte del árbol, R. Cada vecino le enviará una petición de recolección a su vecino, y así hasta llegar a las hojas, que empezarán a enviar mensajes hacia arriba, de forma que, al finalizar, la raíz habrá absorbido toda la información disponible en el árbol.

Al enviar la petición de recolección desde una raíz fuerte R, si U y V son vecinos con separador S y U está más cerca de de R que V, entonces el hecho de enviar la petición de distribución desde R hace que U absorba de V. Así, tras la recolección, el potencial de S es la marginal de V, y dado que la raíz es fuerte, la marginal será fuerte también.

Tras la petición de recolección, R posee toda la información del árbol, y ahora debe pasarla al resto de nodos mediante una petición de distribución. Esta petición de distribución desde la raíz fuerte R provocará un paso de mensajes que terminará al llegar a las hojas. Cuando esta distribución termine, el árbol será localmente consistente, ya que todos los potenciales de los separadores serán marginales fuertes de potenciales que están más allá de la raíz fuerte. Al ejecutar la petición de distribución, la proposición 3.3 asegura que todas las absorciones cumplen la propiedad (3.11) y por tanto el árbol es locamente consistente con un potencial que tiene la siguiente propiedad, cuya demostración puede encontrarse en (Cowell, Dawid, Lauritzen, y Spiegelhalter 1999).

Teorema 3.1 *Sea $\mathcal{T}$ un árbol de cliques localmente consistente y con una raíz fuerte R y conjunto de cliques $\mathcal{C}$. Sea ϕ_V el potencial conjunto de $\mathcal{T}$ y sea $U \in \mathcal{C}$. Entonces*

$$\sum_{V \setminus U} \phi_V \propto \phi_U$$

Es decir, cada potencial de cada clique es la marginal (débil) del potencial total.

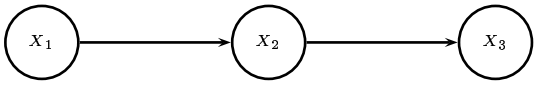
En resumen, después de introducir la evidencia, el árbol se puede hacer consistente enviando sendas peticiones de distribución y recolección desde una raíz fuerte. La marginal débil del potencial de cualquier variable de la red se puede obtener de cualquier clique del árbol que contenga a esa variable, simplemente mediante la operación marginalización débil. En concreto, esto proporciona las probabilidades correctas actualizadas de los estados de cualquier variable discreta y las medias y varianzas correctas actualizadas de cualquier variable continua.

A pesar de que se usan marginales débiles, el árbol contiene todavía después de la propagación una representación correcta del potencial total del sistema dada la evidencia, es decir, se cumple que,

$$f(\mathbf{x}) = \frac{\prod_{C \in \mathcal{C}} f^{\downarrow C}(\mathbf{x}_C)}{\prod_{S \in \mathcal{S}} f^{\downarrow S}(\mathbf{x}_S)}$$

por tanto no se pierde información en la propagación de la evidencia y el sistema queda preparado para una nueva incorporación de evidencia.

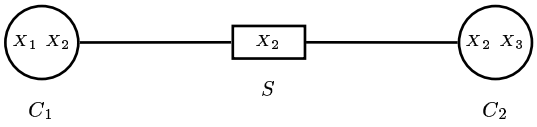
Ejemplo 3.1 *(Cowell, Dawid, Lauritzen, y Spiegelhalter 1999) Para ilustrar esta cuestión, veremos un sencillo ejemplo de un modelo puramente continuo. Consideremos la siguiente red:*



cuyas distribuciones iniciales son :

$$\begin{aligned} X_1 &\rightarrow \mathcal{N}(0,1) \\ X_2|X_1 &\rightarrow \mathcal{N}(x_1,1) \\ X_3|X_2 &\rightarrow \mathcal{N}(x_2,1) \end{aligned}$$

El árbol de cliques en este caso es



y cualquier clique se puede coger como raíz. Sabemos las distribuciones condicionadas de cada variable, veamos cuales son los correspondientes potenciales CG:

1. ***Para*** X_1*: Para esta variable tendremos un potencial de la forma* $\phi_1(x_1)$ *que lo representaremos por sus características canónicas, que serán* (g_1, h_1, K_1)*. Ahora*

bien, como X_1 es continua, eso quiere decir que $g_1 = 0$ (y lo mismo para las otras variables, siempre tendremos $g = 0$). La distribución de X_1 es una $\mathcal{N}(0,1)$, luego, por (3.6) deducimos que[2]

$$\alpha = 0 \quad ; \quad \gamma = 1$$

Ponemos solamente α y no $\alpha(y)$ dado que no tenemos variables discretas. Con estos valores, y a partir de (3.7) obtenemos que[3]

$$g_1 = 0$$

$$h_1 = \frac{\alpha}{\gamma}(1) = 0(1) = 0$$

$$K_1 = \frac{1}{\gamma}(1) = 1(1) = 1$$

Luego $\phi(x_1) \propto \exp\{-\frac{1}{2}x_1(1)x_1\}$

2. ***Para** X_2: Igual que con ϕ_1, en este caso $\phi_2(x_1, x_2)$ vendrá dado por sus características canónicas, $(0, h_2, K_2)$, e igualmente de $\mathcal{N}(x_1, 1)$ y de (3.6) obtenemos que*

$$\alpha = 0 \quad ; \quad \beta = 1 \quad ; \quad \gamma = 1$$

y por tanto a partir de (3.7) recuperamos las características canónicas

$$g_2 = 0$$

$$h_2 = \frac{\alpha}{\gamma}\begin{pmatrix} 1 \\ -\beta \end{pmatrix} = 0\begin{pmatrix} 1 \\ -1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$K_2 = \frac{1}{\gamma}\begin{pmatrix} 1 & -\beta \\ -\beta & \beta\beta \end{pmatrix} = 1\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

[2] No hay β al no ser condicionada esta distribución

[3] En realidad K debería ser $\begin{pmatrix} 1 & -\beta^T \\ -\beta & \beta\beta^T \end{pmatrix}$ pero como no hay padres la tomamos como una matriz unidad (1), al igual que hacemos con h

$$\textit{Luego } \phi_2(x_1, x_2) \propto \exp\left\{ -\tfrac{1}{2} \begin{pmatrix} x_1 & x_2 \end{pmatrix} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\}$$

3. ***Para*** X_3*: Si repetimos lo mismo que con las dos variables anteriores obtendremos que*

$$g_3 = 0$$

$$h_3 = \frac{\alpha}{\gamma} \begin{pmatrix} 1 \\ -\beta \end{pmatrix} = 0 \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$K_3 = \frac{1}{\gamma} \begin{pmatrix} 1 & -\beta \\ -\beta & \beta\beta \end{pmatrix} = 1 \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

$$\textit{Luego } \phi_3(x_2, x_3) \propto \exp\left\{ -\tfrac{1}{2} \begin{pmatrix} x_2 & x_3 \end{pmatrix} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} x_2 \\ x_3 \end{pmatrix} \right\}$$

Ahora el paso siguiente es asignar variables a cliques, de forma que el clique contenga a la variable y a sus padres, luego la única asignación que podemos hacer es

$$\begin{aligned} X_1 \ , \ X_2 &\rightarrow C_1 \\ X_3 &\rightarrow C_2 \end{aligned}$$

y combinar los distintos potenciales para obtener los potenciales asociados a los cliques.

$$\begin{aligned} \phi_{C_1}(x_1, x_2) &= \phi_1(x_1)\phi_2(x_1, x_2) \\ \phi_{C_2}(x_2, x_3) &= \phi_3(x_2, x_3) \end{aligned}$$

Las características canónicas de ϕ_{C_1} vendrán dadas por la suma de las características de los potenciales ϕ_1 y ϕ_2, esto es[4]

$$(0, h_{C_1}, K_{C_1}) = (0+0, h_1+h_2, K_1+K_2) = (0, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 2 & -1 \\ -1 & 1 \end{pmatrix})$$

y las características de ϕ_{C_2} son, obviamente, las mismas que las de ϕ_3.

Como sabemos, el potencial del separador lo inicializamos a 1. Supongamos primero que no hay evidencia, entonces estamos ya en condiciones de empezar a propagar. En este caso es muy sencillo, sólo hay dos mensajes, uno que va de C_1 a C_2 y otro de C_2 a C_1.

1. ***Mensaje de** C_1 **a** C_2: En realidad este mensaje lo que provoca es que C_2 absorba de C_1. Primero hemos de calcular el nuevo potencial del separador, ϕ_S^**

$$\phi_S^*(x_2) = \int \phi_{C_1}(x_1, x_2) dx_1$$

que, según la proposición 3.1, tendrá características canónicas (g_S^*, h_S^*, K_S^*)

$$g_S^* = 0$$

$$h_S^* = 0 - (-1)(2)^{-1}0 = 0$$

$$K_S^* = 1 - (-1)(2)^{-1}(-1) = 1 - \frac{1}{2} = \frac{1}{2}$$

luego $\phi_S^*(x_2) = \exp\left\{-\frac{1}{2}x_2(\frac{1}{2})x_2\right\} = \exp\left\{-\frac{1}{4}x_2^2\right\}$

Sabemos que el nuevo potencial de C_2 viene dado por

$$\phi_{C_2}^* = \phi_{C_2}\frac{\phi_S^*}{\phi_S} = \phi_{C_2}\phi_S^*$$

luego tendrá por caraterísticas canónicas la suma de las características de ϕ_{C_2} y ϕ_S^*:

$$(0, h_{C_2}^*, K_{C_2}^*) = (0, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0.5 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}) = (0, \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1.5 & -1 \\ -1 & 1 \end{pmatrix})$$

[4]Para sumar h_1 y h_2 (y también K_1 y K_2) lo que hacemos es extender el potencial $\phi_1(x_1)$ a otro $\phi_1(x_1, x_2)$

luego $\phi^*_{C_2}(x_2, x_3) = \exp\{-\frac{1}{2}\begin{pmatrix} x_2 & x_2 \end{pmatrix}\begin{pmatrix} 1.5 & -1 \\ -1 & 1 \end{pmatrix}\begin{pmatrix} x_2 \\ x_3 \end{pmatrix}\}$.

2. ***Mensaje de*** C_2 ***a*** C_1*: Ahora ya* ϕ^*_S *será el potencial del separador, no el original, sino el que acabamos de obtener, e igual con* C_1 *y* C_2*. Pues bien, de nuevo hay que calcular el nuevo potencial del separador,*ϕ^*_S*, pero ahora a partir de* C_2*:*

$$\phi^*_S(x_2) = \int \phi_{C_2}(x_2, x_3)dx_3$$

que usando la proposición 3.1 con los subindices adecuados, tiene características canónicas (g^*_S, h^*_S, K^*_S)

$$g^*_S = 0$$

$$h^*_S = 0$$

$$K^*_S = 1.5 - (-1)(1)^{-1}(-1) = 1.5 - 1 = \frac{1}{2}$$

Como vemos, el separador permanece igual, no ha variado, luego el mensaje de C_2 *a* C_1 *no varía en nada la situación que ya tenemos, ya que* $\phi^*_{C_1} = \phi_{C_1}\frac{\phi^*_S}{\phi_S} = \phi_{C_1}$.

Introduzcamos ahora ***evidencia****, por ejemplo,* $X_2 = 1.5$*. Ahora lo que buscamos son las distribuciones marginales de* X_1 *y de* X_3 *dado que sabemos que* $X_2 = 1.5$*.*
Es claro que

$$X_3|X_2 = 1.5 \rightarrow \mathcal{N}(1.5, 1)$$

Veamos cual sería la distribucion $X_3|X_2 = 1.5$*. Sabemos por (3.10) que ahora el potencial* ϕ_{C_1} *quedaría con características canónicas* $(0, h^*_{C_1}, K^*_{C_1})$

$$h^*_{C_1} = 0 - 1.5(-1) = 1.5$$

$$K^*_{C_1} = 2$$

luego, por (3.2), el vector de medias, en este caso unitario, ξ *y la matriz de covarianzas, de nuevo unitaria,* Σ *valen:*

$$\xi = \frac{1}{2}\frac{3}{2} = \frac{3}{4}$$

$$\Sigma = \frac{1}{2}$$

luego $X_1|X_2 = 1.5 \rightarrow \mathcal{N}(\frac{3}{4}, \frac{1}{2})$

En este caso no podemos hacer nada más, es decir, no podemos propagar, ya que al conocer el valor de X_2*, las variables* X_1 *y* X_3 *se vuelven independientes.*

Veamos qué hubiese pasado, si en vez de conocer el valor de X_2 *hubiésemos sabido, por ejemplo, que* $X_3 = 1.5$*. Lo primero que hacemos es introducir la evidencia en el clique* C_2*. Por tanto, siguiendo las indicaciones (3.10) obtenemos que el potencial* $\phi_{C_2}(x_2, 1.5)$ *tiene características canónicas* $(0, h^*, K^*)$

$$h^* = 0 - 1.5(-1) = 1.5$$

$$K^* = 1.5$$

Ahora propagamos, sólo necesitamos enviar un mensaje de C_2 *a* C_1*, puesto que sólo se ha actualizado la información en el segundo potencial. Además, el nuevo potencial del separador,* $\phi_S^*(x_2)$ *será exactamente el mismo que el de* ϕ_{C_2}*, ya que ahora* ϕ_{C_2} *sólo depende de* X_2*. Por tanto, el nuevo potencial de* C_1 *se calcula mediante la expresión* $\phi_{C_1}^* = \phi_{C_1}\frac{\phi_S^*}{\phi_S}$*.*

Calculemos primero el cociente $\frac{\phi_S^*}{\phi_S}$*, que tendrá por características canónicas la resta de las caracteristicas canónicas de los potenciales involucrados, es decir,*

$$\frac{\phi_S^*}{\phi_S} \equiv (g, h, K) = (0, 1.5, 1.5) - (0, 0, 0.5) = (0, 1.5, 1)$$

luego $\phi_{C_1}\frac{\phi_S^*}{\phi_S}$ *tendrá ahora por características canónicas* $(g_{C_1}^*, h_{C_1}^*, K_{C_1^*})$ *la suma de las características canónicas de los potenciales correspondientes, por supuesto extendiendo de forma adecuada el potencial del cociente entre los potenciales de los separadores para poder realizar la operación:*

$$\begin{aligned}(g^*_{C_1}, h^*_{C_1}, K_{C_1^*}) &= (0, \begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}2 & -1\\-1 & 1\end{pmatrix}) + (0, \begin{pmatrix}0\\1.5\end{pmatrix}, \begin{pmatrix}0 & 0\\0 & 1\end{pmatrix})\\ &= (0, \begin{pmatrix}0\\1.5\end{pmatrix}, \begin{pmatrix}2 & -1\\-1 & 2\end{pmatrix}) \ .\end{aligned}$$

Así pues, quedaría

$$\phi^*_{C_1}(x_1, x_2) = \exp\left\{ \begin{pmatrix}x_1 & x_2\end{pmatrix} \begin{pmatrix}0\\1.5\end{pmatrix} - \frac{1}{2} \begin{pmatrix}x_1 & x_2\end{pmatrix} \begin{pmatrix}2 & -1\\-1 & 2\end{pmatrix} \begin{pmatrix}x_1\\x_2\end{pmatrix} \right\}$$

Por tanto, para calcular la distrbución marginal de X_2 dado que $X_3 = 1.5$ sólo hay que fijarse en $\phi^*_{C_2}$, que ahora sólo depende de X_2, y por tanto, usando (3.2) tenemos que

$$\xi = K^{-1}h = (\frac{3}{2})^{-1}\frac{3}{2} = 1$$

$$\Sigma = K^{-1} = (\frac{3}{2})^{-1} = \frac{2}{3}$$

y así $X_2|X_3 = 1.5 \to \mathcal{N}(1, \frac{2}{3})$

Para hacer lo mismo con X_1 hay que calcular la marginal de $\phi^*_{C_1}$ sobre X_2. Así, usando la proposición 3.1, el potencial $\int \phi^*_{C_1}(x_1, x_2)dx_2$ tendrá por características canónicas los valores

$$h^* = 0 - (-1)(2)^{-1}1.5 = \frac{1}{2}\frac{3}{2} = \frac{3}{4}$$

$$K^* = 2 - (-1)(2)^{-1}(-1) = 2 - \frac{1}{2} = 1.5$$

y así tenemos que su media y su varianza valen

$$\xi = K^{-1}h = (\frac{3}{2})^{-1}\frac{3}{4} = \frac{1}{2}$$

$$\Sigma = K^{-1} = (\frac{3}{2})^{-1} = \frac{2}{3}$$

y por tanto $X_1|X_3 = 1.5 \to \mathcal{N}(\frac{1}{2}, \frac{2}{3})$

3.5. Conclusiones

Hemos descrito un método para tratamiento de variables continuas basado en el uso de distribuciones condicionadas Gaussianas. Entre las propiedades más importantes de este modelo podemos destacar que permite una propagación exacta. Aquí hemos descrito un algoritmo para propagar medias y varianzas, pero es posible generalizarlo considerando potenciales que sean mixturas de distribuciones Gaussianas. También es importante destacar que puede representar una distribución condicionada de una variable a dado un conjunto de variables con un número reducido de parámetros (una función lineal de las variables a las que condicionamos y un valor constante para la varianza). Las distribuciones Gaussianas aparecen en muchos procesos naturales e introduciendo variables discretas artificiales se pueden aproximar distribuciones no Gaussianas mediante mixturas de estas distribuciones. En el lado negativo de este modelo, podemos indicar que a menudo nos encontramos con variables que no pueden considerarse Gaussianas, la dificultad de representar relaciones no lineales entre las variables continuas, la restricción de que una variable discreta no puede ser hija de una variable continua y el hecho de que al exigir grafos fuertemente triangulados se puede incrementar notablemente la complejidad de los cálculos.

Capítulo 4

Mixtura de Exponenciales Truncadas

4.1. Introducción

En los capítulos anteriores hemos visto dos enfoques diferentes desde los que se ha abordado en la literatura el tratamiemto de redes híbridas: uno (dsicretización) se basa es intentar aprovechar la teoría desarrollada para redes discretas, y aplicarla transformando para ello los potenciales continuos en discretos, y el otro (CG) se basa en realizar las operaciones de propagación de forma exacta, en el caso concreto de que el modelo siga la distribución adecuada, lo que en el caso de análisis de datos reales no siempre ocurrirá. Además, el modelo CG presenta la restricción de que las variables discretas no pueden tener padres continuos.

Para solucionar este problema, Koller, Lerner, y Anguelov (1999) modelizan la distribución de nodos discretos con padres continuos mediante una mixtura de exponenciales, pero después los métodos de inferencia son de tipo Monte Carlo en lugar de exactos.

Lo que se propone en este capítulo es usar una *mixtura de exponenciales truncadas* para representar la distribución de las variables en la red (Moral, Rumí, y Salmerón

2001). Con este modelo no hay restricciones con las relaciones de las variables en la red, ya que se permite a las variables continuas tener hijos discretos, y además es posible la propagación exacta mediante algoritmos de cálculo local como los vistos en el primer capítulo.

Fundamentalmente, este modelo es una alternativa a la discretización. La discretización (capítulo 2) la podemos ver como aproximar una densidad por una mixtura de uniformes, pero pensamos que se obtendrán mejores aproximaciones si en vez de usar funciones uniformes usamos funciones exponenciales, que tienen un mayor poder de adaptación a la forma de la distribución a aproximar.

Pero el modelo de mixturas de exponenciales truncadas también tiene sentido como distribución exacta, pues veremos que algunas distribuciones notables, como la uniforme (discreta y continua), la multinomial y la exponencial son casos particulares suyos.

4.2. Definiciones

Introducimos una clase de distribuciones mixtas, que llamaremos *mixtura de exponenciales truncadas*, pero antes de definir la distribución estudiaremos el potencial asociado a esta familia de distribuciones y las operaciones básicas sobre él.

Definición 4.1 *Sea* $\mathbf{X}$ *una variable aleatoria n-dimensional. Sean* $\mathbf{Y} = (Y_1, \ldots, Y_d)$ *y* $\mathbf{Z} = (Z_1, \ldots, Z_c)$ *las partes discreta y continua de* $\mathbf{X}$ *respectivamente, con* $c + d = n$. *Decimos que una función* $\phi : \Omega_X \rightarrow \mathbb{R}_0^+$ *es un potencial de la clase* ***mixtura de exponenciales truncadas*** *(Potencial MTE) si se verifica una de las siguientes dos condiciones:*

i. ϕ *es de la forma*

$$\phi(\mathbf{x}) = \phi(\mathbf{y}, \mathbf{z}) = a_0 + \sum_{i=1}^{m} a_i \exp\left\{\sum_{j=1}^{d} b_i^{(j)} y_j + \sum_{k=1}^{c} b_i^{(d+k)} z_k\right\} \qquad (4.1)$$

para cualquier $\mathbf{x} \in \Omega_{\mathbf{x}}$*, donde* $a_i,\ i = 0, \ldots, m$ *y* $b_i^{(j)},\ i = 1, \ldots, m,\ j = 1, \ldots, n$ *son números reales.*

ii. Existe una partición $\Omega_1, \ldots, \Omega_k$ *de* $\Omega_{\mathbf{X}}$ *tal que el dominio de las variables continuas,* $\Omega_{\mathbf{Z}}$*, esté dividido en hipercubos y tal que* ϕ *venga definida por*

$$\phi(\mathbf{x}) = \phi_i(\mathbf{x}) \quad si \quad \mathbf{x} \in \Omega_i,$$

donde cada $\Omega_i,\ i = 1, \ldots, k$ *es de la forma (4.1) (es decir, cada* ϕ_i *es un potencial MTE sobre* Ω_i*).*

Ejemplo 4.1 *La función* ϕ *definida como*

$$\phi(z_1, z_2) = \begin{cases} 2 + e^{3z_1+z_2} + e^{z_1+z_2} & \text{si} \quad 0 < z_1 \leq 1,\ 0 < z_2 < 2 \\ 1 + e^{z_1+z_2} & \text{si} \quad 0 < z_1 \leq 1,\ 2 \leq z_2 < 3 \\ \frac{1}{4} + e^{2z_1+z_2} & \text{si} \quad 1 < z_1 < 2,\ 0 < z_2 < 2 \\ \frac{1}{2} + 5e^{z_1+2z_2} & \text{si} \quad 1 < z_1 < 2,\ 2 \leq z_2 < 3 \end{cases}$$

es un potencial MTE ya que todas sus partes son potenciales MTE.

Como sabemos, a la hora de llevar a cabo la propagación sobre una red bayesiana, es necesario realizar ciertas operaciones sobre los potenciales definidos en cada uno de los cliques, como son restricción, marginalización, y producto o combinación. Veamos cómo realizar estas operaciones sobre potenciales MTE:

Definición 4.2 *Sea* ϕ *un potencial MTE sobre* $\mathbf{X} = (\mathbf{Y}, \mathbf{Z})$*. Supongamos un conjunto de variables* $\mathbf{X}' = (\mathbf{Y}', \mathbf{Z}') \subseteq \mathbf{X}$*, cuyos valores* $\mathbf{x}^{\downarrow \Omega_{\mathbf{X}'}}$ *están fijados (*$\mathbf{x}^{\downarrow \Omega_{\mathbf{X}'}} = \mathbf{x}' = (\mathbf{y}', \mathbf{z}')$*). La* ***restricción*** *de* ϕ *a los valores* $(\mathbf{y}', \mathbf{z}')$ *es un nuevo potencial definido sobre* $\Omega_{\mathbf{X} \setminus \mathbf{X}'}$ *de acuerdo con la siguiente expresión:*

$$\phi^{R(\mathbf{X}'=\mathbf{x}')}(\mathbf{w}) = \phi^{R(\mathbf{Y}'=\mathbf{y}', \mathbf{Z}'=\mathbf{z}')}(\mathbf{w}) = \phi(\mathbf{x})$$

para todos aquellos $\mathbf{w} \in \Omega_{\mathbf{X} \setminus \mathbf{X}'}$ *tales que* $\mathbf{x} \in \Omega_{\mathbf{X}}$*,* $\mathbf{x}^{\downarrow \Omega_{\mathbf{X} \setminus \mathbf{X}'}} = \mathbf{w}$ *y* $\mathbf{x}^{\downarrow \Omega_{\mathbf{X}'}} = \mathbf{x}'$*. En otras palabras, la restricción es el potencial que se obtiene al sustituir cada ocurrencia de* $\mathbf{X}'$ *por su valor* $\mathbf{x}'$*.*

Ejemplo 4.2 *Consideremos el potencial del ejemplo 4.1. Supongamos que queremos restringirlo a* $Z_1 = 1.5$. *Lo conseguimos seleccionando el intervalo apropiado y reemplazando* z_1 *por el valor* 1.5. *El resultado es*

$$\phi^{R(Z_1=1.5)}(z_2) = \begin{cases} \frac{1}{4} + e^{2\times 1.5+z_2} & 0 < z_2 < 2 \\ \frac{1}{2} + e^{1.5+z_2} & 2 \leq z_2 < 3 \end{cases} = \begin{cases} \frac{1}{4} + e^3 e^{z_2} & 0 < z_2 < 2 \\ \frac{1}{2} + 5e^{1.5} e^{z_2} & 2 \leq z_2 < 3 \end{cases}$$

Definición 4.3 *Sea* ϕ *un potencial MTE sobre* $\mathbf{X} = (\mathbf{Y}, \mathbf{Z})$. *La* ***marginal*** *de* ϕ *para un conjunto de variables* $\mathbf{X'} = (\mathbf{Y'}, \mathbf{Z'}) \subseteq \mathbf{X}$ *es el potencial :*

$$\phi^{\downarrow \mathbf{X'}}(\mathbf{y'}, \mathbf{z'}) = \sum_{\mathbf{y} \in \Omega_{\mathbf{Y} \setminus \mathbf{Y'}}} \left(\int_{\Omega_{\mathbf{Z''}}} \phi(\mathbf{y}, \mathbf{y'}, \mathbf{z}, \mathbf{z'}) d\mathbf{z''} \right)$$

donde $\mathbf{Z''} = \mathbf{Z} \setminus \mathbf{Z'}$. *Obsérvese que esta función está definida sobre* $\Omega_{\mathbf{X'}}$.

Ejemplo 4.3 *Supongamos que queremos obtener la marginal del potencial del ejemplo 4.1 para la variable* Z_2. *Sólo hay que integrar sobre* Z_1*:*

$$\phi^{\downarrow Z_2}(z_2) = \begin{cases} \int_0^1 2 + e^{3z_1+z_2} + e^{z_1+z_2} dz_1 + \int_1^2 \frac{1}{4} + e^{2z_1+z_2} dz_1 & 0 < z_2 < 2 \\ \int_0^1 1 + e^{z_1+z_2} dz_1 + \int_1^2 \frac{1}{2} + 5e^{z_1+2z_2} dz_1 & 2 \leq z_2 < 3 \end{cases}$$

y simplificando la expresión de arriba obtenemos:

$$\phi^{\downarrow z_2}(z_2) = \begin{cases} \frac{9}{4} + \frac{3e^4+2e^3-3e^2+6e-8}{6} e^{z_2} & 0 < z_2 < 2 \\ \frac{3}{2} + (e-1)e^{z_2} + 5(e^2-e)e^{z_2} & 2 \leq z_2 < 3 \end{cases}$$

Definición 4.4 *Sean* ϕ_1 *y* ϕ_2 *potenciales MTE sobre* $\mathbf{X_1} = (\mathbf{Y_1}, \mathbf{Z_1})$ *y* $\mathbf{X_2} = (\mathbf{Y_2}, \mathbf{Z_2})$ *respectivamente. La* ***combinación*** *de* ϕ_1 *y* ϕ_2 *es un nuevo potencial definido sobre* $\mathbf{X} = \mathbf{X_1} \cup \mathbf{X_2}$ *que se calcula como:*

$$\phi(\mathbf{x}) = \phi_1(\mathbf{x}^{\downarrow \Omega_{\mathbf{X_1}}}) \cdot \phi_2(\mathbf{x}^{\downarrow \Omega_{\mathbf{X_2}}}) \quad \textit{para todo } \mathbf{x} \in \Omega_{\mathbf{X}} \ .$$

Ejemplo 4.4 *Para ilustrar esta operación, combinaremos el potencial del ejemplo 4.1 con este otro:*

$$\phi'(x) = \begin{cases} 3 + e^{x} & 0 < x \leq 2 \\ 2 + e^{-2x} & 2 < x < 4 \end{cases}$$

El resultado será un potencial MTE $\phi''(z_1, z_2, x) = \phi(z_1, z_2) \cdot \phi'(x)$ *definido en 8 regiones :*

$$\phi''(z_1, z_2, x) = \begin{cases} 6 + 2e^{x} + 3e^{z_1+z_2} + 3e^{3z_1+z_2} + e^{z_1+z_2+x} + e^{3z_1+z_2+x} \\ \text{si } 0 < z_1 \leq 1, 0 < z_2 < 2, 0 < x \leq 2 \\ 4 + 10e^{-2x} + 2e^{3z_1+z_2} + 2e^{z_1+z_2} + 5e^{3z_1+z_2-2x} + 5e^{z_1+z_2-2x} \\ \text{si } 0 < z_1 \leq 1, 0 < z_2 < 2, 2 < x < 4 \\ 3 + e^{x} + 3e^{z_1+z_2} + e^{z_1+z_2+x} \\ \text{si } 0 < z_1 \leq 1, 2 \leq z_2 < 3, 0 < x \leq 2 \\ 2 + 5e^{-2x} + e^{z_1+z_2} + 5e^{z_1+z_2-2x} \\ \text{si } 0 < z_1 \leq 1, 2 \leq z_2 < 3, 2 < x < 4 \\ \frac{3}{4} + \frac{e^{x}}{4} + 3e^{2z_1+z_2} + e^{2z_1+z_2+x} \\ \text{si } 1 < z_1 < 2, 0 < z_2 < 2, 0 < x \leq 2 \\ \frac{1}{2} + \frac{5e^{-2x}}{4} + 2e^{2z_1+z_2} + 5e^{2z_1+z_2-2x} \\ \text{si } 1 < z_1 < 2, 0 < z_2 < 2, 2 < x < 4 \\ \frac{3}{2} + \frac{e^{x}}{2} + 15e^{z_1+2z_2} + 5e^{z_1+2z_2+x} \\ \text{si } 1 < z_1 < 2, 2 \leq z_2 < 3, 0 < x \leq 2 \\ 1 + \frac{5e^{-2x}}{2} + 10e^{z_1+2z_2} + 25e^{z_1+2z_2-2x} \\ \text{si } 1 < z_1 < 2, 2 \leq z_2 < 3, 2 < x < 4 \end{cases}$$

Proposición 4.1 *La clase de potenciales MTE es cerrada bajo las operaciones de restricción, marginalización y combinación.*

Demostración:

1. *Restricción.* Cuando restringimos un potencial MTE a un valor w_j de una variable W_j, discreta o continua, simplemente multiplicamos el correspondiente coeficiente a_i de cada exponencial afectada por la variable W_j por su constante $exp\{b_i^{(j)} y_i\} \quad \forall i = 1, \ldots, m$ y entonces eliminamos el término que contiene a la variable W_j del exponente.
   Si la variable a la que restringimos es la única en un término exponencial dado, entonces ese término es eliminado y se suma a la constante a_0 las constantes $exp\{b_i^{(j)} y_i\} \quad \forall i = 1, \ldots, m$. Así, el resultado es otro potencial MTE con diferentes coeficientes y sin contener a la variable W_j.

2. *Marginalización.* De acuerdo con la ecuación (4.1), marginalizar sobre un conjunto de variables continuas $\mathbf{Z}'$ es equivalente a multiplicar cada coeficiente a_i de cada exponencial afectada por cualquier variable $W_j \in \mathbf{Z} \setminus \mathbf{Z}'$ por su constante $\int_{\Omega_{Z_k}} \exp\{b_i^{d+k} z_k\} dz_k \quad \forall i = 1, \ldots, m$ y así obtenemos otro potencial MTE. La marginalización sobre un conjunto de varibles discretas $\mathbf{Y}'$ se calcula sumando potenciales MTE , y el resultado es obviamente otro potencial MTE.

3. *Combinación.* El producto de dos potenciales del tipo mostrado en (4.1) es otro potencial del mismo tipo, ya que el producto de funciones exponenciales da como resultado otra función exponencial aplicada a la suma de los exponentes, así que la combinación de dos potenciales MTE, que únicamente es productos de los coeficientes y sumas de las exponenciales de ambos potenciales, es obviamente un potencial MTE.

■

Ahora ya si podemos definir una distribución de probabilidad basada en los potenciales MTE.

Definición 4.5 *Sea* $\mathbf{X} = (\mathbf{Y}, \mathbf{Z})$ *una variable aleatoria mixta n-dimensional. Decimos*

que $\mathbf{X}$ *sigue una distribución MTE, si su densidad* f *es un potencial MTE y*

$$\sum_{\mathbf{y}\in\Omega_{\mathbf{Y}}}\int_{\Omega_{\mathbf{Z}}} f(\mathbf{y},\mathbf{z})d\mathbf{z} = 1$$

Un potencial verificando estas condiciones lo notaremos por densidad MTE para $\mathbf{X}$.

Pero como vimos, en una red Bayesiana, el modelo viene en términos de un conjunto de distribuciones condicionales, en vez de distribuciones conjuntas. La distribución condicional MTE la definimos como:

Definición 4.6 *Sean* $\mathbf{X_1} = (\mathbf{Y_1}, \mathbf{Z_1})$ *y* $\mathbf{X_2} = (\mathbf{Y_2}, \mathbf{Z_2})$ *dos variables aleatorias mixtas. Decimos que un potencial MTE* f *definido sobre* $\Omega_{\mathbf{X_1}\cup\mathbf{X_2}}$ *es una densidad condicional MTE si para cada* $\mathbf{x_2} \in \Omega_{\mathbf{X_2}}$, *se verifica que* $f^{R(\mathbf{X_2}=\mathbf{x_2})}$ *es una densidad MTE para* $\mathbf{X_1}$.

Lo que hace que las distribuciones MTE sean dignas de estudio es su versatilidad; muchos modelos se pueden aproximar por una distribución MTE, pero además, algunos modelos son casos particulares de las MTE, como por ejemplo el modelo uniforme y la multinomial.

Proposición 4.2 *Sea* X *una variable aleatoria continua cuya distribución es una uniforme* $\mathcal{U}(a,b)$. *Entonces* X *sigue una distribución MTE.*

Demostración: la demostración es trivial, un potencial MTE definido para la variable X definido sobre un único intervalo $[a,b]$, tal que no tenga ningún término exponencial y el término independiente sea $\dfrac{1}{b-a}$.

■

Proposición 4.3 *Sea* X *una variable aleatoria discreta cuya distribución es una multinomial de parámetros* $M(1, p_1, \ldots, p_n)$. *Entonces* X *sigue una distribución MTE.*

Demostración: El potencial

$$f(x) = \begin{cases} p_1 & \text{Si } X = x_1 \\ \dots & \dots \\ p_i & \text{Si } X = x_i \\ \dots & \dots \\ p_n & \text{Si } X = x_n \end{cases}$$

es un potencial MTE definido sobre la variable X que representa su distribución de probabilidad.

■

Cuando especificamos una red bayesiana, damos un conjunto de distribuciones condicionadas que forman una factorización de la distribución conjunta de todas las variables de la red, como vimos en el primer capítulo. Si las distribuciones condicionadas son MTE, entonces con la siguiente proposición demostramos que la distribución conjunta es también de clase MTE.

Proposición 4.4 *Sea $\mathcal{G}$ una red bayesiana sobre una variable aleatoria mixta n-dimensional $\mathbf{X} = (\mathbf{Y}, \mathbf{Z})$. Si cada distribución condicionada asociada con $\mathcal{G}$ es una densidad condicional MTE, entonces la distribución conjunta sobre $\mathbf{X}$ se puede representar mediante una densidad MTE.*

Demostración: Sabemos que el producto de la n densidades de una red es la distribución conjunta de la variable n-dimensional $\mathbf{X}$, y también sabemos que el producto de todas las densidades MTE asociadas con la red bayesiana es otro potencial MTE, luego la densidad conjunta es una densidad MTE.

■

Una vez que hemos definido el tipo de potenciales que tendrán las redes mixtas MTE, queda definir el tipo de estructura de datos que usaremos para representarlos:

4.3. Árboles de probabilidad mixtos

En el primer capítulo la representación que se usó para los potenciales es la de matrices de números reales, es decir, la representación clásica de las tabla de probabilidad.

El principal problema de las tablas es que no permiten reflejar algunas de las posibles regularidades presentes en los potenciales; en la mayoría de los casos, no permiten sacar partido de la aparición de valores repetidos. Esto sugirió el estudio de representaciones dispersas, que permitan de alguna forma, representar más información en menos espacio, como es el caso de los *árboles de probabilidad* (Bouckaert, Castillo, y Gutiérrez 1996; Cano y Moral 1997). En general se pretende utilizar las posibles regularidades presentes en las distribuciones condicionadas de la red para simplificarla de cara a facilitar la labor de los algoritmos de inferencia.

Un árbol de probabilidad es un árbol dirigido etiquetado en el que cada nodo interior representa una variable, y cada nodo hoja un valor de probabildad. Cada nodo interior tendrá tantos arcos salientes como casos tenga la variable que representa. El número real almacenado en cada hoja es el valor del potencial representado por el árbol para la configuración de las variables que se corresponde con el camino desde la raíz hasta la hoja. Los árboles de probabilidad se han mostrado como herramientas adecuadas para representar *independencias basadas en el contexto* (Bouckaert, Castillo, y Gutiérrez 1996).

Los árboles de probabilidad pueden utilizarse para representar cualquier potencial, y no solamente distribuciones condicionadas. Por lo tanto, podremos emplearlos para representar los potenciales que surgen en los pasos intermedios de los algoritmos de propagación, que no siempre son distribuciones de probabilidad. Para poder realizar estos algoritmos son necesarias las operaciones de restricción, combinación y marginalización, que se definen de forma natural (Cano y Moral 1997) sobre los árboles de probabilidad, como veremos posteriormente.

Para representar y operar con potenciales MTE, usamos una versión ampliada de los árboles de probabilidad, que llamaremos, *árboles de probabilidad mixtos*.

Definición 4.7 *Diremos que un árbol $\mathcal{T}$ es un árbol de probabilidad mixto si se cumplen las siguientes condiciones:*

i. Cada nodo interno representa una variable aleatoria (discreta o continua).

ii. Cada arco saliente de una variable continua Z tiene por etiqueta un intervalo de valores de Z, de forma que el dominio de la variable es la unión de los intervalos correspondientes a los arcos salientes.

iii. Cada variable discreta tiene un arco saliente para cada uno de sus casos.

iv. Cada nodo hoja contiene un potencial MTE definido sobre las variables en el camino de la raíz a dicha hoja.

Definición 4.8 *Definimos la etiqueta de un nodo de un árbol de probabilidad mixto como la variable aleatoria que representa, si el nodo es interno, o el potencial MTE que contiene si se trata de un nodo hoja.*

Los árboles de probabilidad mixtos pueden representar potenciales MTE definidos a trozos. Cada rama completa en el árbol determina una subregión del espacio donde el potencial es definido, y la función almacenada en la hoja de una rama es la definición del potencial en la correspondiente subregión.

Ejemplo 4.5 *Consideremos el siguiente potencial MTE, definido para una variable*

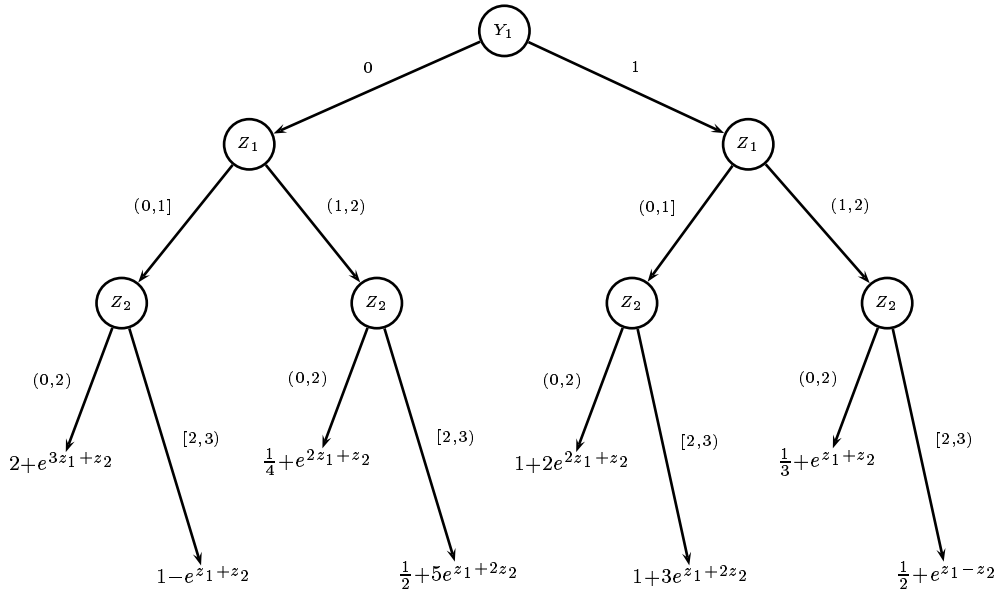


Figura 4.1: árbol de probabilidad mixto representando al potencial de la fórmula (4.2).

discreta, Y_1, y dos variables continuas, (Z_1,Z_2).

$$\phi(y_1, z_1, z_2) = \begin{cases} 2 + e^{3z_1+z_2} & \text{si } \; y_1 = 0,\ 0 < z_1 \leq 1,\ 0 < z_2 < 2 \\ 1 - e^{z_1+z_2} & \text{si } \; y_1 = 0,\ 0 < z_1 \leq 1,\ 2 \leq z_2 < 3 \\ \frac{1}{4} + e^{2z_1+z_2} & \text{si } \; y_1 = 0,\ 1 < z_1 < 2,\ 0 < z_2 < 2 \\ \frac{1}{2} + 5e^{z_1+2z_2} & \text{si } \; y_1 = 0,\ 1 < z_1 < 2,\ 2 \leq z_2 < 3 \\ 1 + 2e^{2z_1+z_2} & \text{si } \; y_1 = 1,\ 0 < z_1 \leq 1,\ 0 < z_2 < 2 \\ 1 + 3e^{z_1+z_2} & \text{si } \; y_1 = 1,\ 0 < z_1 \leq 1,\ 2 \leq z_2 < 3 \\ \frac{1}{3} + e^{z_1+z_2} & \text{si } \; y_1 = 1,\ 1 < z_1 < 2,\ 0 < z_2 < 2 \\ \frac{1}{2} + e^{z_1-z_2} & \text{si } \; y_1 = 1,\ 1 < z_1 < 2,\ 2 \leq z_2 < 3 \end{cases} \tag{4.2}$$

Una posible representación de este potencial en términos de árbol de probabilidad mixto la mostramos con la figura 4.1.

Las operaciones de restricción, marginalización y combinación sobre árboles de probabilidad mixtos se pueden llevar a cabo por medio de algoritmos muy similares a aquellos descritos en (Kozlov y Koller 1997; Salmerón, Cano, y Moral 2000).

Para definir la *restricción*, damos el concepto de *árbol restringido*.

Definición 4.9 *Sea $\mathcal{T}$ un árbol de probabilidad mixto y X una variable de $\mathcal{T}$.*

1. *Si X es discreta, el* **árbol restringido** *de $\mathcal{T}$ para un valor $x \in \Omega_X$, que notamos por $\mathcal{T}^{R(X=x)}$, es el árbol que se obtiene de $\mathcal{T}$ reemplazando cada nodo etiquetado con X por su hijo correspondiente al valor x.*

2. *Si X es continua, el* **árbol restringido** *de $\mathcal{T}$ para un intervalo $(a, b) \in \Omega_X$, que notamos por $\mathcal{T}^{R(X\in(a,b))}$, es el árbol que se obtiene de $\mathcal{T}$ repitiendo el siguiente proceso para cada nodo etiquetado con X:*

   - *Si hay un arco saliente de X etiquetado con un intervalo que contenga a (a, b), sustituimos X por el hijo de X correspondiente a ese arco.*
   - *Si no, eliminamos todos los hijos de X correspondientes a arcos etiquetados con intervalos cuya intersección con (a, b) sea vacía, y sustituimos las etiquetas de los arcos restantes por sus intersecciones con (a, b).*

Describimos a continuación, las otras dos operaciones, *combinación* y *marginalización*.

Dados dos árboles $\mathcal{T}_1$ y $\mathcal{T}_2$ representando a los potenciales ϕ_1 y ϕ_2 respectivamente, el siguiente algoritmo calcula un árbol que representa al potencial $\phi = \phi_1 \cdot \phi_2$ (*combinación*).

COMBINAR($\mathcal{T}_1$,$\mathcal{T}_2$)

ENTRADA: dos árboles de probabilidad mixtos $\mathcal{T}_1$ y $\mathcal{T}_2$.
SALIDA: la combinación de $\mathcal{T}_1$ y $\mathcal{T}_2$.

1. Crear un nodo $\mathcal{T}_r$ en principio sin etiqueta.
2. Sean L_1 y L_2 las etiquetas de los nodos raíz de $\mathcal{T}_1$ y $\mathcal{T}_2$ respectivamente.
3. Si L_1 y L_2 son potenciales MTE , entonces hacer que $L_1 \cdot L_2$ sea la etiqueta de $\mathcal{T}_r$.
4. Si L_1 es un potencial MTE pero L_2 es una variable,
   a) Hacer que L_2 sea la etiqueta de $\mathcal{T}_r$.
   b) Para cada árbol $\mathcal{T}$ hijo del nodo raíz de $\mathcal{T}_2$,
   hacer que $\mathcal{T}_h :=$ **COMBINAR(**$\mathcal{T}_1$**,**$\mathcal{T}$**)** sea un hijo de $\mathcal{T}_r$.
5. Si L_1 es una variable, supongamos que X es esa variable.
   a) Hacer que X sea la etiqueta de $\mathcal{T}_r$.
   b) Si X es discreta,
      1) Para cada $x \in \Omega_X$,
         - Hacer que $\mathcal{T}_h :=$ **COMBINAR(**$\mathcal{T}_1^{R(X=x)}$**,**$\mathcal{T}_2^{R(X=x)}$**)** sea un hijo de $\mathcal{T}_r$.
   c) Si X es continua,
      1) Para cada intervalo (a, b) correspondiente a arcos salientes de X,
         - Hacer que $\mathcal{T}_h :=$ **COMBINAR(**$\mathcal{T}_1^{R(X\in(a,b))}$**,**$\mathcal{T}_2^{R(X\in(a,b))}$**)** sea un hijo de $\mathcal{T}_r$.
6. DEVOLVER $\mathcal{T}_r$.

Ejemplo 4.6 *Combinemos los árboles representados en la figura 4.2. La secuencia de pasos para multiplicarlos viene representada en las figuras 4.3 y 4.4. Se restringen los dos árboles a cada uno de los intervalos correspondientes a los hijos del primer árbol, y se multiplican, hasta llegar a multiplicar dos funciones MTE. El resultado se puede ver en la figura 4.5.*

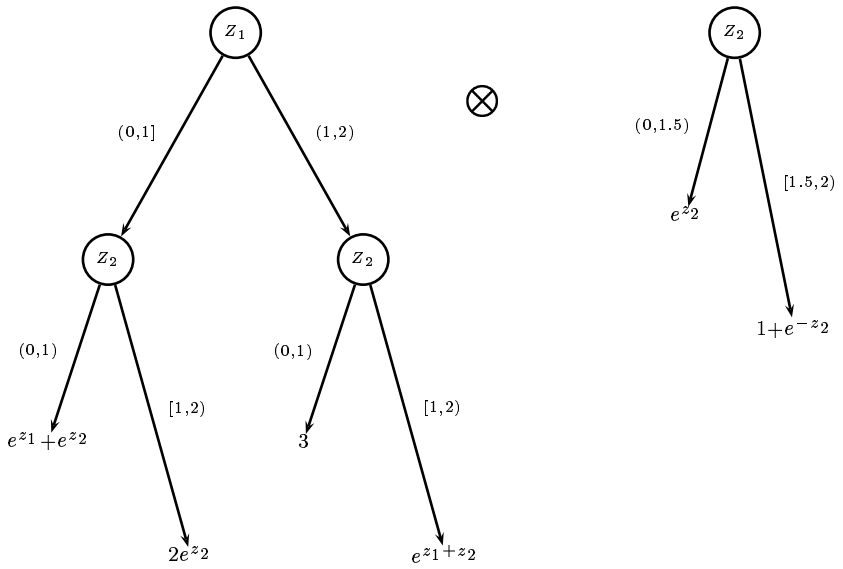


Figura 4.2: árboles a combinar.

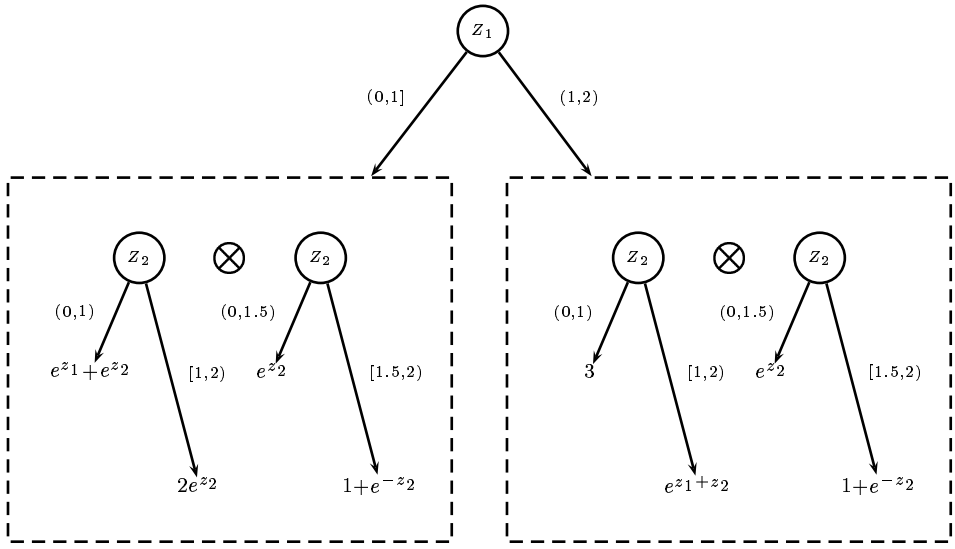


Figura 4.3: Paso 1: se combinan las restricciones de los árboles a $Z_1 \in (0, 1]$ y $Z_1 \in (1, 2)$.

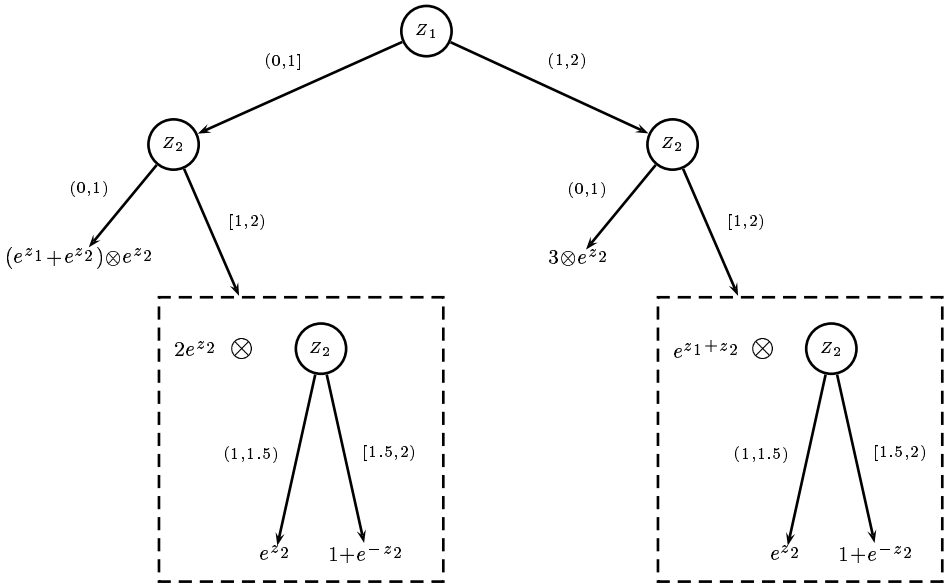


Figura 4.4: Paso 2: se combinan las restricciones de los árboles a $Z_2 \in (0,1)$ y $Z_2 \in [1,2)$.

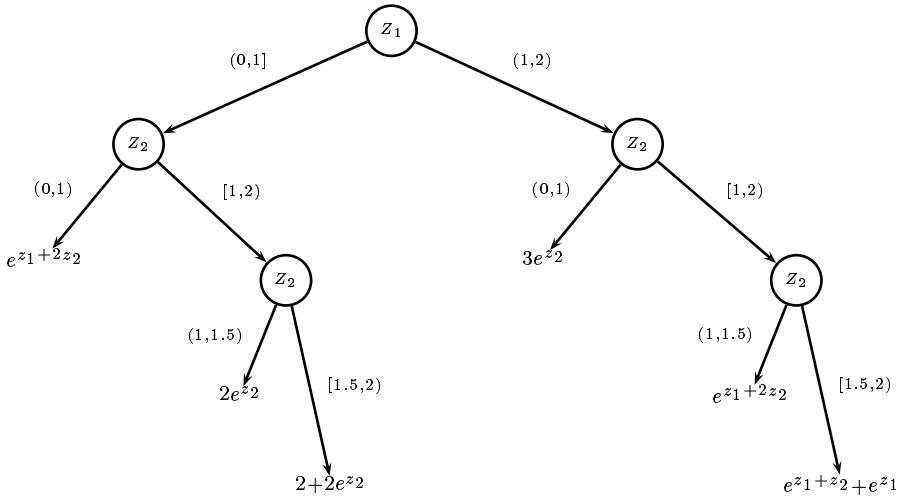


Figura 4.5: árbol resultado de la multiplicación de los dos árboles.

Dado un árbol $\mathcal{T}$ representando un potencial MTE definido sobre un conjunto $\mathbf{X}$ de variables, el siguiente algoritmo calcula un árbol que representa el potencial $\phi^{\downarrow(\mathbf{X}\setminus\{X_i\})}$. Esto es, elimina la variable X_i del árbol $\mathcal{T}$.

ELIMINA($\mathcal{T}$,X_i)

ENTRADA: un árbol de probabilidad mixto $\mathcal{T}$ y una variable X_i.
SALIDA: la marginal de $\mathcal{T}$ para las variables que contiene excepto X_i.

1. Si X_i es discreta,

   $\mathcal{T}_r :=$ **SUMAR($\mathcal{T}$,X_i)**.

2. Si no

   Sea (a, b) el intervalo donde X_i toma los valores.

   $\mathcal{T}_r :=$ **INTEGRA($\mathcal{T}$,X_i,a,b)**.

3. DEVUELVE $\mathcal{T}_r$.

Las operaciones **SUMAR** e **INTEGRA** las definimos a continuación:

SUMAR($\mathcal{T}$,X_i)

ENTRADA: un árbol de probabilidad mixto $\mathcal{T}$ y una variable X_i.
SALIDA: un árbol obtenido de $\mathcal{T}$ eliminando X_i sumando los subárboles correspondientes a sus hijos.

1. Sea L la etiqueta del nodo raíz de $\mathcal{T}$.
2. Si L es un potencial MTE, crear un nodo $\mathcal{T}_r$ con etiqueta $L \times |\Omega_{X_i}|$.
3. Si no, sea X la variable correspondiente a la etiqueta L.
4. Si X es discreta,
   a) Si $X = X_i$,
      - Sean $\mathcal{T}_1, \ldots, \mathcal{T}_s$ los hijos del nodo raíz de $\mathcal{T}$.
      - $\mathcal{T}_r := \mathcal{T}_1$.
      - Para $i := 2$ a s, $\mathcal{T}_r :=$ **ADICION(**$\mathcal{T}_r$,$\mathcal{T}_i$**)**.

   b) Si no
      - Crear un nodo $\mathcal{T}_r$ con etiqueta X.
      - Para cada $x \in \Omega_X$
        - Hacer que $\mathcal{T}_h :=$ **SUMAR(**$\mathcal{T}^{R(X=x)}$,X_i**)** sea el hijo siguiente de $\mathcal{T}_r$.
5. Si X es continua,
   a) Crear un nodo $\mathcal{T}_r$ con etiqueta X.
   b) Para cada intervalo (a, b) correspondiente a arcos salientes de X,
      - Hacer que $\mathcal{T}_h :=$ **SUMAR(**$\mathcal{T}^{R(X \in (a,b))}$,$X_i$**)** sea el siguiente hijo de $\mathcal{T}_r$.
6. DEVUELVE $\mathcal{T}_r$.

Este algoritmo utiliza la operación **ADICION(**$\mathcal{T}_1$,$\mathcal{T}_2$**)**, que calcula la suma de $\mathcal{T}_1$ y $\mathcal{T}_2$, y se puede definir como sigue:

ADICION($\mathcal{T}_1$,$\mathcal{T}_2$**)**

ENTRADA: dos árboles de probabilidad mixtos $\mathcal{T}_1$ y $\mathcal{T}_2$.
SALIDA: la suma de $\mathcal{T}_1$ y $\mathcal{T}_2$.

1. Crear un nodo $\mathcal{T}_r$, en principio sin etiqueta.
2. Sean L_1 y L_2 las etiquetas de los nodos raíz de $\mathcal{T}_1$ y $\mathcal{T}_2$ respectivamente.
3. Si L_1 y L_2 son potenciales MTE, hacer que $L_1 + L_2$ sea la etiqueta de $\mathcal{T}_r$.
4. Si L_1 es un potencial MTE pero L_2 es una variable,
   a) Hacer que L_2 sea la etiqueta de $\mathcal{T}_r$.
   b) Para cada hijo $\mathcal{T}$ del nodo raíz de $\mathcal{T}_2$, hacer que $\mathcal{T}_h :=$ **ADICION(**$\mathcal{T}_1$,$\mathcal{T}$**)** sea un hijo de $\mathcal{T}_r$.
5. Si L_1 es una variable, supongamos que X es esa variable.
   a) Hacer que X sea la etiqueta de $\mathcal{T}_r$.
   b) Si X es discreta,
      - Para cada $x \in \Omega_X$,
        - Hacer que $\mathcal{T}_h :=$ **ADICION(**$\mathcal{T}_1^{R(X=x)}$,$\mathcal{T}_2^{R(X=x)}$**)** sea un hijo de $\mathcal{T}_r$.

   c) Si X es continua,
      1) Para cada intervalo (a, b) correspondiente a arcos salientes de X,
         - Hacer que $\mathcal{T}_h :=$ **ADICION(**$\mathcal{T}_1^{R(X\in(a,b))}$,$\mathcal{T}_2^{R(X\in(a,b))}$**)** sea un hijo de $\mathcal{T}_r$.
6. DEVUELVE $\mathcal{T}_r$.

INTEGRA($\mathcal{T}$,X_i,a,b**)**

ENTRADA: un árbol de probabilidad mixto $\mathcal{T}$, una variable X_i y dos números reales a y b.
SALIDA: un árbol obtenido de $\mathcal{T}$ donde la variable X_i ha sido eliminada integrando sobre (a, b).

1. Sea L la etiqueta del nodo raíz de $\mathcal{T}$.
2. Si L es un potencial MTE ϕ, crear un nodo $\mathcal{T}_r$ con etiqueta $\int_a^b \phi(x)\mathrm{d}x$.
3. Si no, sea X la variable correspondiente a la etiqueta L.
4. Si X es discreta,
   a) Hacer que X sea la etiqueta de $\mathcal{T}_r$.
   b) Para cada hijo $\mathcal{T}_h$ del nodo raíz de $\mathcal{T}$,
      - Hacer que $\mathcal{T}_h' :=$ **INTEGRA($\mathcal{T}_h$,X_i,a,b)** sea un hijo de $\mathcal{T}_r$.
5. Si X es continua,
   a) Si $X = X_i$,
      1) Etiquetar $\mathcal{T}_r$ con un potencial 0.
      2) Para cada intervalo (α, β) correspondiente a arcos salientes de X,
         - Sea $\mathcal{T}_h$ el árbol correspondiente a ese arco.
         - $(\alpha', \beta') := (\alpha, \beta) \cap (a, b)$.
         - Si $(\alpha', \beta') \neq \emptyset$,
            - $\mathcal{T}_r :=$ **ADICION($\mathcal{T}_r$, INTEGRA($\mathcal{T}_h$,X_i,α',β'))**
   b) Si no
      1) Hacer que X sea la etiqueta of $\mathcal{T}_r$.
      2) Para cada hijo $\mathcal{T}_h$ del nodo raíz de $\mathcal{T}$,
         - hacer que $\mathcal{T}_h' :=$ **INTEGRA($\mathcal{T}_h$,X_i,a,b)** sea un hijo de $\mathcal{T}_r$.
6. DEVUELVE $\mathcal{T}_r$.

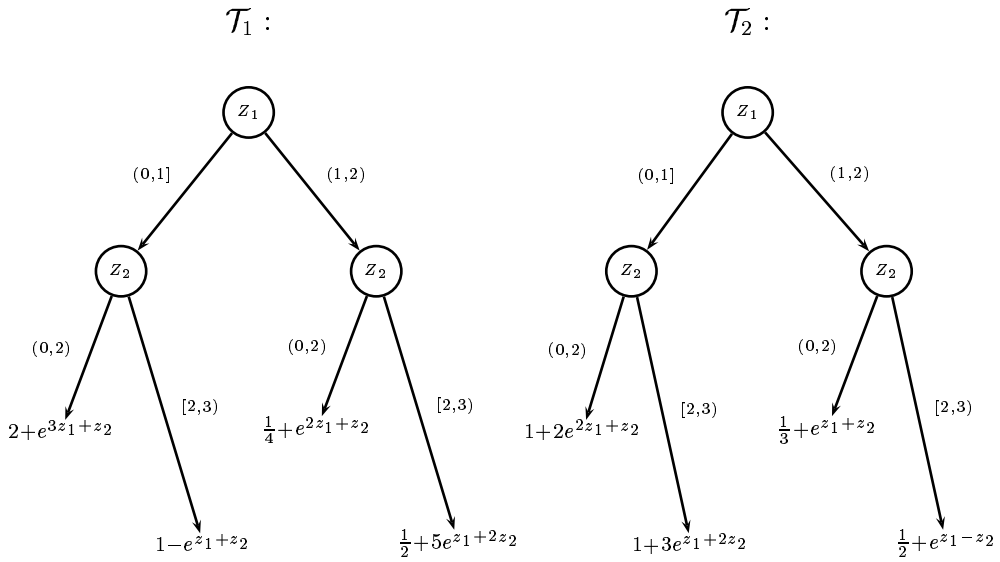


Figura 4.6: árboles $\mathcal{T}_1$ y $\mathcal{T}_2$.

Ejemplo 4.7 *Supongamos que queremos eliminar la variable Y_1 del árbol del ejemplo 4.5, entonces aplicamos el algoritmo* **ELIMINA($\mathcal{T}, Y_1$)**; *como la variable Y_1 es discreta, llamaría al algoritmo* **SUMAR($\mathcal{T}, Y_1$)**, *y éste lo que hace es llamar al algoritmo* **ADICION($\mathcal{T}_1, \mathcal{T}_2$)**, *siendo $\mathcal{T}_1$ y $\mathcal{T}_2$ los hijos del nodo raíz de $\mathcal{T}$ (ver figura 4.6). La adición de los árboles $\mathcal{T}_1$ y $\mathcal{T}_2$ se realiza como sigue: como el primer nodo de $\mathcal{T}_1$ es una variable, Z_1, el árbol de salida tendrá como nodo raíz esa variable, y los hijos de ese nodo serán el resultado de la adición de los dos árboles restringidos a cada uno de los intervalos correspondientes a los arcos salientes de Z_1 en el primer árbol $\mathcal{T}_1$, es decir, tendrá dos hijos, para $0 < Z_1 \leq 1$ el hijo será el resultado de* **ADICION($\mathcal{T}_1^{R(Z_1 \in (0,1])}$,$\mathcal{T}_2^{R(Z_1 \in (0,1])}$)**, *y para $1 < Z_1 < 2$ el hijo será el resultado de* **ADICION($\mathcal{T}_1^{R(Z_1 \in (1,2])}$,$\mathcal{T}_2^{R(Z_1 \in (1,2])}$)**.

$0 < Z_1 \leq 1$: *la restricción de los dos árboles $\mathcal{T}_1$ y $\mathcal{T}_2$ a éste intervalo de Z_1 se muestra en la figura 4.7. La adición de los árboles de la figura 4.7 da como resultado un*

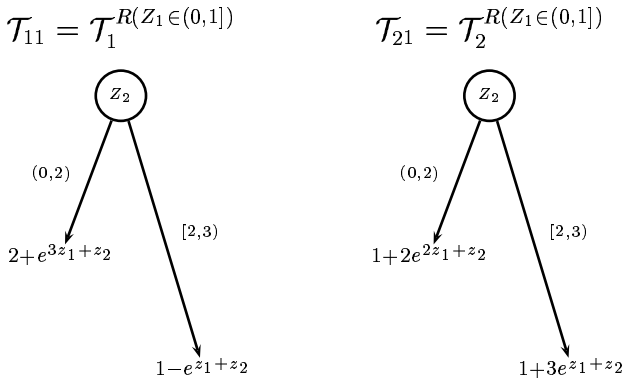


Figura 4.7: restricción de $\mathcal{T}_1$ y $\mathcal{T}_2$ a $Z_1 \in (0, 1]$.

árbol cuyo nodo raíz es Z_2 y que tiene como hijos el resultado de la adición de cada uno de estos dos árboles $\mathcal{T}_{11}$ y $\mathcal{T}_{21}$ restringidos a cada intervalo correspondiente a arcos salientes de Z_2 en $\mathcal{T}_{11}$:

$0 < Z_2 < 2$: *las restricciones de los árboles $\mathcal{T}_{11}$ y $\mathcal{T}_{21}$ son:*

$$\mathcal{T}_{11}^{R(Z_2 \in (0,2))} = 2 + e^{3z_1+z_2},$$

$$\mathcal{T}_{21}^{R(Z_2 \in (0,2))} = 1 + 2e^{2z_1+z_2},$$

luego la adición de estos dos árboles es la suma de esas dos cantidades, que es:

$$\textbf{ADICION}(\mathcal{T}_{11}^{R(Z_2 \in (0,2))}, \mathcal{T}_{21}^{R(Z_2 \in (0,2))}) = 3 + e^{3z_1+z_2} + 2e^{2z_1+z_2}.$$

$2 \leq Z_2 < 2$: *la restricción al otro intervalo es:*

$$\mathcal{T}_{11}^{R(Z_2 \in [2,3))} = 1 - e^{z_1+z_2},$$

$$\mathcal{T}_{21}^{R(Z_2 \in ([2,3))} = 1 + 3e^{z_1+z_2},$$

y de nuevo la adición de estos dos árboles es la suma de esas dos cantidades:

$$\textbf{ADICION}(\mathcal{T}_{11}^{R(Z_2 \in [2,3))}, \mathcal{T}_{21}^{R(Z_2 \in [2,3))}) = 2 + 2e^{z_1+z_2}.$$

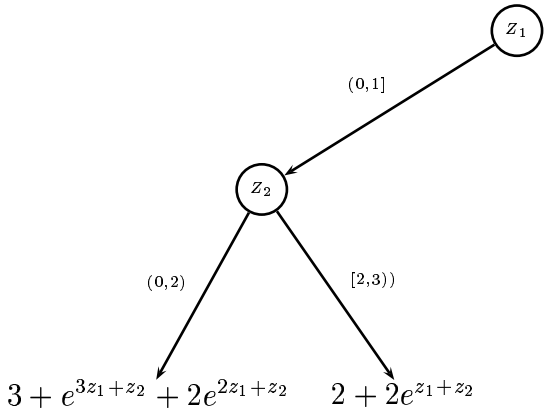


Figura 4.8: subárbol izquierdo resultado de eliminar Y_1 del árbol de la figura 4.1.

Ya que no podemos seguir con la parte izquierda del árbol (figura 4.8), se continua con la parte derecha. El otro intervalo de definición de Z_1 es:

$1 < Z_1 < 2$: *la restricción de los dos árboles $\mathcal{T}_1$ y $\mathcal{T}_2$ a éste intervalo de Z_1, ver figura 4.9, da como resultado dos árboles $\mathcal{T}_{11}$ y $\mathcal{T}_{22}$. La adición de éstos da como resultado un árbol cuyo nodo raíz es Z_2 y que tiene como hijos el resultado de la adición de cada uno de estos dos árboles $\mathcal{T}_{12}$ y $\mathcal{T}_{22}$ restringidos a cada intervalo correspondiente a arcos salientes de Z_1 en $\mathcal{T}_{12}$:*

$0 < Z_2 < 2$: *las restricciones de estos árboles son:*

$$\mathcal{T}_{12}^{R(Z_2\in(0,2))} = \frac{1}{4} + e^{2z_1+z_2},$$

$$\mathcal{T}_{22}^{R(Z_2\in(0,2))} = \frac{1}{3} + e^{z_1+z_2},$$

luego la adición de estos dos árboles es la suma de esas dos cantidades, que es:

$$\textbf{ADICION}(\mathcal{T}_{12}^{R(Z_2\in(0,2))}, \mathcal{T}_{22}^{R(Z_2\in(0,2))}) = \tfrac{7}{12} + e^{2z_1+z_2} + e^{z_1+z_2}.$$

$2 \leq Z_2 < 2$: *la restricción al otro intervalo es :*

$$\mathcal{T}_{12}^{R(Z_2\in[2,3))} = \frac{1}{2} + 5e^{z_1+2z_2},$$

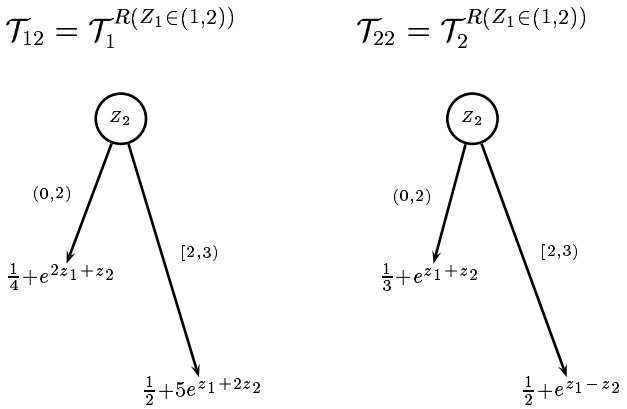


Figura 4.9: restricción de $\mathcal{T}_1$ y $\mathcal{T}_2$ a $Z_1 \in (1,2)$.

$$\mathcal{T}_{22}^{R(Z_2 \in [2,3))} = \frac{1}{2} + e^{z_1 - z_2},$$

y de nuevo la adición de estos dos árboles es la suma de esas dos cantidades, que es:

$$\textbf{ADICION}(\mathcal{T}_{12}^{R(Z_2 \in [2,3))}, \mathcal{T}_{22}^{R(Z_2 \in [2,3))}) = 1 + 5e^{z_1+2z_2} + e^{z_1-2z_2}.$$

Así pues, el resultado final de la eliminación de la variable Y_1 lo podemos ver en la figura 4.10.

Ejemplo 4.8 *Si estamos interesados ahora en eliminar la variable Z_1 del árbol $\mathcal{T}$ resultado del ejemplo 4.7 (figura 4.10), al aplicar el algoritmo* **ELIMINA**$(\mathcal{T}, z_1)$*, éste llamaría al algorimto* **INTEGRA**$(\mathcal{T}, \mathbf{Z_1}, 0, 2)$*, que, al tener $\mathcal{T}$ como nodo raíz la propia Z_1 calcula la integral de cada uno de sus hijos en el correspondiente intervalo y los suma (ver figura 4.11). Como el nodo raíz de cada subárbol es una variable distinta de Z_1, la integral se traduce en integrar las hojas (ver figura 4.12):*

1. ***Hoja 1**:*

$$\int_0^1 3 + e^{3z_1+z_2} + 2e^{2z_1+z_2} dz_1 = 3 - \frac{4}{3}e^{z_2} + \frac{1}{3}e^{3+z_2} + e^{2+z_2} =$$
$$= 3 + \left(\frac{-4 + e^3 + 3e^2}{3}\right) e^{z_2},$$

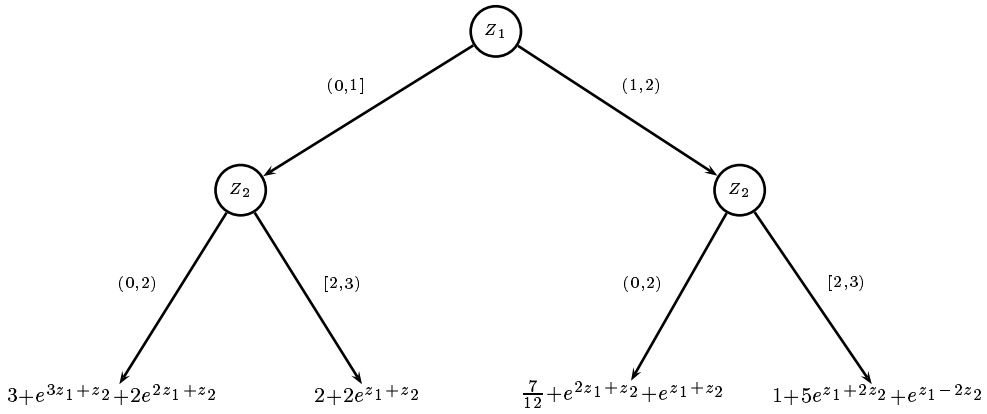


Figura 4.10: resultado de eliminar la variable Y_1 del árbol de la figura 4.1.

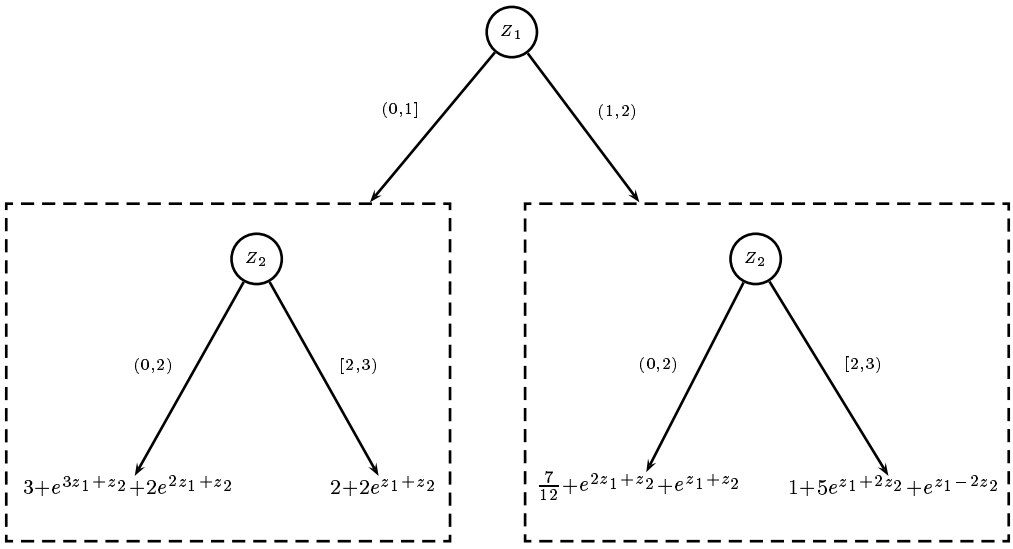


Figura 4.11: Paso 1: integrar los subárboles encuadrados y sumarlos.

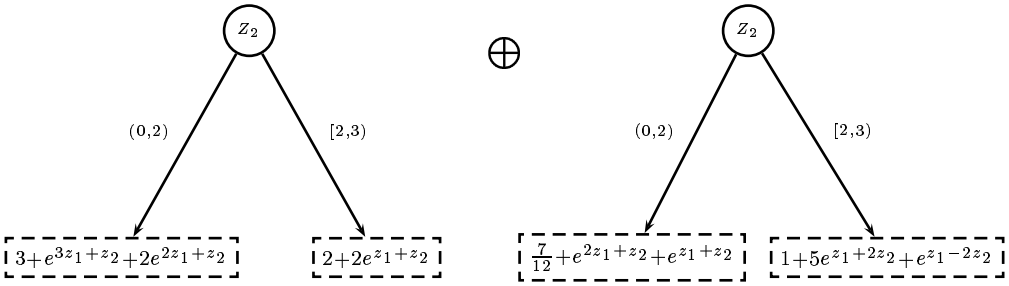


Figura 4.12: Paso 2: integrar las funciones encuadrados y realizar la suma de los árboles.

2. ***Hoja 2:***

$$\int_0^1 2 + 2e^{z_1+z_2} dz_1 = 2 + e^{1+z_2} - e^{z_2} = 2 + (e-1)e^{z_2},$$

3. ***Hoja 3:***

$$\int_1^2 \frac{7}{12} + e^{2z_1+z_2} + e^{z_1+z_2} dz_1 = \frac{7}{12} + \frac{1}{2}(e^{4+z_2} - e^{2+z_2}) + (e^{2+z_2} - e^{1+z_2}) =$$
$$= \frac{7}{12}\left(\frac{e^4 + e^2 - 2e}{2}\right) e^{z_2},$$

4. ***Hoja 4:***

$$\int_1^2 1 + 5e^{z_1+2z_2} + e^{z_1-2z_2} dz_1 = 1 + 5(e^{2+2z_2} - e^{1+2z_2}) + (e^{2-2z_2} - e^{1-2z_2}) =$$
$$= 1 + 5(e^2 - e)e^{2z_2} + (e^2 - e)e^{-2z_2}.$$

Tras realizar las integrales (figura 4.13) el resultado final aparece en la figura 4.14

4.4. Conclusiones

Hemos introducido la distribución MTE como un modelo para las redes bayesianas híbridas, definiendo para ello los potenciales, densidades y densidades condicionadas

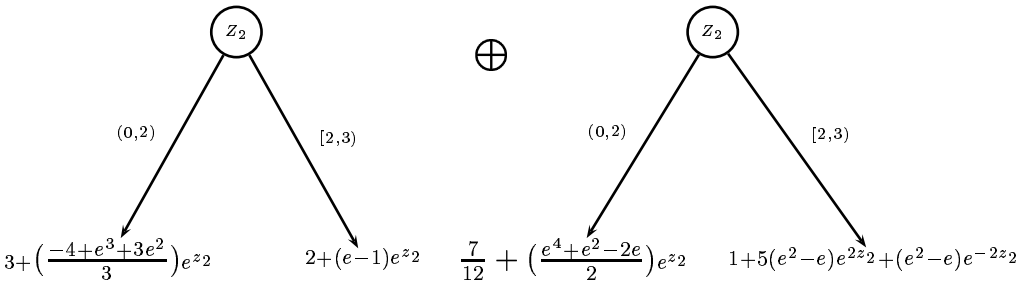


Figura 4.13: Paso 3: realizar la suma de los árboles.

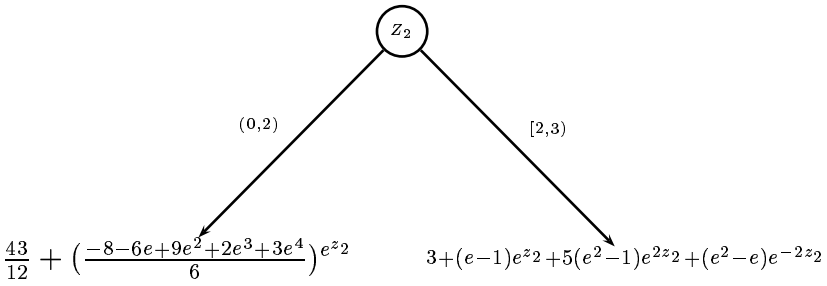


Figura 4.14: árbol resultante de eliminar Z_1 del árbol de la figura 4.10.

que aparecen en una red bayesiana, así como las operaciones usuales sobre ellos. También hemos comprobado que algunos modelos destacados son clases particulares de éste.

Además, hemos aportado una estructura adecuada para su representación, los árboles de probabilidad mixtos, extendiendo de forma natural los ya conocidos árboles de probabilidad.

Una vez que ya sabemos que podemos operar con este tipo de potenciales de forma similar a como lo hacíamos con tablas de probabilidad para variables exclusivamente discretas, nos queda ser capaces de obtener este tipo de potenciales a partir de una base de datos, con el fin de construir redes MTE, y ser capaces de adaptar algunos de los algoritmos de propagación conocidos a estas redes MTE.

Capítulo 5

Estimación de MTE a partir datos

5.1. Introducción

En este capítulo tratamos de resolver el paso siguiente a dar para poder trabajar con las MTE; su obtención a partir de una base de datos.

Abordar el problema general de estimar una densidad multivariante MTE tal y como la definimos en (4.1) resulta una tarea compleja en extremo, por lo que comenzamos estimando una densidad univariante MTE, para más adelante afrontar el problema más general de las densidades condicionadas.

Podemos encontrar algunos trabajos generales sobre distribuciones truncadas y estimadores de sus parámetros en la literatura (Tukey 1949; Smith 1957), y más proximo a nuestro caso encontramos casos concretos de funciones Gamma (Hegde y Dahiya 1989; Nath 1975), pero en la mayoría de los casos los parámetros desconocidos de la distribución se reducen a uno, y el estimador máximo verosímil, que no siempre existe, o el estimador UMVUE se suele obtener mediante métodos numéricos (Sathe y Varde 1969; El-Taha y Evans 1992). En nuestro caso particular, debido al gran número de parámetros que hemos de tener en cuenta, este tipo de estimadores resultaron imposibles de calcular.

Un método iterativo clásico para la obtención de los estimadores máximo verosímiles

de los parámetros de una distribución es el *algoritmo EM* (Redner y Walker 1984; Wu 1983), propuesto por Dempster, Laird, y Rubin (1977), que se estructura en dos fases bien diferenciadas, una primera de cálculo de la esperanza de la verosimilitud de los datos condicionada a los valores de los parámetros, y una segunda de maximización de dicha esperanza. Pero en este caso nos encontramos con el problema de calcular las esperanzas condicionadas.

Debido a la imposibilidad de usar los métodos de estimación antes citados, definimos un nuevo algoritmo de estimación que se basa en la regresión exponencial, dado que la función exponencial es la componente fundamental de nuestras distribuciones MTE. Se trata de un proceso iterativo, en el cual en cada paso se van calculando nuevas estimaciones de los parámetros desconocidos de la distribución.

Dividiremos este capítulo en dos partes principales, una primera dedicada a presentar el algoritmo de estimación, y otra con los experimentos realizados.

5.2. Algoritmo de estimación

Centraremos nuestra atención en estimar distribuciones MTE continuas univariantes, es decir, marginales, o aquellas que se corresponderían con los nodos raíz de una red bayesiana.

En el caso de querer estimar distribuciones discretas, ya vimos en la proposición 4.3 que este tipo de distribuciones se podían representar mediante un potencial MTE. Los correspondientes valores de probabilidad de cada estado de la variable se estiman mediante la fórmula (1.5).

Dado que la variable es continua, partimos de una función de densidad $f(x)$ para la cual queremos obtener la densidad MTE que mejor la aproxime. Obtener la densidad MTE que mejor represente a un conjunto de datos es un caso particular de éste, ya que la densidad empírica asociada a los datos se puede considerar como esta densidad $f(x)$.

Como vimos en el capítulo anterior, una densidad MTE puede estar definida en varios intervalos, y en cada uno de ellos puede tener un número distinto de términos exponenciales. Así pues lo primero que tendremos que resolver será cómo partir el dominio de definición, y cuántos términos exponenciales introducir en los distintos intervalos de definición, para después pasar a estimar el modelo concreto en cada uno de los intervalos.

5.2.1. Partición del dominio

La forma en la que partiremos el dominio viene determinada por las propiedades de la función exponencial, que es la parte principal del modelo MTE. Como podemos observar, la función $exp(x)$ es cóncava y creciente en todo su dominio de definición; por tanto, la partición que hagamos de éste debe ser tal que en cada uno de los subintervalos la densidad $f(x)$ que queremos aproximar no muestre cambios de concavidad/convexidad o crecimiento/decrecimiento. En cualquier caso se debe imponer un límite superior al número de subintervalos, para evitar un crecimiento excesivo de la complejidad del modelo.

5.2.2. Número de términos exponenciales

El esquema de aprendizaje que hemos diseñado permitiría incorporar nuevos términos exponenciales en cada subintervalo siempre que aumentase la precisión del modelo estimado. Sin embargo, hemos impuesto un límite superior de dos términos exponenciales (además de un término constante) para cada subintervalo.

La razón de por qué usar a lo sumo dos términos exponenciales es por la forma en la que partimos el dominio: en cada subintervalo no hay cambios de crecimiento ni de concavidad, por lo tanto sólo dos términos exponenciales nos bastan para ajustar de forma precisa casi cualquier curva. Evidentemente, la precisión de la estimación podría mejorar si usásemos más términos, pero el incremento de dificultad y tiempo de computación que conllevaría puede que no resulte rentable.

Por tanto, en cada intervalo estimaremos un término independiente y no más de dos términos exponenciales, esto es, el potencial MTE estimado en cada intervalo será:

$$f^*(x) = K + a\exp\{bx\} + c\exp\{dx\} \tag{5.1}$$

5.2.3. Ajuste del modelo en cada subintervalo

Veamos pues como estimar el potencial (5.1) en un intervalo D_j concreto. Tenemos como punto de partida dos vectores, $\mathbf{x}$ e $\mathbf{y}$. En $\mathbf{x}$ tenemos un conjunto de puntos dentro del intervalo D_j, y en $\mathbf{y}$ el valor de la densidad f (empírica) en cada punto de $\mathbf{x}$, es decir, $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ e $\mathbf{y} = (f(x_1), f(x_2), \ldots, f(x_n))$. Organizamos la información de esta manera para poder atacar el problema como si de un problema de *regresión exponencial* se tratase. En una regresión exponencial tenemos un vector de puntos $(\mathbf{x}, \mathbf{y})$, e intentamos ajustar a estos puntos una función de la forma

$$y = f^*(x) = a\exp\{bx\}'$$

que minimice el error cuadrático medio. Para conseguir esto tomamos logaritmos

$$\ln\{y\} = \ln\{a\}\exp\{bx\} = \ln\{a\} + bx \ .$$

Por tanto podemos escribir

$$y^* = a^* + bx \ ,$$

con $a^* = \ln\{a\}$ e $y^* = \ln\{y\}$, esto es, una regresión lineal cuya solución es

$$(y^* - \bar{y^*}) = \frac{S_{xy^*}}{S_x^2}(x - \bar{x}) \ .$$

El cálculo del término constante lo hacemos así: tenemos

$$y = f^*(x) = a\exp\{bx\} + c\exp\{dx\} + K$$

y queremos obtener un $K \in I\!R$ que minimice la función error

$$E = \sum_{i=1}^{n} \frac{(y_i - f^*(x_i))^2}{n} \ .$$

Si sustituimos $f^*(x)$ por su verdadero valor obtenemos

$$E = \sum_{i=1}^{n} \frac{(y_i - a\exp\{bx_i\} - c\exp\{dx_i\} - K)^2}{n} \ .$$

Para obtener lo valores mínimos es necesario derivar

$$\frac{\partial E}{\partial K} = \sum_{i=1}^{n} \frac{-2(y_i - a\exp\{bx_i\} - c\exp\{dx_i\} - K)}{n} \ ,$$

y tras resolver la ecuación $\frac{\partial E}{\partial K} = 0$ obtenemos:

$$K = \frac{1}{n}\sum_{i=1}^{n}(y_i - a\exp\{bx_i\} - c\exp\{dx_i\}) \ , \tag{5.2}$$

y para comprobar que se trata de un mínimo,

$$\frac{\partial^2 E}{\partial K^2}(K) = 2 > 0 \ .$$

Seguiremos un algoritmo iterativo para estimar los parámetros del potencial MTE. Primero estimaremos una exponencial y luego la otra, de tal forma que sólo introduciremos el segundo término exponencial si hace disminuir el error.

Necesitamos algunos valores iniciales de a, b y K, y como valores iniciales de c y d tomaremos 0, así pues empezamos el algoritmo con un término exponencial y el término independiente conocidos.

MTE-learn(a,b,K,$\mathbf{x}$,$\mathbf{y}$,M)

ENTRADA: los valores $(\mathbf{x}, \mathbf{y})$ de la densidad empírica, los valores iniciales de los parámetros y el número máximo de iteraciones M.
SALIDA: los valores a, b, c, d y K de los parámetros.

1. Hacer $c := d := 0, \quad i := 0$.

2. Calcular el error cuadrático medio E.

3. Mientras ($i < M$) y el error decrezca en alguno de los pasos,

   a) $Error\_decrece := FALSO$

   b) $\mathbf{w} = \mathbf{y} - a\exp\{b\mathbf{x}\} - K$.

   c) Obtenemos c^* y d^* mediante la regresión exponencial $\mathbf{w} = c^*\exp\{d^*\mathbf{x}\}$.

   d) Calculamos el error cuadrático medio E^* con los nuevo valores c^* y d^*. Si es menor que E

      1) $E := E^*$

      2) $c := c^*$

      3) $d := d^*$

   e) $\mathbf{w} = \mathbf{y} - c\exp\{d\mathbf{x}\} - K$.

   f) Obtenemos a^* y b^* mediante la regresión exponencial $\mathbf{w} = a^*\exp\{b^*\mathbf{x}\}$.

   g) Calculamos el error cuadrático medio E^* con los nuevo valores a^* y b^*. Si es menor que E

      1) $E := E^*$

      2) $c := c^*$

      3) $d := d^*$

   h) Calculamos el término independiente K^* como se muestra en la Ecuación (5.2).

   i) Calculamos el error cuadrático medio E^* con el nuevo valorK^*. Si es menor que E

      1) $E := E^*$

      2) $K := K^*$

4. DEVUELVE a, b, c, d, K

Aclaremos algunos aspectos de este algoritmo: cuando tenemos una exponencial y pasamos a estimar los parámetros de la otra, por ejemplo c y d, obtenemos la exponencial que mejor aproxima la nueva pareja $(\mathbf{x}, \mathbf{w})$, pero puede que estemos sobreestimando los datos, es decir, puede que lo mejor que podamos hacer sea no introducir este término, pero estamos forzando a que lo haga. Para evitar esta situación calculamos un coeficiente H, minimizando el error, que multiplica a esta exponencial, para obtener la influencia real de este término:

$$\mathbf{y} = a \exp\{b\mathbf{x}\} + Hc \exp\{d\mathbf{x}\} + K \ .$$

El valor de H que minimiza el error es

$$H = \frac{\sum_{i=1}^{n}(y_i - a \exp\{bx_i - K\})(\exp\{dx_i\})}{\sum_{i=1}^{n} c \exp\{2dx_i\}} \ ,$$

por tanto hacemos $c = c * H$, e igual hacemos con a.

Además, en los pasos 2 y 5 creamos un nuevo vector $\mathbf{w}$. El vector $\mathbf{y}$ es siempre positivo, al tratarse de los valores de una densidad, pero este nuevo vector puede ser negativo, por lo que necesitamos transformar los datos antes de resolver la regresión exponencial de los pasos 3 y 6. Lo que hacemos es sumar una constante a $\mathbf{w}$ de forma que hagamos cada $w_i > 0$. Tras obtener los parámetros de la regresión deberíamos deshacer el transformación, pero no necesitamos hacerlo, ya que en el paso 8 calculamos el término independiente que minimiza el error.

Los valores iniciales de c y d los fijamos a cero de forma que no los cambiamos a menos que haya una reducción en el error, esto es, no introducimos el segundo término exponencial a menos que mejore la estimación.

Los valores iniciales de a, b y K podrían ser cualesquiera, pero sugerimos dos métodos diferentes para calcularlos:

- Regresión exponencial: obtener a y b de $\mathbf{y} = a \exp\{b\mathbf{x}\}$ y K como se explica en Ecuación (5.2).

- Método de la derivada: queremos obtener $f(x) \approx a \exp\{bx\} + K$, así que la derivada de ambas funciones deberían ser iguales: $f'(x) = ab \exp\{bx\}$, pero nosotros no trabajamos con densidades, sino con parejas $(\mathbf{x}, \mathbf{y})$, así que $f'(x)$ significa la pendiente de las líneas que unen los puntos $(\mathbf{x}, \mathbf{y})$. El método sería:

  1. Obtener la ecuación de la recta entre los puntos $(x_{(i-1)*m}, y_{(i-1)*m})$ y (x_{i*m}, y_{i*m}) para $i = 1, \ldots, n/m$.
  2. $x_i^* = \frac{x_{(i-1)*m} + x_{i*m}}{2}$ para $i = 1, \ldots n/m$.
  3. Tomar cómo y_i^* la pendiente de la recta entre $(x_{(i-1)*m}, y_{(i-1)*m})$ y (x_{i*m}, y_{i*m}).
  4. Resolver la regresión exponencial $\mathbf{y}^* = a^* \exp\{b\mathbf{x}^*\}$ dónde $a^* = ab$, y obtener los valores iniciales para a y b.
  5. Obtener K como se explica en Ecuación (5.2).

Este es el método iterativo que proponemos para estimar una MTE a partir de los datos. De hecho, lo que obtenemos no es una densidad, sino un potencial que habrá que normalizar posteriormente.

5.3. Experimentos

Para probar la capacidad de la distribución MTE de modelizar situaciones comunes, y para comprobar la precisión del algoritmo, hemos llevado a cabo varios experimentos para estimar algunas distribuciones conocidas así como datos reales.

5.3.1. Distribuciones conocidas

Veremos cómo podemos representar tres distribuciones diferentes, muy comunes, mediante una densidad MTE.

5.3.1.1. Distribución uniforme

Como se ha comentado anteriormente, una de las ventajas de usar las MTE es que se tratan de una generalización de la discretización, en el sentido de que podemos ver una discretización como una mixtura de uniformes, así pues, si una MTE puede representar a una distribución uniforme, también puede representar una discretización.

La densidad uniforme es

$$f(x) = \begin{cases} \dfrac{1}{l_2 - l_1} & x \in (l_1, l_2) \ , \\ 0 & \text{en otro caso} \ . \end{cases}$$

Esta distribución se obtiene exactamente al aplicar el algoritmo. No tenemos que partir el dominio, al no haber cambios de concavidad ni de crecimiento, y en la primera iteración obtenemos los valores exactos para los parámetros, esto es $a = b = c = d = 0$, y $K = \frac{1}{l_2 - l_1}$.

5.3.1.2. Distribución exponencial

La densidad exponencial es:

$$f(x) = \lambda \exp\{-\lambda x\} \qquad x > 0 \ .$$

Lo primero que tenemos que hacer antes de aplicar el algoritmo MTE-learn a esta densidad es definir el dominio de nuestra MTE, ya que éste debe ser finito. Para ello seleccionamos un intervalo $(0, l)$ dónde l es tal que $P(X > l) \approx 0$. Así pues sólo tenemos un intervalo en el dominio, y el algoritmo obtiene el valor exacto de los parámetros.

5.3.1.3. Distribución Normal

Nos centraremos en la distribución normal estándar, $\mathcal{N}(0, 1)$, ya que podemos obtener cualquier $\mathcal{N}(\mu, \sigma)$ a partir de ésta. Para aplicar el algoritmo dividimos el dominio de

la densidad normal en cuatro intervalos, $(-4,-1),(-1,0),(0,1)$ y $(1,4)$, en los cuales la densidad no tiene cambios de concavidad ni de crecimiento. Aplicando el algoritmo a estos cuatro intervalos obtenemos:

Intervalo $(-4,-1)$:

$$\begin{aligned} a &= 2.67632782323753 \\ b &= 2.0995654834596547 \\ c &= -4.307200936472499\text{E}-4 \\ d &= 0.0022590818669798686 \\ K &= 0.0013430276452163264 \end{aligned}$$

Intervalo $(-1,0)$:

$$\begin{aligned} a &= -0.025332983750353668 \\ b &= -1.9593439331969362 \\ c &= 0 \\ d &= 0 \\ K &= 0.4201166819186963 \end{aligned}$$

Intervalo $(0,1)$:

$$\begin{aligned} a &= -0.013883101566178647 \\ b &= 2.618433856043035 \\ c &= 0 \\ d &= 0 \\ K &= 0.4087927677026328 \end{aligned}$$

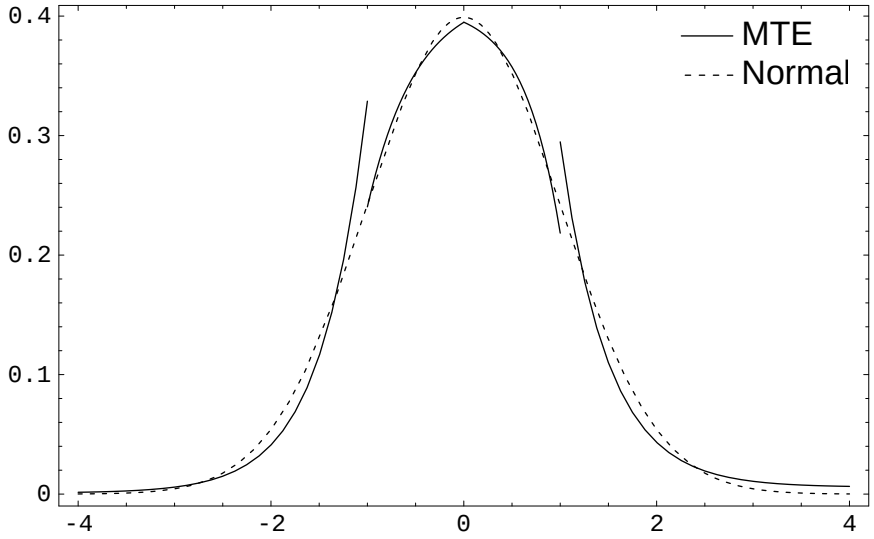


Figura 5.1: Comparación de la densidad real y la densidad MTE estimada para la distribución $N(0,1)$.

Intervalo $(1,4)$:

$$\begin{aligned} a &= 2.2257931326098874 \\ b &= -2.04177256076009 \\ c &= -0.0018154157845694822 \\ d &= 0.004952177870949646 \\ K &= 0.007669343990692532 \end{aligned}$$

Como podemos ver en la figura 5.1, las diferencias entre la $\mathcal{N}(0,1)$ y esta densidad MTE definida en cuatro partes son mínimas. Para probar que realmente se trata de una buena aproximación, simulamos 100 valores de la distribución normal estándar y otros 100 valores de la distribución MTE y llevamos a cabo un test de Kolmogorov-Smirnov para dos muestras siendo la hipótesis nula que ambas muestras provienen de la misma distribución. Obtuvimos un p-Valor de 0.6994, que apoya la idea de que ambas muestras proceden de la misma distribución.

5.3.2. Datos reales

Cuando intentamos aprender una red bayesiana no tenemos las densidades explícitas de las variables continuas, sino una muestra de valores de la variable. En esta sección veremos cómo el algoritmo **MTE-learn** puede tratar esta cuestión.

Sea $\mathbf{z} = (z_1, \ldots, z_n)$ una muestra de una variable continua Z. Según vimos en la Sección 3 el algoritmo requiere de dos vectores, $\mathbf{x} = (x_1, \ldots, x_m)$ e $\mathbf{y} = (y_1, \ldots, y_m)$, donde $\mathbf{x}$ son los valores de la variable y cada y_i representa el valor de la densidad de la variable en x_i. Por consiguiente, antes de aplicar el algoritmo debemos transformar los datos para obtener estos dos vectores. Usamos para ello el siguiente algoritmo:

Densidad-Empirica($\mathbf{z}$,m)

ENTRADA: los valores $\mathbf{z}$ de la variables continua y el número m de intervalos en los que dividir el dominio.
SALIDA: la densidad empírica correspondiente

1. Dividimos el dominio de la variable $\mathbf{Z}$ en m sub-intervalos, $I_1, \ldots, I_m$ disjuntos:

$$\Omega_Z = \bigcup_{i=1}^{m} I_i \ .$$

2. Calculamos las frecuencias n_j para cada intervalo I_j, $j = 1, \ldots, m$ en el vector $\mathbf{z}$.

3. Tomamos como x_i el punto medio del intervalo I_i, $i = 1, \ldots, m$.

4. $y_i = \dfrac{n_i/n}{longitud(I_i)}$, con n igual al tamaño de la muestra.

5. DEVOLVER $(\mathbf{x}, \mathbf{y})$.

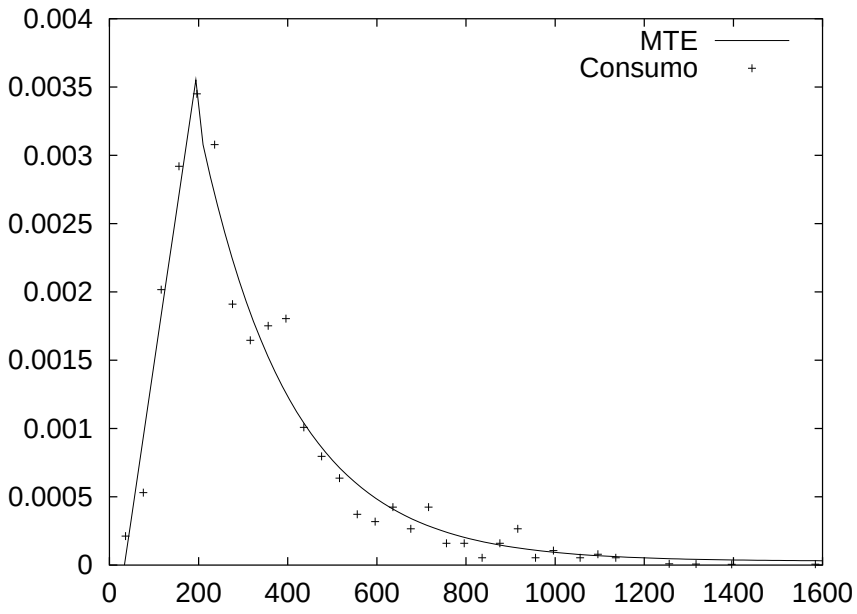


Figura 5.2: Aproximación de la densidad empírica de la variable Consumo por un modelo MTE

Hemos usado este algoritmo para estimar la densidad de dos variables continuas extraídas de una encuesta agrícola acerca de los invernaderos de la provincia de Almería: Consumo y Cosecha. Para probar la precisión de la estimación hemos usado un test χ^2 en vez de un test de Kolmogorov-Smirnov ya que algunos de los valores se repiten debido al redondeo de los datos al responder a las preguntas de la encuesta.

5.3.2.1. Consumo

Tenemos una muestra de 471 valores de esta variable, que van desde 36.1 a 1583.79. Para obtener la densidad empírica hemos aplicado el algoritmo Densidad-Empirica escogiendo intervalos de longitud 40. El modelo resultante se puede ver en la figura 5.2. Podemos ver dos partes bien diferenciadas, en lo que se refiere a cambios de crecimiento y concavidad, lo que da lugar a una partición del dominio en dos intervalos: $(36.1, 196.1)$ y $(196.1, 1583.79)$ y en cada uno de ellos estimamos un potencial MTE:

Intervalo $(36.1, 196.1)$:

$$\begin{aligned} a &= -1.00481152209664 \\ b &= -2.2123865768185145\text{E} - 5 \\ c &= -4.649576063595301\text{E} - 16 \\ d &= -3.414123099380927\text{E} - 10 \\ K &= 1.00406169448145 \end{aligned}$$

Intervalo $(196.1, 1583.79)$:

$$\begin{aligned} a &= 0.008503182179895076 \\ b &= -0.004880418989153087 \\ c &= 7.840998789880702\text{E} - 8 \\ d &= -7.61875283797691\text{E} - 8 \\ K &= 2.8252423901716575\text{E} - 5 \end{aligned}$$

Para probar que esta densidad MTE es una buena aproximación de la densidad real, hemos aplicado un test χ^2 para determinar si la muestra original se puede decir que provenga de dicha densidad MTE o no. Fijando el nivel de significación a $\alpha = 0.05$ la región de rechazo resultante es $(0, 0.484) \cup (11.20, \infty)$, y el valor del estadístico obtenido en el test fue 3.172402699364092, por lo que podemos decir que el test apoya la hipótesis nula de que la muestra original viene de la distribución MTE estimada.

Es interesante resaltar que en el primer intervalo, la densidad MTE es casi una línea recta. Esto nos muestra que podemos también aproximar funciones de este estilo mediante las MTE.

5.3.2.2. Cosecha

De esta variable, al igual que de la anterior, tenemos una muestra de 471 valores, que van de 0.45 a 26. De nuevo tenemos dos intervalos bien diferenciados en los que partir el dominio, como podemos ver en la figura 5.3:

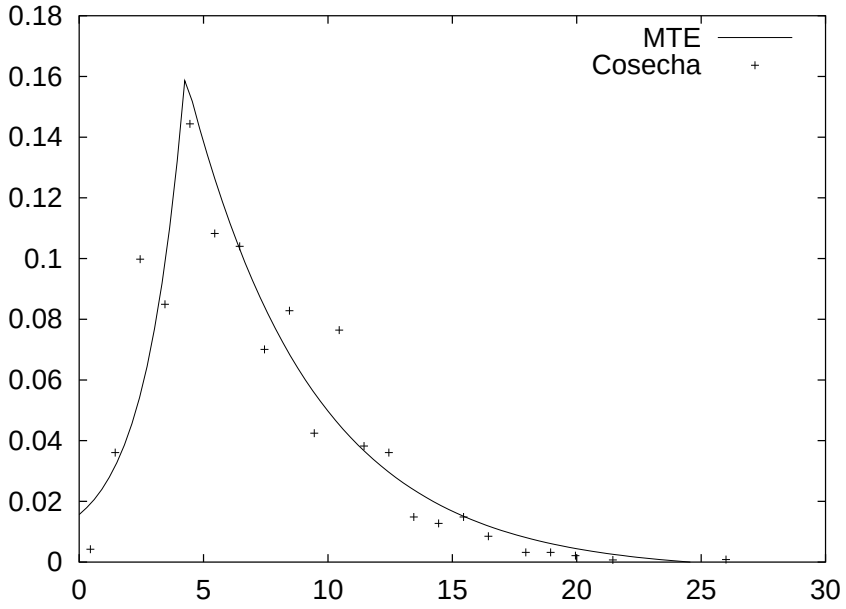


Figura 5.3: Aproximación de la densidad empírica de la variable Cosecha por una densidad MTE

Intervalo $(0.45, 4.45)$:

$$\begin{aligned} a &= 0.010826612517872319 \\ b &= 0.6255472893484579 \\ c &= -5.416676195266647\mathrm{E}-4 \\ d &= 0.015687129036406677 \\ K &= 0.0053790556847500455 \end{aligned}$$

Intervalo $(4.45, 26)$:

$$\begin{aligned} a &= 0.23314178683678177 \\ b &= -0.18400083794081074 \\ c &= 0.14954712040375723 \\ d &= -0.22459486950530835 \\ K &= -0.0031378040872907897 \end{aligned}$$

Realizando el test con los mismos parámetros que en el caso anterior, la región crítica es la misma, y obtenemos un valor para el estadístico χ^2 de 5.704724703434343, apoyando la hipótesis de que la muestra original proviene de la densidad MTE.

5.4. Conclusiones

Una vez presentadas y definidas las distribuciones MTE surge la necesidad de obtener algún procedimiento por el cual podamos calcular estimadores de los párametros de dichas distribuciones.

Debido a la dificultad del problema a abordar, hemos ido paso a paso; centrándonos en este capítulo en estimar distribuciones unidimensionales.

Primero partimos el dominio de definición de la variable en intervalos en los que estimar potenciales MTE. Dicha partición se realiza verificando las condiciones de que en cada intervalo no hay cambios ni de crecimiento ni de concavidad, siendo por tanto los intervalos resultantes óptimos para la estimación en cada uno de ellos de potenciales MTE.

El siguiente paso consiste en reducir el número de términos exponenciales, y por tanto de parámetros a estimar, de tal modo que la distribución MTE resultante siga aproximado de forma adecuada la distribución original. Dado que los intervalos verifican las propiedades anteriores, es suficiente considerar dos términos exponenciales y un término independiente en cada intervalo, debido a la flexibilidad de las funciones así definidas.

Tras estos pasos previos sólo queda estimar estos parámetros, para lo cual definimos un algoritmo iterativo que en cada paso va mejorando las estimaciones de los parámetros.

Una vez que ya tenemos un método de estimación de distribuciones univariantes, vamos un poco más alla y en el capítulo siguiente nos planteamos la estimación de *distribuciones condicionadas.*

Capítulo 6

Estimación de densidades MTE condicionadas

6.1. Introdución

Según vimos en (1.8), las distribuciones de probabilidad que aparecen asociadas a una red Bayesiana son distribuciones condicionadas a los padres para cada variable. Sólo en el caso de variables sin padres la distribución asociada es una marginal a priori.

Por tanto, a la hora de asignar probabilidades a las variables de una red Bayesiana, necesitamos conocer las distribuciones condicionadas a los padres de cada una de ellas, es decir, hemos de ser capaces de aprender una distribución MTE condicionada a partir de una base de datos.

El conjunto de padres de una variable puede estar formado tanto por variables continuas como por variables discretas, lo que ya de por sí supone una ventaja con respecto al modelo CG, que describimos en el Capítulo 3.

Sea X la variable cuya densidad vamos a estimar, y $\mathbf{\Pi}_X = \mathbf{W} = (\mathbf{Y}, \mathbf{Z})$ el conjunto formado por sus variables padre en la red, donde $\mathbf{Y}$ representa las variables discretas y $\mathbf{Z}$ las continuas, es decir, aquellas variables a las que condicionar. Nuestro objetivo es obtener una distribución de probabilidad condicionada $f(X|\mathbf{W})$.

Por definición, $f(X|\mathbf{W}) = \frac{f(X,\mathbf{W})}{f(\mathbf{W})}$, por lo que una posibilidad sería estimar $f(X|\mathbf{W})$ y $f(\mathbf{W})$ por separado, usando la técnica introducida en el capítulo anterior. Sin embargo, en general el cociente de dos potenciales MTE no es un potencial MTE.

Otra posibilidad es diseñar un algoritmo similar al del capítulo anterior para distribuciones condicionadas. Sin embargo, no hemos sido capaces de expresar analíticamente, hasta el momento, las restricciones sobre los parámetros.

En el capítulo 3 las distribuciones condicionadas de variables continuas a padres discretos vienen definidas como una distribución Gaussiana para cada configuración de los padres. Siguiendo esta filosofía, y dado que las distribuciones continuas MTE están definidas por intervalos, iremos obteniendo distintas configuraciones de los padres, tanto discretos como continuos, sobre las que definir distribuciones MTE univariantes.

Representaremos las distribuciones condicionadas resultantes mediante árboles de probabilidad mixtos, y el método de aprendizaje de éstas distribuciones lo enfocaremos como el proceso de construcción de dicho árbol (Moral, Rumí, y Salmerón 2003).

6.2. Construcción de árboles de probabilidad mixtos

Vamos a particionar el dominio de las variables condicionadas $\mathbf{W}$ para luego ajustar una densidad univariante $f(x)$ en cada una de las partes. Lógicamente, la precisión de la densidad estimada dependerá del número de partes en las que dividamos el dominio; cuanto mayor sea este número mayor será la capacidad de ajuste. Sin embargo, otro factor que determinará la bondad del ajuste es el tamaño de la muestra. Si el número de partes en las que dividimos el dominio es demasiado elevado, el subconjunto de la muestra que se usaría para estimar la densidad en cada región podría ser demasiado pequeño, o incluso cero. Este hecho debería ser tenido en cuenta a la hora de decidir cómo particionar el dominio.

El proceso de partir el dominio de las variables de $\mathbf{W}$ se puede ver como construir un

árbol de probabilidad mixto donde cada nodo interno es una variable $W_j \in \mathbf{W}$ y cada nodo hoja será una densidad MTE para la variable X dentro del intervalo determinado por la rama del árbol que lleva hasta ella. Para diseñar un algoritmo que construya el árbol, debemos tener en cuenta las siguientes cuestiones:

1. Seleccionar la variable por la que partir. Similar al caso de árboles de probabilidad, puede ayudar a construir árboles más pequeños sin pérdida de precisión.

2. Determinar el número de partes en las que dividiremos el dominio de la variable seleccionada.

3. Definir un criterio para parar de expandir el árbol. De nuevo, siguiendo las directrices de (Salmerón, Cano, y Moral 2000), proponemos varios criterios de parada, fijándonos bien en el número de hojas, bien en el tamaño del subconjunto de la muestra que corresponde a cada región.

A continuación estudiaremos estas cuestiones con más detalle.

6.2.1. Selección de la variable por la que expandir

El paso inicial en la construcción del árbol que estamos buscando es construir un árbol que tenga una única hoja, que será una densidad MTE ajustada a todos los casos de la muestra, es decir, la densidad condicionada $f(x|\mathbf{w})$ será la misma para cualquier valor de $\mathbf{w}$. Después, la variable elegida para expandir el árbol será aquella que divide el dominio de $\mathbf{W}$ en sub-regiones lo más diferentes posibles, entendiendo la diferencia entre sub-regiones como la bondad del ajuste de la densidad condicional actual $f(x|\mathbf{w})$ en cada una de ellas. La idea que surge tras este criterio es evitar expandir el árbol por aquellas variables con las cuales no ganaríamos en precisión.

Para determinar cómo de diferente es la bondad del ajuste entre las distintas sub-regiones, usamos la entropía del error cuadrático medio normalizado en cada sub-región, donde el error cuadrático medio lo calculamos entre la densidad ajustada $f(x|\mathbf{w})$y la densidad empírica de los datos correspondientes a esta sub-región.

Definición 6.1 *Sea X una variable aleatoria continua, y $f(x)$ una densidad MTE. Dada una muestra D de la variable X se define el* ***error cuadrático medio de*** *f* ***en*** *D, $\mathbf{e}(\mathbf{f}, \mathbf{D})$ como*

$$e(f, D) = \frac{1}{k} \sum_{i=1}^{k} (f(x_i) - y_i)^2 \tag{6.1}$$

donde $(\mathbf{x}, \mathbf{y}) = (x_1, \ldots, x_k, y_1, \ldots, y_k)$ es la densidad empiírica que se obtiene a partir de la muestra D.

Definición 6.2 *Sea Y una variable aleatoria discreta con k estados, y $f(y)$ una distribución MTE. Dada una muestra D de la variable X se define el* ***error cuadrático medio de*** *f* ***en*** *D, $\mathbf{e}(\mathbf{f}, \mathbf{D})$ como*

$$e(f, D) = \frac{1}{k} \sum_{i=1}^{k} (p_i - q_i)^2 \tag{6.2}$$

donde p_i son los valores de probabilidad de f y q_i son los valores de probabilidad que se obtienen si aprendiesemos una distribución para Y a partir de los valores de D.

Definición 6.3 *Sea $f(x)$ la densidad MTE de la variable continua objetivo X, nodo hoja de un árbol mixto que queremos expandir. Sea $W \in \mathbf{W}$ una variable que no ha sido expandida anteriormente en la rama del árbol, y $D_1, D_2, \ldots, D_n$ las sub-regiones en las que particionaríamos el dominio de $\mathbf{W}$ si expandiésemos por la variable W. Sean $e_i = e(f, D_i),\ i = 1, \ldots, n$ los errores cuadráticos medios normalizados de $f(x)$ en $D_i,\ i = 1, \ldots, n$ respectivamente. Definimos la* ganancia al partir *de la variable W con la función $f(x)$ como*

$$SG(f, W) = \sum_{i=1}^{n} e_i \log e_i \ . \tag{6.3}$$

Lema 6.1 *Dada una variable W y una función $f(x)$, la ganancia será mínima cuando todos los e_i sean iguales.*

Demostración: Sea A una vaiable aleatoria que toma los valores $a_1, \ldots, a_n$ con probabilidad $p(a_1) = e_1, \ldots, p(a_n) = e_n$ respectivamente. La entropía de esta variable A es (Kullback y Leibler 1951)

$$H(A) = -\sum_{i=1}^{n} p(a_i) log(p(a_i)) = -\sum_{i=1}^{n} e_i log(e_i) = -SG(f, W) \ ,$$

y dicha entropía está acotada superiormente por la cantidad $log(|A|) = log(n)$ (Kullback y Leibler 1951), por tanto

$$SG(f, W) = -H(A) \geq -log(n) = log(\frac{1}{n}) \ .$$

Sean ahora los $e_i = \frac{1}{n} \ \ \forall i = 1, \ldots, n$, entonces

$$SG(f, W) = \sum_{i=1}^{n} e_i log(e_i) = \frac{1}{n} \sum_{i=1}^{n} log(\frac{1}{n}) = \frac{1}{n} n log(\frac{1}{n}) = log(\frac{1}{n}) \ ,$$

es decir, $SG(f, W)$ alcanza su valor mínimo cuando todos los e_i son uniformes.

■

Nuestro propósito es expandir el árbol por aquella variable que maximize la ganancia al partir, ya que la partición que induce dicha variable discrimina los datos en regiones claramente diferenciadas en cuanto a su error, indicando cuales son las regiones por las que debemos seguir dividiendo la muestra.

6.2.1.1. Partición del dominio de la variable.

Si la variable es discreta, la partición más natural es obtener una sub-región para cada posible valor de la variable. Si la variable es continua, necesitamos dividir el dominio de la variable en intervalos. Lo ideal sería seleccionar los puntos de corte del dominio de tal forma que cada subconjunto de datos se ajustara de la mejor forma posible a una densidad MTE en cada una de las partes. Intentar obtener una partición como esta no es sencillo, y podría resultar demasiado costoso, ya que una estrategia de partición óptima debería tener en cuenta aspectos como el tamaño de muestra restante

en cada parte y la precisión de los modelos ajustados con respecto a todas las particiones posibles, para poder escoger la mejor de ellas. Esto lo podemos ver como un problema de optimización en el cual cada movimiento a través del espacio de búsqueda requiere la estimación de nuevas densidades MTE y la evaluación de su precisión. Éste es el motivo por el cual hemos decidido partir el dominio en intervalos de igual anchura, o en intervalos con igual frecuencia, siendo el número de intervalos un parámetro dado por el usuario.

6.2.1.2. Criterios de parada

El árbol ideal resultado del proceso de aprendizaje sería aquel en el que en cada rama que llevase a un nodo hoja apareciesen todas las variables condicionantes; pero esto en muchos casos puede no ser viable. Por ejemplo si tenemos muchas variables condicionantes, el árbol resultante puede ser demasiado grande, en términos de número de hojas, o bien la cantidad de tiempo necesaria para poder aprenderlo ser excesiva, por lo que su cálculo podría llegar a ser inviable, o consumir demasiados recursos.

Por tanto, dada una rama del árbol, **no** seguiremos partiendo esta rama si se da alguna de las siguientes condiciones:

- No quedan variables por las que expandir.
- El número de hojas que hay en el árbol excede un tamaño límite M.
- El subconjunto de la muestra que corresponde a esta rama del árbol no tiene el suficiente tamaño para poder estimar con garantías una densidad MTE sobre él. Como la forma de aprender una densidad MTE es a partir de la densidad empírica que se obtiene de la muestra, esta restricción se traduce fundamentalmente en poder ser capaces de obtener dicha densidad empírica.

6.2.2. Algoritmo de aprendizaje

Para resumir lo que hemos presentado hasta ahora, describimos el algoritmo de construcción de un árbol mixto con a lo sumo M hojas para la densidad condicional de una variable X dados sus padres $\mathbf{W}$, a partir de una base de datos D, y partiendo el dominio de cada variable continua en j intervalos.

APRENDE_ARBOL_MIXTO(D,X,$\mathbf{W}$,M,j)

ENTRADA: Una base de datos D con los elementos de la muestra.
SALIDA: Un árbol mixto representando la distribución condicionada $f(x|\mathbf{w})$

1. Sea f = **MTE-learn** una densidad para X a partir de D.
2. Si no se da ninguna de las condiciones de parada,
   - *a*) Escoger la variable $W \in \mathbf{W}$ tal que
     $$W = \text{argmax}_{W \in \mathbf{Y}} SG(W) \ .$$
   - *b*) Crear un nodo con etiqueta W.
   - *c*) Si W es continua
     1) Dividir el dominio de W en j intervalos.
     2) Crear j nodos vacíos, hijos de W..
     3) Dividir D en j partes $D_1, \ldots, D_j$ según la partición de W
     4) Para $i = 1$ a j
        APRENDE_ARBOL_MIXTO(D_i,X,$\mathbf{W} \setminus \{W\}$,$M$,$j$).
   - *d*) Si W es discreta
     1) Sea k el número de posibles valores de la variable.
     2) Para cada valor, crear un nodo vacío, hijo de W.

3) Dividir D en k partes $D_1, \ldots, D_k$ según la partición de W

4) Para $i = 1$ a k

APRENDE_ARBOL_MIXTO(D_i,X,$\mathbf{W} \setminus \{W\}$,$M$,$j$**)**.

3. DEVUELVE $\mathcal{T}$.

6.3. Experimentos con la distribución Normal

Para ilustrar cómo funciona el algoritmo describiremos aquí los resultados obtenidos al ajustar un árbol de probabilidad mixto a una distribución normal condicionada. Las características del experimento fueron las siguientes:

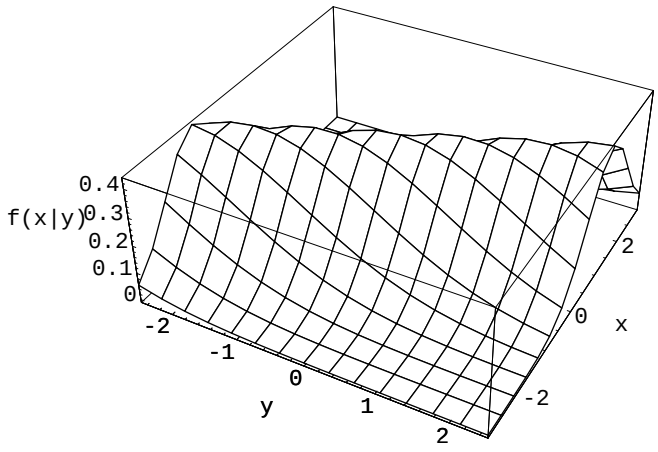


Figura 6.1: Función de densidad de una distribución $\mathcal{N}(0.5y, \sqrt{0.75})$.

Consideramos un par de variables aleatorias X e Y, que siguen una distribución

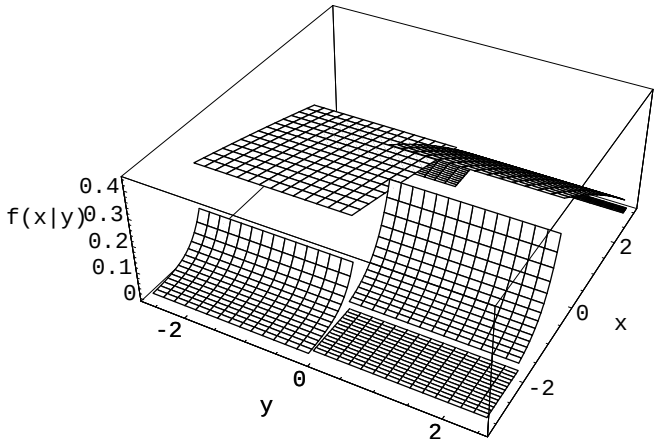


Figura 6.2: Densidad ajustada usando dos intervalos.

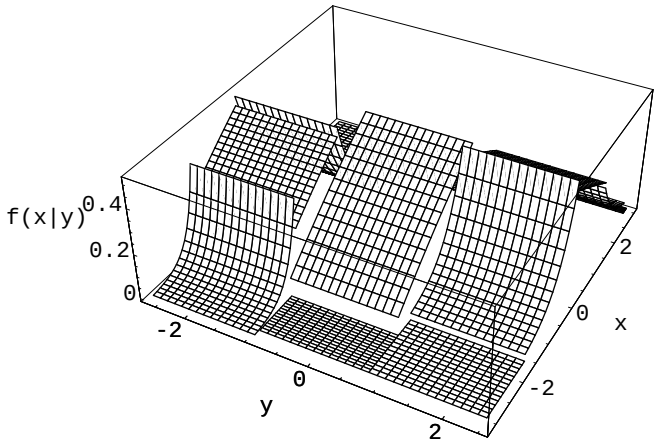


Figura 6.3: Densidad ajustada usando tres intervalos.

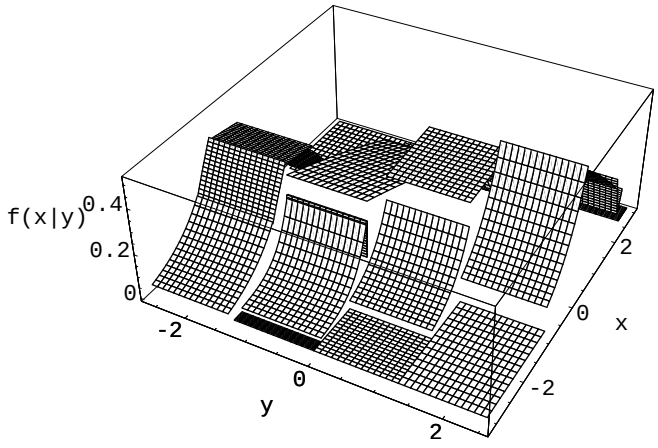


Figura 6.4: Densidad ajustada usando cuatro intervalos.

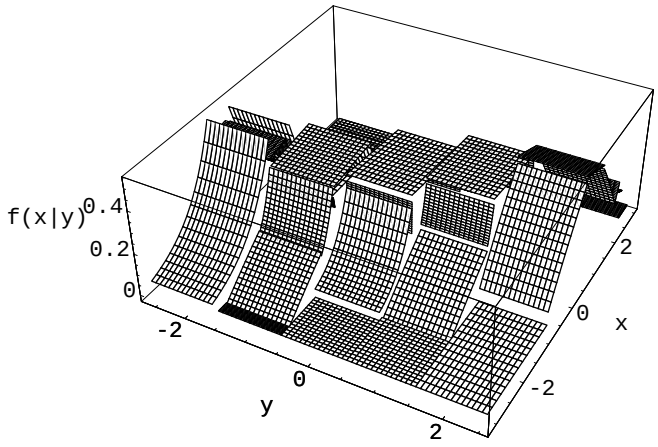


Figura 6.5: Densidad ajustada usando cinco intervalos.

normal bivariante con vector de medias $\overrightarrow{\mu} = (0, 0)$ y matriz de covarianzas

$$\mathrm{Cov}(X, Y) = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix} ,$$

lo que significa que $\sigma_{XY} = 0.5$, la distribución marginal de Y es $\mathcal{N}(0, 1)$ y la distribución condicional de X dado Y es $\mathcal{N}(0.5y, \sqrt{0.75})$. La densidad condicional correspondiente a esta distribución se muestra en la Figura 6.1.

Generamos una muestra de 500 valores de X a partir de la distribución condicionada. Para simular un valor de X, primero obtenemos un valor para Y a partir de la distribución $\mathcal{N}(0, 1)$, y entonces obtenemos un valor de X a partir de distribución $\mathcal{N}(0.5y, \sqrt{0.75})$, sustituyendo y por el valor simulado. Hemos aplicado el algoritmo con 2, 3, 4 y 5 intervalos en los que dividir el dominio de la variable por la que expandir, usando el método de igual longitud. Las densidades condicionadas ajustadas se muestran en las figuras 6.2, 6.3, 6.4 y 6.5.

Se puede observar cómo la precisión de las densidades estimadas aumenta conforme crece el número de intervalos.

6.4. Conclusiones

Este capítulo, en conjunción con el anterior, nos ofrece un método válido para poder aprender distribuciones condicionadas, $f(X|\mathbf{W})$ a partir de una muestra.

Somos capaces ya, por tanto, de aprender totalmente la parte cuantitativa de una red MTE, y por tanto estamos en condiciones de plantearnos cómo realizar propagación de probabilidades sobre este tipo de redes.

Capítulo 7

Propagación de probabilidades en redes MTE

En los capítulos anteriores fuimos capaces de obtener las densidades condicionadas que necesitamos para poder definir correctamente una red Bayesiana. Sólo nos queda pues aplicar los algoritmos de propagación que ya conocemos.

En la sección 7.1 estudiaremos la corrección del algoritmo de Shenoy y Shafer (1990) sobre redes MTE. En la sección 7.2 propondremos dos algorimtos aproximados: uno de ellos basado en la propagación Penniless (Cano, Moral, y Salmerón 2000) y otro en MCMC (Pearl 1987).

7.1. Propagación exacta de probabilidades en una red MTE

Consideremos una red MTE definida sobre una variable n-dimensional $\mathbf{X}$. Sea f la distribución conjunta de $\mathbf{X}$. Si denotamos por $f_i(x_i|\boldsymbol{\pi}_i)$ a la densidad condicional MTE de la variable $X_i,\ i = 1, \ldots, n$, dados sus padres $\boldsymbol{\Pi}_i$, entonces como sabemos por (1.1) , la distribución conjunta la podemos expresar como el producto de todas las

condicionadas,

$$f(\mathbf{x}) = \prod_{i=1}^{n} f_i(x_i|\boldsymbol{\pi}_i).$$

Si llamamos $H = \{f_i(x_i|\boldsymbol{\pi}_i)|i = 1, \ldots, n\} \cup \{\delta_i(x_i; e_i)|e_i \in \mathbf{e}\}$, entonces para cualquier variable $X_i \in \mathbf{X}$

$$f^{\downarrow X_i}(x_i, \mathbf{e}) = (\prod_{\phi \in H} \phi(\mathbf{x}, x_i, \mathbf{e}))^{\downarrow X_i}$$

El objetivo de la propagación, la obtención de $f^{\downarrow X_i}(x_i|\mathbf{e})$ es equivalente a obtener $f^{\downarrow X_i}(x_i, \mathbf{e})$. Veamos como calcular estos valores con el algoritmo de propagación de Shenoy-Shafer para la propagación exacta.

Obsérvese que la propagación sólo requiere combinaciones y marginalizaciones de potenciales MTE (ver fórmula 1.2). Así, cualquier nuevo potencial intermedio que se genere a lo largo de la propagación es también de clase MTE, como vemos en la siguiente proposición (Moral, Rumí, y Salmerón 2001).

Proposición 7.1 *La clase de potenciales MTE es cerrada bajo la propagación Shenoy-Shafer.*

Demostración: Inicialmente, el potencial guardado en cada nodo del árbol de cliques es el producto de algunas densidades MTE. Esto se mantiene incluso después de la propagación, ya que sólo se realizan combinaciones y marginalizaciones y estas operaciones son cerradas para potenciales MTE. Así las densidades marginales calculadas en la propagación Shenoy-Shafer, $f^{\downarrow X_i}$, son también de clase MTE y por tanto también lo es la densidad condicional

$$f(x_i|\mathbf{e}) = \frac{f^{\downarrow X_i}(x_i, \mathbf{e})}{f^{\downarrow \mathbf{E}}(\mathbf{e})},$$

al ser la división de un potencial MTE por un número real, que claramente es un potencial MTE.

■

En realidad, la clase de potenciales MTE es cerrada para todos aquellos algoritmos de propagación que no requieran de la división entre sus operaciones, ya que, si dividimos dos potenciales MTE, claramente el resultado no tiene por qué ser un potencial MTE. Por tanto, algunos algoritmos, como por ejemplo HUGIN, no los podemos utilizar a la hora de propagar sobre las redes MTE (Cobb y Shenoy 2003).

El algoritmo de propagación de Shenoy-Shafer trabaja sobre el árbol de cliques, que ya definimos en el capítulo 1, donde también vimos un algoritmo para su obtención, algoritmo que dependía de un orden de eliminación de las variables, ya que distintas secuencias de eliminación darán lugar a distintos árboles de cliques.

Seleccionar el orden de eliminación de las variables adecuado no es una tarea sencilla; se han propuesto varias heurísticas (Cano y Moral 1995; Kjærulff 1992), orientadas a variables discretas.

Nosotros seleccionaremos las variables a eliminar según sea el tamaño del clique resultante que surgiría al eliminar dicha variable, es decir, el tamaño del potencial asociado a dicho clique (Cano, Moral, y Salmerón 2000).

Definición 7.1 *Definimos el tamaño de un potencial MTE representado según un árbol de probabilidad mixto como el número de nodos hojas por la cantidad de términos exponenciales que hay en cada hoja, incluyendo el término constante.*

Tal y como aprendemos las densidades condicionadas según vimos en el capítulo anterior, los árboles resultantes son tales que el dominio de todas las variables continuas se divide en el mismo número de intervalos, y todas las densidades aprendidas tienen a lo sumo tres términos exponenciales. Por tanto es posible dar una cota superior del tamaño del árbol resultante del producto de dos árboles bajo esas dos condiciones:

Proposición 7.2 *Sean $\mathcal{T}_1, \ldots, \mathcal{T}_h$ h árboles de probabilidad mixtos aprendidos según los algoritmos del capítulo anterior, con $\mathbf{Y}_i$, $\mathbf{Z}_i$ las variables discretas y continuas de cada uno de los árboles $\mathcal{T}_i$, y n_i la cantidad de intervalos en los que se divide el dominio de las variables continuas en cad árbol $\mathcal{T}_i$. Sea Ω_{Y_i} el conjunto de posibles valores que*

puede tomar la variable discreta Y_i *.El tamaño del árbol* $\mathcal{T} = \mathcal{T}_1 \times \mathcal{T}_2 \times \ldots \times \mathcal{T}_h$ *no supera la cantidad*

$$\left(\prod_{Y_i \in \bigcup_{i=1}^{h} \mathbf{Y}_i} |\Omega_{Y_i}| \right) \times \left(|\bigcup_{i=1}^{h} \mathbf{Z}_i|^{\sum n_i} \right) \times t^k$$

donde k *es el número de árboles que están definidos para una variable continua, y* t *el número de términos exponenciales permitidos.*

Demostración:

Cada una de las variables discretas, $Y_i \in \bigcup_{i=1}^{h} \mathbf{Y}_i$ tendrá tantos hijos como estados tenga. Las variables continuas tienen tantos hijos como indica el correspondiente n_i, pero a la hora de multiplicar los árboles, hemos de considerar las intersecciones de los intervalos en los que está definida la variable en los dos árboles, esto es, la misma variable continua, $Z_i \in \bigcup_{i=1}^{h} \mathbf{Z}_i$ puede tener diferentes intervalos de partición de su dominio en cada uno de los dos árboles, $\mathcal{T}_i$ y $\mathcal{T}_j$, por lo que hemos de considerar tantos intervalos distintos como den lugar las intersecciones de los de cada árbol, por tanto, si en cada árbol tiene n_i intervalos distintos, en el árbol resultado de la multiplicación , tendrá como cota superior $\sum n_i$ intervalos de partición de su dominio.

Por tanto, combinando las variables discretas y las continuas, resulta que habrá un máximo de $\prod_{Y_i \in \bigcup_{i=1}^{h} \mathbf{Y}_i} |\Omega_{Y_i}| \times |\bigcup_{i=1}^{h} \mathbf{Z}_i|^{\sum n_i}$ nodos hoja, y cada uno de ellos tendrá el resultado del producto de las densidades definidas sobre ellas, que, al provenir del aprendizaje expuesto en los temas anteriores, serán de la forma $k + \sum_{i=1}^{t-1} a_i exp(b_i z_j)$ para cualquier variable continua Z_j y un número real p_i, para las discretas, y por tanto al multiplicar, k de ellas continuas, el resultado tiene t^k términos.

■

7.2. Propagación aproximada

En la sección 7.1 vimos que podíamos llevar a cabo la propagación de probabilidades en redes MTE de forma exacta mediante el algoritmo de Shenoy-Shafer. Sin embargo, durante la propagación, el tamaño de los potenciales involucrados en los cálculos puede crecer de tal modo que la propagación se vuelva inviable. Es entonces cuando entran en juego los algoritmos aproximados.

Como ya vimos en el capítulo 1, existen fundamentalmente dos tipos de algoritmos aproximados, los que se basan en la simulación y los que simplifican la red de alguna manera. Veremos a continuación un algorimto de cada clase:

7.2.1. Penniless Simple para redes MTE

El algoritmo de propagación *Penniless* (Cano, Moral, y Salmerón 2000) se basa en el algorimto *Shenoy-Shafer*, pero modificando su esquema de tal forma que lo que obtenemos son resultados aproximados.

La característica principal del algoritmo es que los mensajes que se mandan durante la propagación sufren un proceso de aproximación, en el que se reduce su tamaño. El proceso de aproximación de los potenciales, que en este algoritmo se representan mediante árboles de probabilidad se lleva a cabo mediante es **poda** de árboles (Cano, Moral, y Salmerón 2000; Cano, Moral, y Salmerón 2003; Salmerón, Cano, y Moral 2000).

Otra diferencia fundamental con el algoritmo *Shenoy-Shafer* es que en éste se realizan dos pasadas, con las cuales se recopila toda la información que subyace en la red, pero en el algoritmo *Penniless* se pueden realizar más pasadas, ya que en cada una de ellas los mensajes se van mejorando incrementalmente en función de los mensajes que van en la dirección contraria, haciendo uso por tanto de la información que va fluyendo por el árbol.

Se trata pues de un método que simplifica la red, más concretamente los potenciales

de la red, que en nuestro caso vienen definidos por árboles de probabilidad mixtos. Para poder aplicar esta filosofía *Penniless* a las redes MTE necesitamos definir la nueva operación *poda* sobre los potenciales:

7.2.1.1. Poda de un árbol de probabilidad mixto

Definíamos en la sección 7.1 el tamaño de un árbol en función del número de hojas que tiene y del numero de términos exponenciales que hay en cada hoja. Parece lógico entonces que la forma de reducir el tamaño de un árbol sea actuando sobre estas dos cantidades. Por tanto llevaremos a cabo la poda sobre un nodo cuyos hijos sean funciones MTE.

Toda poda conlleva un determinado error, por lo tanto la metodología general será podar, siempre y cuando el error que cometamos no supere un cierto parámetro ϵ que fijemos. En nuestro caso mediremos el error en términos de distancia entre árboles.

Definición 7.2 *Sea $\mathcal{T}$ un árbol de probabilidad mixto que representa un potencial MTE ϕ definido sobre la variable $\mathbf{X} = (\mathbf{Y}, \mathbf{Z})$. Sea $\mathcal{T}^*$ un subárbol de $\mathcal{T}$ con raíz $Z \in \mathbf{Z}$ con todos los hijos de Z funciones MTE. Sea ϕ_1 el potencial representado por $\mathcal{T}^*$. Sea $\mathcal{T}_P^*$ un árbol obtenido a partir de $\mathcal{T}^*$ cambiando ϕ_1 por otro potencial ϕ_2 con la única restricción de que $\int_{\Omega_\mathbf{Z}} \phi_1 d\mathbf{z} = \int_{\Omega_\mathbf{Z}} \phi_2 d\mathbf{z}$. Se define la distancia entre $\mathcal{T}^*$ y $\mathcal{T}_P^*$ como*

$$D(\mathcal{T}^*, \mathcal{T}_P^*) = E_{\phi_1^*}[(\phi_1^* - \phi_2^*)^2] = \int_{\Omega_\mathbf{Z}} \frac{\phi_1(\mathbf{z})}{\Delta}(\frac{\phi_1(\mathbf{z})}{\Delta} - \frac{\phi_2(\mathbf{z})}{\Delta})^2 d\mathbf{z},$$

donde ϕ_i^* son los potenciales ϕ_i normalizados, y Δ representa el peso total del potencial

$$\Delta = \sum_{\mathbf{Y}} \int_{\Omega_\mathbf{Z}} \phi(\mathbf{y}, \mathbf{z}) d\mathbf{z}.$$

Este valor Δ como constante de normalización normalización, tiene en cuenta las distancias entre el potencial original y el potencial resultado de la poda en relación con el peso total del potencial, pues esa distancia será significativa dependiendo del valor de dicho peso total.

Así pues, definimos tres tipos de poda sobre los árboles de probabilidad mixtos.

1. **Eliminación de términos exponenciales**

   Trataremos de eliminar aquellos términos exponenciales, en este caso, sin incluir el término constante, que tengan poco peso, o influencia en la función de densidad. Esto es, si tenemos en un nodo hoja la función

   $$f(\mathbf{z}) = k + \sum_{i=1}^{n} a_i e^{b_i \mathbf{z}}$$

   queremos detectar aquellos términos exponenciales $a_i e^{b_i \mathbf{z}}$ que tengan menor influencia. Para ello calculamos el *peso*, p_i, de cada factor:

   $$p_i = \int_{\Omega_{\mathbf{Z}}} a_i e^{b_i \mathbf{z}} d\mathbf{z}.$$

   Si este peso en valor absoluto $|p_i|$ es menor que un determinado valor δ prefijado, entonces estaríamos en condiciones de podar ese término, siempre y cuando el error que cometamos no supere el ϵ prefijado.

   Para podar el término y no modificar el peso total de la función hemos de añadir al término independiente una constante, que en este caso es $\frac{k}{|\mathbf{Z}|}$, donde $|\mathbf{Z}| = \int_{\Omega_{\mathbf{Z}}} d\mathbf{z}$

PODA_TERMINOS($\mathcal{T}$,δ,ϵ)

ENTRADA: un árbol de probabilidad mixto $\mathcal{T}$ cuyos hijos son funciones MTE.
SALIDA: el árbol resultado de la poda.

a) Sea X la etiqueta del nodo raíz: Para cada hijo de X:

   1) Sea $\phi(\mathbf{z}) = k + \sum_{i=1}^{m} a_i e^{b_i \mathbf{z}}$ su función MTE.
   2) Para i:=1 hasta m

      a' Sea $p_i = \int_{\Omega_{\mathbf{Z}}} a_i e^{b_i \mathbf{z}} d\mathbf{z}$.

      b' Si $|p_i| < \delta$

      - Sea $k' = k + \frac{p_i}{|\mathbf{Z}|}$.

- Sea $\phi_P(\mathbf{z}) = k' + \sum_{j \neq i} a_j e^{b_j \mathbf{z}}$.
- Sea $\mathcal{T}_P$ el árbol resultado de sustituir ϕ por ϕ_P en este hijo.
- Si $D(\mathcal{T}, \mathcal{T}_P) < \epsilon$, hacer $\mathcal{T} = \mathcal{T}_P$

b) DEVUELVE $\mathcal{T}$.

---

2. **Unificación de dos hijos de una variable continua**

Sea un árbol $\mathcal{T}$, cuya etiqueta del nodo raíz es una variable continua X, tal que sus hijos son funciones MTE. Por defecto, el dominio de X está definido por intervalos, y para cada uno de esos intervalos, I_j tenemos definida una función de la forma $f_j(\mathbf{z}) = k^j + \sum_{i=1}^{n} a_i^j e^{b_i^j \mathbf{z}}$. Puede que esas funciones sean muy parecidas todas entre si, y que por lo tanto podamos unificar varios de esos hijos en uno solo, es decir, transformar dos funciones $f_{j_1}(\mathbf{z})$ y $f_{j_2}(\mathbf{z})$ definidas para X sobre dos intervalos contiguos I_{j_1} e I_{j_2} en una sola función $f(\mathbf{z})$ definida para la variable X sobre el intervalo $I_{j_1} \cup I_{j_2}$.

Esta nueva función sería la media de las dos funciones anteriores ponderada por su correspondiente probabilidad, es decir, si en este caso

$$p_{j_1} = \int_{\Omega_{\mathbf{Z}}} f_{j_1}(\mathbf{z}) d\mathbf{z} \quad y \quad p_{j_2} = \int_{\Omega_{\mathbf{Z}}} f_{j_2}(\mathbf{z}) d\mathbf{z}$$

son los correspondientes pesos de las funciones, la función por la que podamos es

$$f(\mathbf{z}) = \frac{p_{j_1} f_{j_1}(\mathbf{z}) + p_{j_2} f_{j_2}(\mathbf{z})}{p_{j_1} + p_{j_2}}$$

El proceso de unión de los hijos de la variable X se va repitiendo para todos los hijos con intervalos contiguos, de tal forma que podríamos llegar a una situación de unión completa de todos los intervalos en uno, lo que indicaría que la presencia del nodo con la variable X en el árbol es innecesaria, ya que no aporta ninguna discriminación de los valores del potencial, y por tanto procederíamos a su sustitución por el potencial definido en su único hijo.

PODA_UNION($\mathcal{T}$,ϵ)

ENTRADA: un árbol de probabilidad mixto $\mathcal{T}$ cuyo nodo raíz es una variable continua, con todos sus hijos funciones MTE.
SALIDA: el árbol resultado de la poda.

a) Sea X la etiqueta del nodo raíz.

b) Si sólo hay un hijo:

Cambiamos la etiqueta del nodo raíz y la sustituimos por la probabilidad del único hijo.

c) Si hay más de un hijo

- Repetir
  1) Sean $\phi_1(\mathbf{z})$ y $\phi_2(\mathbf{z})$ dos potenciales de dos hijos definidos para la variable X en los intervalos contiguos I_1 e I_2 respectivamente.
  2) Sean $p_1 = \int_{\Omega_\mathbf{Z}} \phi_1(\mathbf{z})$ y $p_2 = \int_{\Omega_\mathbf{Z}} \phi_2(\mathbf{z})$
  3) Sea
  $$\phi(\mathbf{z}) = \frac{p_1\phi_1(\mathbf{z}) + p_2\phi_2(\mathbf{z})}{p_1 + p_2}$$
  un potencial definido para X sobre $I = I_1 \cup I_2$
  4) Sea $\mathcal{T}_P$ el árbol resultado de sustituir en $\mathcal{T}$ estos dos hijos primeros del nodo raíz por un nodo con etiqueta $\phi(\mathbf{z})$
  5) Si $D(\mathcal{T}, \mathcal{T}_P) < \epsilon$, hacer $\mathcal{T} :=$ **PODA_UNION($\mathcal{T}_P$,ϵ)**.
- Hasta que todos los pares de hijos contiguos hayan sido explorados.

d) DEVUELVE $\mathcal{T}$.

---

3. **Poda de variables discretas**

Tal y como se aprenden las distribuciones MTE, tanto las marginales como las condicionadas, los valores de las variables discretas no aparecerán nunca en la expresión funcional de los potenciales, ni en los originales ni en los intermedios que se generan al combinar, restringir o marginalizar los potenciales originales.

Esto es debido a que sólo se tiene en cuenta el valor de la variable discreta a la hora de partir el dominio, incluso los potenciales marginales definidos para variables discretas son equivalentes a tablas de probabilidad.

Por tanto, si Y es una variable discreta en un nodo en un árbol de probabilidad mixto, cuyos hijos son funciones MTE, estas funciones estarán definidas sobre las variables continuas del árbol, o bien serán exclusivamente números reales. Si estamos en el primer caso podemos aplicar la poda de términos exponenciales, y si estamos en el segundo caso, podamos el árbol tal y como definen Salmerón, Cano, y Moral (2000).

Para este tipo de poda se utiliza otro tipo de distancia, la *divergencia de Kullback-Leibler* (Kullback y Leibler 1951), que ya vimos en la definición (2.3) (capítulo 2), pero entonces para dos funciones de densidad continuas.

Definición 7.3 *Se define la divergencia de Kullback-Leibler entre dos potenciales ϕ_1 y ϕ_2 definidos sobre la misma variable discreta Y, representados por $p_i \quad i = 1, \ldots, d$ el primero y por $q_i \quad i = 1, \ldots, d$ el segundo como*

$$DK(\phi_1, \phi_2) = \sum_i p_i log(\frac{p_i}{q_i})$$

La poda en esta situación se lleva a cabo como se muestra en (Salmerón, Cano, y Moral 2000): Sea $\mathcal{T}$ un árbol que representa un potencial ϕ cuya raíz es la variable discreta Y, y sus hijos son todos probabilidad, todos ellos números reales, $p_i \in I\!R$ con $i = 1, \ldots, d$. Sea $\overline{p} = \frac{\sum_i p_i}{d}$. Se sustituye el nodo Y por $\overline{p}$ si $D(\phi, \overline{p}) < \xi$, donde ξ representa el límite para el incremento de la distancia.

Lo que estamos haciendo al podar las variables discretas del árbol de esta forma es eliminar aquellas hojas que se asemejan a una distribución uniforme, de tal forma

que las podemos representar mediante un único valor sin demasiada pérdida de información.

Para compararla con la distribución uniforme, Salmerón, Cano, y Moral (2000) calculan ξ como la entropía (Kullback y Leibler 1951) de la distribución de una variable binaria que toma como valores de probabilidad $0.5-\epsilon$ y $0.5+\epsilon$, esto es, una distribución que se diferencia de una distribución uniforme en ϵ.

$$\xi = -((0.5-\epsilon)log(0.5-\epsilon) + (0.5+\epsilon)log(0.5+\epsilon))$$

El algoritmo se puede enunciar así:

PODA_DISCRETA($\mathcal{T}$,ϵ)

ENTRADA: un árbol de probabilidad mixto $\mathcal{T}$, con raíz una variable discreta.
SALIDA: el árbol resultado de la poda.

a) Sea Y la etiqueta del nodo raíz, y sea d el número de hijos de Y.

1) Sea
$$\overline{p} = \frac{1}{d}\sum_{i=1}^{d} p_i$$
2) Sea $\xi = -((0.5-\epsilon)log(0.5-\epsilon) + (0.5+\epsilon)log(0.5+\epsilon))$
3) Si $\sum_{i=1}^{d} p_i log(\frac{p_i}{\overline{p}}) < \xi$, hacer $\mathcal{T} := \overline{p}$.

b) DEVUELVE $\mathcal{T}$.

Una vez que ya tenemos los tres métodos de poda que podemos efectuar sobre un potencial, veamos cómo los agrupamos en un único método:

PODA($\mathcal{T}$,δ,ϵ)

ENTRADA: un árbol de probabilidad mixto, $\mathcal{T}$.
SALIDA: el árbol resultado de la poda.

1. Sea X la etiqueta del nodo raíz de $\mathcal{T}$.

   a) Si los hijos de X son probabilidad:

      1) Si X es continua:

         a' Hacer $\mathcal{T}$:= **PODA_UNION($\mathcal{T}$,ϵ)**.

         b' Hacer $\mathcal{T}$:= **PODA_TERMINOS($\mathcal{T}$,δ,ϵ)**.

      2) Si X es discreta

         a' Hacer $\mathcal{T}$:= **PODA_TERMINOS($\mathcal{T}$,δ,ϵ)**.

         b' HAcer $\mathcal{T}$:= **PODA_DISCRETA($\mathcal{T}$,ϵ)**.

      3) Si no:

         a' Si X es continua, para cada hijo :

         - Sean max y min los extremos del intervalo correspondientes a este hijo.
         - Hacer $\mathcal{T}$:= **PODA($\mathcal{T}^{R(X\in(min,max))}$)**.

         b' si no, para cada hijo:

         - Sea a el valor correspondiente a la variable X para este hijo.
         - Hacer $\mathcal{T}$:= **PODA($\mathcal{T}^{R(X=a)}$)**.

   b) DEVUELVE $\mathcal{T}$.

Si la variable a podar es continua, primero intentamos unir la mayor cantidad de hijos posible y luego eliminamos los términos exponenciales menos importantes, dado

que si lo hiciésemos en orden inverso, al unir los intervalos nos estaríamos generando en ese hijo más términos exponenciales de los que teníamos.

Si la variable es discreta, primero eliminamos los términos exponenciales con la intención de que sus funciones MTE queden definidas únicamente sobre números reales, y así poder aplicar la poda discreta.

La forma de integrar estos métodos de poda con el algoritmo de propagación es podar todos aquellos árboles resultado de la combinación y de la marginalización de potenciales.

A diferencia de algoritmo *Penniless* original, sólo realizamos dos pasadas, las mismas que en el algoritmo exacto *Shenoy-Shafer* , ya que la poda no la hemos definido de forma condicionada a otro potencial, y por tanto el realizar más pasadas no nos proporcionaría más información.

Así pues el algoritmo *Penniless* aplicado a redes MTE es igual que el *Shenoy-Shafer*, pero tras combinar y tras marginalizar efectuamos la poda del árbol resultante.

Penniless_Simple($\mathcal{T}$,δ,ϵ)

ENTRADA: un árbol de cliques $\mathcal{T}$.
SALIDA: el árbol de cliques $\mathcal{T}$ tras la propagación.

1. Seleccionar un nodo raíz R.
2. Inicializamos los potenciales de los cliques.
3. Para cada $C \in ne(R)$,
   - **PS_PropagarArriba(R,C,δ,ϵ)**
4. Para cada $C \in ne(R)$,

- Calcular $\phi = \phi_R \cdot \left(\prod_{C_k \in ne(R) \setminus C} \phi_{C_k \to R} \right)$.
- Hacer ϕ:= **PODA(**ϕ,δ,ϵ**)**.
- Hacer $\phi := \phi^{\downarrow R \cap C}$.
- Calcular el mensaje $\phi_{R \to C}$:=**PODA(**ϕ,δ,ϵ**)**.
- **PS_PropagarAbajo(**R,C,δ,ϵ**)**

5. DEVOLVER $\mathcal{T}$.

---

donde ahora **PS_PropagarArriba** y **PS_PropagarAbajo** son como los correspondientes **PropagarArriba** y **PropagarAbajo** del algoritmo **Shenoy-Shafer** pero incorporando la poda. Ambos se detallan a continuación.

PS_PropagarArriba(C_1,C_2,δ,ϵ**)**

---

1. Para cada $C \in ne(C_2) \setminus C_1$,
   - **PS_PropagarArriba(**C_2,C,δ,ϵ**)**
2. Calcular $\phi = \phi_{C_2} \cdot \left(\prod_{C_k \in ne(C_2) \setminus C_1} \phi_{C_k \to C_2} \right)$.
3. Hacer ϕ:= **PODA(**ϕ,δ,ϵ**)**.
4. Hacer $\phi := \phi^{\downarrow C_1 \cap C_2}$.
5. Calcular el mensaje $\phi_{C_2 \to C_1}$:=**PODA(**ϕ,δ,ϵ**)**.

PS_PropagarAbajo(C_1,C_2,δ,ϵ)

1. Para cada $C \in ne(C_2) \backslash C_1$,

   - Calcular $\phi = \phi_{C_2} \cdot \left(\prod_{C_k \in ne(C_2) \backslash C_1} \phi_{C_k \to C_2} \right)$.
   - Hacer ϕ:= **PODA(ϕ,δ,ϵ)**.
   - Hacer $\phi := \phi^{\downarrow C \cap C_2}$.
   - Calcular el mensaje $\phi_{C_2 \to C}$:=**PODA(ϕ,δ,ϵ)**.
   - **PS_PropagarAbajo(C_2,C,δ,ϵ)**

7.2.2. Algoritmo de propagación basado en MCMC

Este otro algoritmo se basa en la simulación de las variables de la red. En esta situación, las probabilidades tras la propagación se pueden estimar por medio del método *MCMC (Markov Chain Monte Carlo)*, tal y como comentamos en el primer capítulo.

La propagación MCMC genera primero una muestra de las variables de la red para después usar dicha muestra para estimar la distribución de las variables de interés. La muestra se genera a partir de una configuración inicial de las variables, donde las variables observadas $\mathbf{E}$ son inicializadas a las observaciones $\mathbf{e}$. Esta configuración inicial se puede obtener por muestreo hacia delante (Henrion 1988).

Una vez que hemos obtenido esta configuración inicial, obtenemos una nueva simulando cada variable no observada X_i de acuerdo con su distribución condicionada a su envolvente de Markov restringida a los valores de la configuración actual.

El procedimiento de simulación descrito arriba se puede aplicar a redes MTE; el único aspecto que habría que clarificar es cómo simular un valor para una variable con distribución MTE (Moral, Rumí, y Salmerón 2001).

7.2.2.1. Simulando de una distribución MTE

Cuando vamos a simular una variable X_i a partir de su distribución condicional $f_i(x_i|\mathbf{W}_{X_i})$ restringida a los valores $\mathbf{w}_{x_i}$ en la configuración actual, simulamos de una distribución que depende sólo de X_i. Si X_i es discreta, es sencillo simular un valor para ésta variable: generamos un número aleatorio y aplicamos el método de inversión (Rubinstein 1981).

Si la variable X_i es continua, podemos encontrar su función de densidad definida por intervalos, y además, el método de inversión no se puede aplicar en general a densidades MTE. Sin embargo sí que podemos simular valores aplicando el *método de composición* (Rubinstein 1981).

Supongamos que la densidad de la variable a simular viene definida como:

$$f(x) = f_i(x) \quad \text{si} \quad \alpha_i \leq x < \beta_i, \quad i = 1, \ldots, k \ ,$$

donde cada una de las funciones f_i son de la forma

$$f_i(x) = a_0 + a_1 e^{k_1 x} + a_2 e^{k_2 x} + \cdots + a_n e^{k_n x} \quad \alpha_i \leq x < \beta_i \ . \tag{7.1}$$

La forma de simular a partir de $f(x)$ por el método de composición es escoger una de las funciones f_i con probabilidad igual a $\int_{\alpha_i}^{\beta_i} f_i(x)dx$ y entonces simular un valor dentro del intervalo (α_i, β_i) a partir de una densidad f_i^* proporcional a f_i:

$$f_i^*(x) = \frac{f_i(x)}{\int_{\alpha_i}^{\beta_i} f_i(x)dx} \quad \alpha_i \leq x < \beta_i \ ,$$

que es también una función de la forma (7.1).

Para simular a partir de f_i^* distinguiremos dos situaciones:

- Todos los a_i de (7.1) son positivos:

  Entonces podemos aplicar de nuevo el método de composición.

  1. Descomponemos la densidad f_i^* como una suma ponderada de densidades.

  $$f_i^*(x) = p_1 f_1'(x) + \cdots + p_m f_m'(x) \tag{7.2}$$

  con

  $$\sum_{j=1}^{m} p_j = 1 \ .$$

  2. Generamos dos números aleatorios u_1 y u_2.
  3. Usamos u_1 para escoger una de las f_j' con probabilidad p_j, y usamos u_2 para obtener un valor para la variable X aplicando el método de inversión a la función de distribución correspondiente a f_j'.

- Si algún/os a_i de (7.1) son negativos, usaremos el método propuesto por Bignami y Matties (1971):

  1. Descomponemos la densidad f_i^* como una suma ponderada de densidades

  $$f_i^*(x) = p_1 f_1'(x) + \cdots + p_m f_m'(x) \tag{7.3}$$

  con

  $$\sum_{j=1}^{m} p_j = 1 \ .$$

  2. Sea N el conjunto de todos los i tales que $p_i < 0$. Escribimos entonces

  $$\begin{aligned} f_i^*(x) &= \left(\sum_{i\in N} p_i\right)\left(\frac{1}{\sum_{i\in N} p_i}\sum_{i\in N} p_i f_i'(x)\right) + \sum_{i\notin N} p_i f_i'(x) \\ &= \left(\sum_{i\in N} p_i\right) g(x) + \sum_{i\notin N} p_i f_i'(x) \end{aligned} \tag{7.4}$$

3. Simular un valor x^* a partir de la densidad $g(x)$. Este valor vendrá de $f_i'(x)$ con probabilidad $\frac{p_i}{\sum_{i\in N} p_i}$
4. Generar un número aleatorio u.
5. Si $u \leq \frac{f_i'(x^*)}{\sum_{i\in N} p_i f_i'(x^*)}$, aceptamos x^* como valor generado para X, si no repetimos a partir del paso 3.

Este método tiene una **eficiencia** (Rubinstein 1981) ef de

$$ef = \frac{1}{\sum_{i\notin N} p_i} \ ,$$

lo que quiere decir que el número esperado de valores x^* que hemos de generar para simular N valores de la variable X es de $\frac{1}{ef}N$.

La descomposición de (7.2) y (7.4) debe ser tal que podamos calcular la inversa de la función de distribución correspondiente a cada f_j'. Podemos obtener tal descomposición como sigue. Sea

$$c_j = \int_{\alpha_i}^{\beta_i} e^{k_j x} dx \quad j = 1, \ldots, n \ ,$$

entonces

$$f_j'(x) = \frac{1}{c_j} e^{k_j x}, \quad j = 1, \ldots, n$$

es una función de densidad en (α_i, β_i). Para $j = 0$,

$$c_0 = \int_{\alpha_i}^{\beta_i} dx = \beta_i - \alpha_i$$

y por tanto

$$f_0(x) = \frac{1}{c_0}$$

es una densidad en (α_i, β_i).

Así, multiplicando y dividiendo cada término por el correspondiente c_j podemos escribir

$$f_i^*(x) = \frac{1}{\int_{\alpha_i}^{\beta_i} f_i(x)dx}\left(a_0c_0\frac{1}{c_0} + a_1c_1\frac{1}{c_1}e^{k_1x} + a_2c_2\frac{1}{c_2}e^{k_2x} + \cdots + a_nc_n\frac{1}{c_n}e^{k_nx}\right) \quad \alpha_i \le x \le \beta_i$$

de dónde podemos tomar cómo pesos $p_j = \frac{a_jc_j}{\int_{\alpha_i}^{\beta_i} f_i(x)dx}$, $j = 0, \ldots, n$. Veamos que realmente estos pesos suman uno:

$$\begin{aligned}\sum_{j=0}^{n} p_j &= \frac{1}{\int_{\alpha_i}^{\beta_i} f_i(x)dx}\left(a_0c_0 + a_1c_1 + \cdots + a_nc_n\right)\\ &= \frac{1}{\int_{\alpha_i}^{\beta_i} f_i(x)dx}\left(a_0\int_{\alpha_i}^{\beta_i} dx + a_1\int_{\alpha_i}^{\beta_i} e^{k_1x}dx + \cdots + a_n\int_{\alpha_i}^{\beta_i} e^{k_nx}dx\right)\\ &= \int_{\alpha_i}^{\beta_i} f_i^*(x)dx = 1 \ .\end{aligned}$$

Veamos finalmente cómo podemos calcular de forma sencilla el inverso de la función de distribución de cada f_j', tal y como la hemos construido aquí. Si $j = 0$, la función de distribución es la uniforme en (α_i, β_i). Si $j > 0$, para $x \in (\alpha_i, \beta_i)$, la función de distribución es

$$F_j'(x) = \int_{-\infty}^{x} f_j'(t)dt = \int_{\alpha_i}^{x} \frac{1}{c_j}e^{k_jt}dt = \frac{1}{c_jk_j}\left(e^{k_jx} - e^{k_j\alpha_i}\right) \ .$$

Dado un número aleatorio u, $0 < u < 1$, su inverso se calcula así:

$$\begin{aligned}u = \frac{1}{c_jk_j}\left(e^{k_jx} - e^{k_j\alpha_i}\right) &\Rightarrow e^{k_jx} = c_jk_ju + e^{k_j\alpha_i} \Rightarrow\\ k_jx = \log\left(c_jk_ju + e^{k_j\alpha_i}\right) &\Rightarrow x = \frac{1}{k_j}\log\left(c_jk_ju + e^{k_j\alpha_i}\right) \ .\end{aligned}$$

Por tanto, para $j > 0$,

$$F_j'^{-1}(u) = \frac{1}{k_j} \log\left(c_j k_j u + e^{k_j \alpha_i}\right) \quad 0 < u < 1 \ .$$

7.2.2.2. Estimación a posteriori de las marginales

Una vez obtenida la muestra, las distribuciones a posteriori pueden calcularse de las siguientes dos maneras:

1. Estimar las probabilidades del tipo $P(a < X_i < b)$ contando aquellas configuraciones de la muestra para las cuales X_i cae en (a, b).
2. Usar las técnicas del capítulo 5 para estimar las densidades de cada variable.

7.3. Conclusiones

Hemos visto en este capítulo cómo adaptar a nuestras redes MTE tres de los métodos más utilizados a la hora de propagar probabilidades en las redes bayesianas.

Primero hemos demostrado que el algoritmo exacto Shenoy-Shafer (Shenoy y Shafer 1990) se puede aplicar a las redes MTE, dado que todos los potenciales intermedios que se obtienen son a su vez potenciales MTE. De hecho, no sólo podemos aplicar el algoritmo *Shenoy-Shafer*, si no que podríamos utilizar cualquiera que no demandase la operación división a la hora de propagar.

Pero sabemos que hay ciertos problemas que con los métodos exactos no se pueden resolver, debido fundamentalmente al tamaño de los potenciales que se generan a la hora de propagar, que resultan demasiado costosos de calcular y de almacenar. Para resolver este tipo de problemas surgen los *métodos aproximados*. De todos ellos nos hemos centrado en dos, para ver cómo los podemos adaptar a las redes MTE. Como segundo método de propagación hemos introducido una versión simplificada del algoritmo *Penniless* (Cano, Moral, y Salmerón 2000), que se basa en la idea de *poda* de un árbol. Por *poda* entendemos la simplificación de un árbol por otro cuyo tamaño sea menor, con la menor pérdida de información posible. Hemos introducido tres métodos

de poda distintos, dependiendo de la situación que nos encontremos en el árbol, para acabar describiendo cómo introduciríamos esta nueva operación en el algoritmo exacto *Shenoy-Shafer* para obtener el método *Penniless Simple para redes MTE*.

El último método de propagación que hemos descrito para las redes MTE se basa en *MCMC*, que ya describimos para redes discretas en el capítulo 1. El funcionamiento de dicho algoritmo en nuestro caso particular es el mismo, una vez definido el proceso de simulación de valores de una densidad MTE univariante.

Capítulo 8

Conclusiones y Líneas Abiertas

Hemos estudiado las alternativas actuales en el tratamiento de redes bayesianas con variables discretas y continuas: la discretización (Kozlov y Koller 1997) y el modelo condicional gaussiano (Lauritzen 1992). Como resultado de este estudio, y tratando ampliar la clase de problemas tratables desde el punto de vista de las redes bayesianas, hemos propuesto el modelo MTE, que trata de capturar las buenas propiedades de los dos enfoques anteriormente referidos.

El modelo MTE mejora en expresividad, y por tanto en poder de ajuste, a los modelos discretizados, y a la vez elimina la restricción que el modelo condicional gaussiano impone a las relaciones entre las variables (variables continuas no pueden tener hijos discretos).

Presentamos como estructura de representación de los potenciales MTE los llamados árboles de probabilidad mixtos, que aparecen como una extensión natural de los árboles de probabilidad, y sobre los cuales se pueden especificar las operaciones básicas de manipulación de información probabilística (combinación, restricción y marginalización).

Hemos propuesto esquemas para la estimación de los dos tipos de densidades MTE que pueden aparecer en una red bayesiana: distribuciones univariantes (las distribuciones a priori sobre los nodos raíz de la red) y distribuciones condicionadas (correspondientes a los no raíz). Estos esquemas se basan en la regresión exponencial y, en el caso

de las condicionadas, están inspirados en los árboles de clasificación.

Por último, para completar la utilidad de modelos prácticos a la hora de obtener conclusiones ante la llegada de nueva información, hemos demostrado la posibilidad de aplicar algoritmos de propagación exacta que no requieran la operación de división. En concreto, hemos particularizado el estudio al caso del método de Shenoy-Shafer (Shenoy y Shafer 1990). También hemos propuesto dos algoritmos aproximados: uno de ellos basado en la propagación Penniless (Cano, Moral, y Salmerón 2000), para lo cual hemos definido métodos de poda de los árboles mixtos de probabilidad, y otro basado en MCMC, para lo cual tuvimos que resolver el problema de la simulación de valores a partir de densidades MTE, usando los métodos de inversión y composición.

Todo el software desarrollado para implementar los métodos propuestos en esta memoria se ha realizado usando el sistema Elvira (Consortiun 2002), que, al ser de dominio público, posibilitará que otros investigadores puedan utilizar los modelos aquí presentados sin necesidad de volver a programarlos.

Pensamos que el trabajo presentado en esta memoria tiene perspectivas de continuidad, pues los problemas resueltos a su vez sugieren el estudio de nuevos problemas o el refinamiento de los métodos desarrollados. En ese sentido, podemos citar las siguientes líneas de actuación para trabajos futuros:

- Mejorar los algoritmos de aprendizaje, con la intención de incorporar las variables condicionantes a la expresión funcional de la distribución condicionada. Para ello, habrá que estudiar las restricciones sobre los parámetros para garantizar que un potencial MTE sea una densidad condicionada.

- Estudiar el aprendizaje estructural de redes MTE. Esto implica la propuesta de métricas que tengan en cuenta la expresión de las densidades MTE.

- Adaptar más algoritmos de propagación de probabilidades a las redes MTE, con el objetivo de ampliar lo máximo posible el abanico de problemas tratables. En ese sentido, cabe plantearse diseñar algoritmos de muestreo por importancia similares a los desarrollados en (Salmerón, Cano, y Moral 2000; Salmerón y Moral

2001), que han proporcionado buenos resultados para variables discretas. También se puede mejorar el algoritmo Penniless simple, mediante la definición de información condicional para potenciales MTE que permita podar un potencial en un mensaje teniendo en cuenta el mensaje en la dirección opuesta.

- Aplicar las redes MTE a problemas de clasificación donde hay involucradas variables continuas que no se ajustan a modelos gaussianos.

Apéndice

A continuación explicaremos las herramientas utilizadas para la implementación de los diferentes algoritmos presentados en la memoria, y haremos una breve descripción de los métodos utilizados para ello.

El paquete Elvira

La tarea de implementación de los algoritmos ha sido mucho más asequible debido a la existencia del sistema *ELVIRA* (Consortiun 2002)[1] Se trata de una herramienta implementada en JAVA (Castillo, Cobo, Gómez, y Solares 1997) para construir y manipular modelos gráficos probabilísticos.

Permite trabajar tanto con redes bayesianas como con diagramas de influencia, así como también seleccionar el tipo de estructura de datos a utilizar.

Dispone de un interfaz gráfico par interactuar con el usuario, pero la principal ventaja es que podemos acceder libremente a todos los paquetes que ya tiene implementados.

Los paquetes en los que se organizan las clases del sistema ELVIRA son los siguientes:

1. **Base de datos** (`database`). Soporta todas las clases necesarias para el manejo de Bases de Datos.

[1]Se puede conseguir gratuitamente en la dirección http://leo.ugr.es/ elvira.

2. **Interfaz gráfico** (`gui`). Para interaccionar con el usuario en un entorno de ventanas en vez de desde la linea de comandos.

3. **Inferencia** (`inference`). Aquí están agrupadas todas las clases relacionadas con los diferentes algoritmos de propagación, tanto exactos como aproximados.

4. **Aprendizaje** (`learning`). Los diferentes método de aprendizaje, tanto estructural como paramétrico los podemos encontrar aquí ya implementados.

5. **Potenciales** (`potential`). Las clases relacionadas con las estructuras de datos para manejar la información numérica.

A la hora de implementar un nuevo algoritmo en ELVIRA sólo hemos de concentrarnos en como implementar los aspectos novedosos de éste, dado que los problemas de estructuras de datos y algoritmos o métodos ya conocidos que se hayan de utilizar ya se encuentran implementados y las correspondientes clases funcionando.

En la siguiente sección comentaremos cómo se realizaron las implementaciones de los diferentes algoritmos de esta memoria en el sistema ELVIRA.

MTE en Elvira

En esta sección explicaremos brevemente las diferentes clases implementadas en el paquete Elvira para poder manejar las redes y potenciales MTE. Podemos estructurar estas clases según su utilización: *estructura, aprendizaje y propagación.*

Estructura

Para poder definir un potencial MTE de forma correcta han sido necesarias dos clases, `ContinuousProbabilityTree` y `MixtExpDensity`, que pasamos a explicar a continuación:

1. `ContinuousProbabilityTree` : Con esta clase implementamos la estructura que definimos en el capitulo 4 para representar los potenciales MTE, los *árboles de probabilidad mixtos*.

   Se trata de una estructura recursiva; un `ContinuousProbabilityTree` se compone básicamente de su etiqueta y de sus hijos, que serán a su vez `ContinuousProbabilityTree`, si el nodo es interior, y `MixtExpDensity` si no lo son.

   Más concretamente, un objeto `ContinuousProbabilityTree` tiene los siguientes campos:

   - `child:` Es un vector, que contendrá los hijos de esta variable.
   - `cutPoints:` Vector que contendrá los extremos de los intervalos en los que dividimos el dominio de la variable si es continua. Si es discreta estará vacío.
   - `label:` Un entero que indica el tipo de etiqueta del nodo, variable discreta, variable continua o probabilidad.
   - `var:` En el caso de ser un nodo interior, la variable que representa.
   - `value:` Se trata de un objeto `MixtExpDensity` que representa el valor de probabilidad de las variables que hay en el camino hasta él. Será distinto de `NULL` sólo si la etiqueta del nodo es probabilidad.
   - `numSplits:` Cantidad de cortes que hacemos en el dominio de las variables continuas a la hora de aprender las densidades condicionadas.
   - `numTerms:` Cantidad de términos exponenciales, incluyendo la constante, que permitimos a los potenciales. En principio restringido a 3 en los algoritmos de aprendizaje.

   Hay definidos varios constructores para esta clase:

   - `ContinuousProbabilityTree():` Crea un árbol vacio.
   - `ContinuousProbabilityTree(Continuous var, Vector cp):` Crea un árbol cuya variable es var, y cuyo campo `cutPoints` es cp.

- `ContinuousProbabilityTree(Continuous variable, Vector cp, double x)`: Crea un árbol cuya variable es `var`, cuyo campo `cutPoints` es `cp` y cuyos hijos son todos probabilidad con valor constante `x`.
- `ContinuousProbabilityTree(double p)`: Crea un árbol probabilidad cuya densidad es constante e igual a `p`.
- `ContinuousProbabilityTree(FiniteStates variable)`: Crea un árbol cuya variable es una discreta e igual a `variable`.
- `ContinuousProbabilityTree(MixtExpDensity f)`: Crea un árbol probabilidad cuyo campo `value` es `f`.

Los métodos definidos en esta clase son los usuales, como los de acceso a los diferentes campos de la clase, y también están implementados los diferentes algoritmos que realizan las operaciones entre árboles, como son:

- `add(ContinuousProbabilityTree tree1, ContinuousProbabilityTree tree2)`: Devuelve un `ContinousProbabilityTree` resultado de la suma de estos dos.
- `addVariable(Continuous variable)`: Devuelve un `ContinousProbabilityTree` resultado de eliminar la variable continua integrando con respecto a ella.
- `addVariable(FiniteStates variable)`: Devuelve un `ContinousProbabilityTree` resultado de eliminar la variable discreta sumando en todos sus casos.
- `combine(ContinuousProbabilityTree tree1, ContinuousProbabilityTree tree2)`: Devuelve un `ContinuousProbabilityTree` resultado de la combinación de estos dos árboles.
- `integral(Continuous variable, double lower, double upper)`: Devuelve un `ContinuousProbabilityTree` resultado de integrar el árbol con respecto de `variable` entre `lower` y `upper`.
- `restrict(ContinuousConfiguration c)`: Devuelve un `ContinuousProbabilityTree` resultado de restringirlo a una determinada configuración de las variables (discretas y continuas).

2. `MixtExpDensity` : Con esta clase implementamos la estructura básica de un potencial MTE, la combinación lineal de funciones exponenciales cuyo exponente es una combinación lineal de las variables en cuestión. Fundamentalmente esta clase actúa como nodo hoja de un `ContinuousProbabilityTree`.

   Un objeto `MixtExpDensity` tiene los siguientes campos:

   - `independent:` El término independiente.
   - `terms:` Un vector que contiene, en cada elemento una combinación lineal de variables, que serán los exponentes de las funciones exponenciales.
   - `factors:` Un vector que contiene, para cada elemento el coeficiente que multiplica a la correspondiente exponencial.

   Hay definidos tres constructores para esta clase:

   - `MixtExpDensity():` Crea una densidad constante a 1.
   - `MixtExpDensity(double x):` Crea una densidad constante a `x`.
   - `MixtExpDensity(double a, double b, double d, double f, double k, Continuous x):` Crea una densidad de la forma $k + ae^{bx} + de^{fx}$, que es la que creamos cuando aprendemos.

   Los métodos implementados son aquellos que permiten realizar las operaciones, es decir, combinar, sumar, integrar y restringir, además de los necesarios para gestionar correctamente la clase, es decir, aquellos que dan acceso a los distintos campos de la clase, o los que muestran por pantalla el valor de dichos campos.

Aprendizaje

Para poder realizar el aprendizaje de densidades condicionadas MTE, además de algunos métodos de las dos clases anteriores, definimos una nueva clase, `MTELearning`, que crea, a partir de una base de datos, las distintas densidades condicionadas que se le indiquen. Veamos con algo más de detalle esta clase y los métodos de las anteriores que usamos para aprender:

a) `MTELearning`: Esta clase es la encargada directa del aprendizaje de densidades condicionadas.

Un objeto `MTELearning` tiene los siguientes campos:

- `cases`: Un objeto `DataBaseCases`, es decir, una base de datos con los valores de cada variable para cada individuo.
- `variables`: Las variables para las que está definida la base de datos.

Tiene únicamente dos constructores,

- `MTELearning()`: Crea el objeto vacío.
- `MTELearning(DataBaseCases d)`: Crea un `MTELearning` con la base de datos `d`.

Los métodos implementados son:

- `learnConditional(Continuous y, NodeList X, DataBaseCases cases, int intervals)`: Devuelve un `ContinuousProbabilityTree` resultado de aprender para la base de datos `cases` una densidad condicionada para la variable `y` dados sus padres `X`, partiendo el dominio de las variables continuas en `intervals` intervalos distintos.
- `Splitting(ContinuousProbabilityTree t, NodeList X, Continuous y, ContinuousCaseListMem sample, int intervals)`: Método recursivo que va creando el árbol resultado del aprendizaje según la muestra, `sample`, que cae en cada uno de los nodos que se van creando.

Cuando llegamos a una hoja, la forma de estimar el potencial MTE se hace desde las dos clases anteriores:

b) `ContinuousProbabilityTree`: Los dos métodos que se utilizan en esta clase son:

- `learnUnivariate(Continuous X, Vector values, int intervals)`: Devuelve un `ContinuousProbabilityTree` resultado de aprender la densidad univariante para la variable `X`, cuyos valores están en el vector `values`, y cuyo dominio dividiremos en `intervals` intervalos distin-

tos. La forma de partir el dominio de la variable la acomete el método siguiente

- `domainSplitting(Vector x, Vector y, int intervals)`: Devuelve un vector con los límites inferiores y superiores de los distintos intervalos en los que dividiremos el dominio de la función. Los vectores `x` e `y` son los resultantes de obtener la densidad empírica de los datos, densidad que obtenemos con el método:
- `empiricDensity(Vector data, double longitud)`: Devuelve un vector con dos elementos, uno para los valores de la variable y otro para los valores de la densidad, usando el método de intervalos con igual longitud.
- `empiricEqualNumberOfPoints(Vector values, int intervals)`: Devuelve un vector con dos elementos, uno para los valores de la variable y otro para los valores de la densidad, seleccionando un número determinado de intervalos con igual número de valores en cada uno de ellos.

c) **MixtExpDensity**: Una vez que tenemos el dominio de la función dividido en diferentes intervalos, sólo nos queda estimar los parámetros del potencial MTE, con los siguientes métodos de la clase:

- `estimate(Continuous var, Vector x, Vector y)`: Devuelve un `MixtExpDensity` resultado de aprender la densidad empírica `(x,y)`, para lo cual utiliza estos otros métodos:
- `checkData(Vector x, Vector y)`: Transforma los vectores `(x,y)` para poder aplicar una regresión exponencial.
- `exponentialRegression(Vector x, Vector y)`: Calcula la regresión exponencial $y = ae^{bx}$ para la densidad empírica, usando para ello el método:
- `linearRegression(Vector x, Vector y)`: Calcula la regresión lineal de la densidad empírica `(x,y)`.

Propagación

Para cada uno de los tres tipos de propagación que hemos aplicado a las redes MTE hemos creado una nueva clase, pero estos algoritmos de propagación no trabajan directamente sobre la clase `ContinuousProbabilityTree`, sino sobre otra clase que recoge la información de los potenciales: `PotentialContinuouPT`.

1. **PotentialContinuousPT:** Esta clase es una transición entre la estructura y la propagación de probabilidades. A la hora de hacer los cálculos, los distintos algoritmos trabajan sobre un objeto de la clase `PotentialContinuousPT`, que tiene los siguientes campos:

   - `values` : Es un `ContinuousProbabilityTree`, el árbol que representa este potencial.

   Los métodos que hay implementados para esta clase son aquellos necesarios para realizar la propagación, tanto exacta como aproximada, es decir, los que realizan las sumas, combinaciones, las restricciones, simulación de valores de variables, poda y demás operaciones. Estos métodos se apoyan fundamentalmente en sus homólogos de la clase `ContinuousProbabilityTree`.

   Si pasamos a las clases de propagación propiamente dichas, todas son subclases de una interfaz, `Propagation`, que entre sus campos cuenta con:

   - `network` : La red en la que queremos propagar las probabilidades, con un conjunto de evidencias:
   - `observations` : Las observaciones que hemos de incorporar a la red.
   - `results` : Un vector donde guardaremos en cada componente la marginal de cada variable.

   Estas dos clases, a su vez heredan de las correspondientes clases discretas, es decir, implementadas pero sólo para variables discretas.

   El algoritmo exacto Shenoy-Shafer y el Penniless Simple para redes MTE los llevamos a cabo mediante la siguiente clase:

2. **MTESimplePenniless:** Un objeto de esta clase tiene, además de los campos de la clase `Propagation`, los de la clase discreta de la que hereda, entre los que destacamos:

   - `joinTree:` El árbol de cliques sobre el que realizamos la propagación. Para calcular este árbol de cliques usamos las clases `ContinuousJoinTree` y `ContinuousTriangulation`, que de nuevo son las versiones continuas de otras clases, encargadas de calcular el árbol de cliques.

   Esta clase tiene dos constructores:

   - `MTESimplePenniless(Bnet b, Evidence e):` Crea un objeto `MTESimplePenniless` a partir de la red `b` y las evidencias `e`.
   - `MTESimplePenniless(Bnet b):` Crea un objeto `MTESimplePenniless` a partir de la red `b`.

   Los métodos implementados en la clase son los necesarios para poder realizar la propagación, como por ejemplo:

   - `propagate(`δ`,`ϵ`):` Se encarga de llevar a cabo la propagación, es decir, es el encargado de llamar a los correspondientes métodos. Si δ y ϵ son cero, eso significa que estamos aplicando el algoritmo exacto de Shenoy-Shafer, ya que no admitiríamos poda alguna.
   - `initMessages():` Inicializa todos los potenciales del árbol de cliques a uno, así como todos los potenciales de los mensajes.
   - `navigate(NodeJoinTree sender):` Es el método que comienza el paso de mensajes, a partir de un nodo raíz `sender`, primero de la raíz a las hojas y luego de las hojas a la raíz.
   - `navigateUp(NodeJoinTree sender):` Método que transmite los mensajes de la raíz a las hojas.
   - `navigateDown(NodeJoinTree sender):` Método que transmite los mensajes de las hojas a la raíz.

     Estos dos métodos anteriores calculan los distintos mensajes haciendo una llamada al siguiente método:

- `sendMessage(NodeJoinTree sender, NodeJoinTree recipient)`: Calcula el mensaje que el nodo `sender` envía al nodo `recipient`, multiplicando el potencial asociado al nodo `sender` por los potenciales de todos los mensajes que recibe, y marginalizando sobre la intersección de `sender` y `recipient`.
- `computeMarginals()`: Calcula las marginales para todas y cada una de las variables de la red, y guarda dichos potenciales marginales normalizados en el vector `results`.

La poda de los árboles está implementada en la clase

- **ContinuousProbabilityTree:**

  Dado que la poda es algo propio de los árboles, en esta clase tenemos los siguiente métodos:

  - `prune(ContinuouProbabilityTree cpt, double` ϵ`, double` δ`)`: Este método es el general de poda, que va llamando a cada uno de los tres particulares según sea el árbol. Devuelve un `ContinuousProbabilityTree` resultado de la poda.
  - `pruneJoinInt(ContinuouProbabilityTree cpt, double` ϵ`)`: Implementa el método de poda que realiza la unión de dos hijos de intervalos contiguos de una variable continua. Devuelve un `ContinuousProbabilityTree` resultado de la poda.
  - `pruneDeleteExpTerm(ContinuouProbabilityTree cpt, double` ϵ`, double` δ`)`: Método que va eliminando términos exponenciales de una función MTE. Devuelve un `ContinuousProbabilityTree` resultado de la poda.
  - `pruneDiscrete(ContinuouProbabilityTree cpt, double` ϵ`)`: Poda una variable discreta, y la sustituye por su media. Devuelve un `ContinuousProbabilityTre` resultado de la poda.

El método de propagación aproximada por MCMC la realizamos con la siguiente clase:

3. **ContinuousMCMC:** Es una clase que hereda de `MarkocChainMonteCarlo`, la clase que implementa el método MCMC para variables discretas. Dicha clase tiene como campo más destacado :

   - `sample`: Una lista de casos que representa la muestra obtenida mediante este método de simulación.

   Los constructores de esta clase son:

   - `ContinuousMCMC(Bnet b, Evidence e)`: crea una nueva propagación para la red Bayesiana `b`, y la evidencia `e`.
   - `ContinuousMCMC(Bnet b, Evidence e, int s)`: crea una nueva propagación para la red Bayesiana `b`, y la evidencia `e` y una muestra de tamaño `s`.

   Los métodos implementados son aquellos que, fundamentalmente simulan los valores de las variables:

   - `propagate()`: se encarga de llevar a cabo la propagación, es decir, es el encargado de llamar a los correspondientes métodos.
   - `getInitialConfiguration()`: devuelve una configuración para todas las variables de la red, la primera configuración a partir de la que se simularán el resto de casos de la muestra.
   - `generateSample(ContinuousConfiguration initialConf)`: método que genera la muestra a partir de la primera configuración `initialConf`. Este método utiliza métodos propios de las clases `ContinuousProbabilityTree` y `MixtExpDensity` para simular valores de una MTE, que son:

   a) `ContinuousProbabilityTree`:

   - `simulateValue()`: simula un valor para una densidad univariante, tanto discreta como continua. Para ello usa los siguientes métodos de la clase `MixtExpDensity`:

   b) `MixtExpDensity`:

- `simulateGen(Continuous var,double xmin,double xmax)`: simula un valor de var entre `xmin` y `xmax` a partir de la función MTE correspondiente a esta clase. Dependiendo de si hay algún coeficiente negativo, usamos un método u otro.
- `simulate(Continuous var,double xmin,double xmax)`: simula un valor de `var` entre `xmin` y `xmax` si todos los coeficientes son negativos.
- `simulateCoefNeg(Continuous var,double xmin,double xmax)`: simula un valor de `var` entre `xmin` y `xmax` si hay algún coeficiente negativo.

Referencias

Andersen, S., K. Olesen, F. Jensen, y F. Jensen (1989). Hugin - a shell for building bayesian belief universes for expert systems. En *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, pp. 1080 – 1085. Morgan Kaufmann, San Mateo.

Bignami, A. y A. Matties (1971). A note on sampling from combinations of distributions. *J.Inst. Maths Applics 8, 80 – 81.*

Bouckaert, R., E. Castillo, y J. Gutiérrez (1996). A modified simulation scheme for inference in Bayesian networks. International Journal of Approximate Reasoning 14, 55–80.

Cano, A. y S. Moral (1995). Heuristic algorithms for the triangulation of graphs. En B. Bouchon-Meunier, R. Yager, y L. Zadeh (Eds.), *Advances in Intelligent Computing*, pp. 98–107. Springer Verlag.

Cano, A. y S. Moral (1997). Propagación exacta y aproximada con árboles de probabildad. En *Actas de la VII Conferencia de la Asociación Española para la Inteligencia Artificial*, pp. 635–644.

Cano, A., S. Moral, y A. Salmerón (2000). Penniless propagation in join trees. *International Journal of Intelligent Systems 15, 1027–1059.*

Cano, A., S. Moral, y A. Salmerón (2002). Lazy evaluation in Penniless propagation over join trees. Networks 39, 175–185.

Cano, A., S. Moral, y A. Salmerón (2003). Novel strategies to approximate probability trees in Penniless propagation. *International Journal of Intelligent Systems 18, 193–203.*

Cano, A., S. Moral, y A. Salmerón (2000). Penniless propagation in join trees. International Journal of Intelligent Systems 15, 1027 – 1059.

Cano, J., L. Hernández, y S. Moral (1996). Importance sampling algorithms for the propagation of probabilities in belief networks. *International Journal of Approximate Reasoning 15, 77–92.*

Castillo, E., A. Cobo, P. Gómez, y C. Solares (1997). JAVA. Un lenguaje de programación multiplataforma para Internet. *Paraninfo.*

Castillo, E. y J. Gutiérrez (1998). Modeling probabilistic networks of discrete and continuous variables. Journal of Multivariate Analysis 64, 48–65.

Castillo, E., J. Gutiérrez, y A. Hadi (1996). *Sistemas expertos y modelos de redes probabilísticas.* Monografías de la Academía de Ingeniería.

Christofides, A., B. Tanyi, D. Whobrey, y N. Christofides (1999). The optimal discretization of probability density functions. *Computational Statistics and Data Analysis 31, 475 – 486.*

Cobb, B. y P. P. Shenoy (2003). Inference in hibryd bayesian networks with mixtures of truncated exponentials. Working Paper no 294, School of Bussines, University of Kansas.

Consortiun, E. (2002). Elvira: An environment for probabilistic graphical models. En J. Gámez y A. Salmerón (Eds.), Proceedings of the First European Workshop on Probabilistic Graphical Models (PGM'02), *pp. 222 – 230.*

Cooper, G. (1990). The computational complexity of probabilistic inference using Bayesian belief networks. Artificial Intelligence 42, 393–405.

Cooper, G. y E. Herskovits (1992). A bayesian method for the induction of probabilistic networks from data. *Machine Learning 9, 309 – 347.*

Cover, T. y J. Thomas (1991). Elements of Information Theory. *John Wiley & Sons, Chichester, UK.*

Cowell, R., A. Dawid, S. Lauritzen, y D. Spiegelhalter (1999). Probabilistic Networks and Expert Systems. *Statistics for Engineering and Information Science. Springer.*

Dagum, P. y M. Luby (1993). Approximating probabilistic inference in Bayesian belief networks is NP-hard. Artificial Intelligence 60, 141–153.

Darroch, J., S. Lauritzen, y T. Speed (1980). Markov fields and log-linear interaction models for contingency tables. *The Annals of Statistics 8, 522–539.*

Davies, S. (2002). Fast factored density estimation and compression with bayesian networks. *Ph. D. thesis, School of Computer Science. Carnegie Mellon University.*

Davies, S. y A. Moore (2002). Interpolating conditional density trees. En Uncertainty in Artificial Intelligence: Proceedings of the Eighteenth Conference (UAI-2002)*, San Francisco, CA, pp. 119–127. Morgan Kaufmann Publishers.*

de Campos, L. (1998). Sistemas expertos probabilísticos*, Capitulo Aprendizaje automático de modelos gráficos I. Ediciones de la Universidad de Castilla-La Mancha.*

Dempster, A., N. Laird, y D. Rubin (1977). Maximum-likelihood from incomplete data via the em algorithm. J. Royal Statist. Soc. Ser.B 39.

Dougherty, J., R. Kohavi, y M. Sahami (1995). Supervised and unsupervised discretization of continous features. En A. P. y S. Russell (Ed.), *Machine Learning: Proceedings of the Twelfth International Conference.* Morgan Kaufmann, San Francisco.

El-Taha, M. y W. Evans (1992). A new estimation procedure for a right-truncated exponential distribution. En *Proceedings of the 23rd Pittsburgh Conference on Modelling and Simulation*, Volumen 23, pp. 427 – 434.

Fung, R. y K. Chang (1990). Weighting and integrating evidence for stochastic simulation in Bayesian networks. En M. Henrion, R. Shachter, L. Kanal, y J. Lemmer (Eds.), *Uncertainty in Artificial Intelligence*, Volumen 5, pp. 209–220. North-Holland (Amsterdam).

Gilks, W., S. Richardson, y D. Spiegelhalter (1996). *Markov chain Monte Carlo in practice.* Chapman and Hall.

Good, I. (1965). *The estimation of probabilities.* Cambridge: MIT Press.

Heckerman, D., D. Geiger, y D. Chickering (1995). Learning bayesian networks: The combination of knowledge and statistical data. *Machine Learning 20, 197 – 244.*

Hegde, L. M. y R. C. Dahiya (1989). Estimation of the parameters of a truncated gamma distribution. Communications in Statistics. Theory and Methods 18, 561 – 577.

Henrion, M. (1988). Propagating uncertainty by logic sampling in Bayes networks. En J. Lemmer y L. Kanal (Eds.), *Uncertainty in Artificial Intelligence*, Volumen 2, pp. 317–324. North-Holland (Amsterdam).

Hernández, L., S. Moral, y A. Salmerón (1996). Importance sampling algorithms for belief networks based on approximate computation. En *Proceedings of the Sixth International Conference IPMU'96*, Volumen II, Granada (Spain), pp. 859–864.

Hernández, L., S. Moral, y A. Salmerón (1998). A Monte Carlo algorithm for probabilistic propagation in belief networks based on importance sampling and stratified simulation techniques. *International Journal of Approximate Reasoning 18, 53–91.*

Jensen, C., A. Kong, y U. Kjærulff (1995). Blocking Gibbs sampling in very large probabilistic expert systems. International Journal of Human-Computer Studies 42, 647–666.

Jensen, F. (1996). *An introduction to Bayesian networks.* UCL Press.

Jensen, F. (2001). *Bayesian Networks and Decision Graphs.* Springer.

Jensen, F. y S. Andersen (1990). Approximations in Bayesian belief universes for knowledge-based systems. En *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence*, pp. 162–169.

Jensen, F., S. Lauritzen, y K. Olesen (1990). Bayesian updating in causal probabilistic networks by local computation. *Computational Statistics Quarterly 4, 269–282.*

Jensen, F., K. Olesen, y S. Andersen (1990). An algebra of bayesian belief universes for knowledge based systems. Networks 20, 637 – 659.

Kim, J. y J. Pearl (1983). A computational model for causal and diagnostic reasoning in inference systems. En *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, pp. 190 – 193.

Kjærulff, U. (1992). Optimal decomposition of probabilistic networks by simulated annealing. *Statistics and Computing 2, 1–21.*

Kjærulff, U. (1993). Approximation of bayesian networks through edge removals. Research Report IR-93-2007, Dept. of Mathematics and Computer Science, Aalborg University.

Kjærulff, U. (1994). Reduction of computational complexity in Bayesian networks through removal of weak dependencies. En Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence, *pp. 374–382. Morgan Kaufmann, San Francisco.*

Koller, D., U. Lerner, y D. Anguelov (1999). A general algorithm for approximate inference and its application to hybrid Bayes nets. En K. Laskey y H. Prade (Eds.), Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence, *pp. 324–333. Morgan & Kaufmann.*

Kozlov, A. y D. Koller (1997). Nonuniform dynamic discretization in hybrid networks. En Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence, *pp. 314–325.*

Kullback, S. (1978). Information theory and statistics. *Dover Publications Inc.*

Kullback, S. y R. Leibler (1951). On information and sufficiency. Annals of Mathematical Statistics 22, 76–86.

Lauritzen, S. (1992). Propagation of probabilities, means and variances in mixed graphical association models. *Journal of the American Statistical Association 87, 1098–1108.*

Lauritzen, S. (1996). Graphical models, *Volumen 17 de* Oxford Statistical Science Series. *New York: The Clarendon Press Oxford University Press. Oxford Science Publications.*

Lauritzen, S., T. Speed, y K. Vijayan (1984). Decomposable graphs and hypergraphs. Australian Mathematical Society. Journal. Series A. Pure Mathematics and

Statistics 36, 12–29.

Lauritzen, S. y D. Spiegelhalter (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B 50, 157–224.*

Lauritzen, S. y N. Wermuth (1989). Graphical models for associations between variables, some of which are qualitative and some quantitative. The Annals of Statistics 17, 31–57.

Leimer, H. (1989). Triangulated graphs with marked vertices. *Annals of Discrete Mathematics 41, 311–324.*

Madsen, A. y F. Jensen (1999). Lazy propagation: a junction tree inference algorithm based on lazy evaluation. Artificial Intelligence 113, 203–245.

Moral, S., R. Rumí, y A. Salmerón (2001). Mixtures of truncated exponentials in hybrid Bayesian networks. En *Lecture Notes in Artificial Intelligence*, Volumen 2143, pp. 135–143.

Moral, S., R. Rumí, y A. Salmerón (2002). Estimating mixtures of truncated exponentials from data. En J. Gámez y A. Salmerón (Eds.), *Proceedings of the First European Workshop on Probabilistic Graphical Models*, pp. 156–167.

Moral, S., R. Rumí, y A. Salmerón (2003). Approximating conditional mte distributions by means of mixed tress. En *Lecture Notes in Artificial Intelligence*, Volumen 2711.

Nath, G. B. (1975). Unbiased estimates of reliability for the truncated gamma distribution. *Scand. Actuar. J.*, 181 – 186.

Olesen, K. (1993). Causal probabilistic networks with both discrete and continuous variables. *IEEE Transactions on Pattern Analysis and Machine Intelligence 15, 275–279.*

Pearl, J. (1986a). A constraint propagation approach to probabilistic reasoning. Uncertainty in Artificial Intelligence, *357 – 370.*

Pearl, J. (1986b). Fusion, propagation and structuring in belief networks. Artificial Intelligence 29, 241 – 288.

Pearl, J. (1987). Evidential reasoning using stochastic simulation of causal models. *Artificial Intelligence 32, 247–257.*

Pearl, J. (1988). Probabilistic reasoning in intelligent systems. *Morgan-Kaufmann (San Mateo).*

Poole, D. (1993). Average-case analysis of a search algorithm for estimating prior and posterior probabilities in Bayesian networks with extreme probabilities. En Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93)*, pp. 606–612. Morgan Kaufmann Publishers, San Mateo.*

Redner, R. y H. Walker (1984). Mixture densities, maximum likelihood and the em algorithm. SIAM Review 26(2).

Rubinstein, R. (1981). *Simulation and the Monte Carlo Method.* Wiley (New York).

Salmerón, A., A. Cano, y S. Moral (2000). Importance sampling in bayesian networks using probability trees. *Computational Statistics and Data Analysis 34, 387–413.*

Salmerón, A. y S. Moral (2001). Importance sampling in Bayesian networks using antithetic variables. En Lecture Notes in Artificial Intelligence*, Volumen 2143, pp. 168–179.*

Santos, E. y S. Shimony (1994). Belief updating by enumerating high-probability independence-based assignments. En Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pp. 506–513.*

Sathe, Y. y S. Varde (1969). Minimum variance unbiased estimates of reliability for the truncated exponential distribution. Technometrics 11, 609 – 612.

Schmidt, T. y P. Shenoy (1997). Some improvements to the shenoy-shafer and hugin architectures for computing marginals. Working paper no.275, School of Bussines, University of Kansas.

Shachter, R. y M. Peot (1990). Simulation approaches to general probabilistic inference on belief networks. En M. Henrion, R. Shachter, L. Kanal, y J. Lemmer (Eds.), *Uncertainty in Artificial Intelligence*, Volumen 5, pp. 221–231. North Holland (Amsterdam).

Shafer, G. (1996). *Probabilistic expert systems*, Volumen 67 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM).

Shafer, G. y P. Shenoy (1988). Local computation in hypertrees. Technical Report WP-201, School of Business, University of Kansas.

Shenoy, P. (1997). Binary join trees for computing marginals in the Shenoy-Shafer architecture. *International Journal of Approximate Reasoning 17, 239–263.*

Shenoy, P. y G. Shafer (1990). Axioms for probability and belief function propagation. En R. Shachter, T. Levitt, J. Lemmer, y L. Kanal (Eds.), Uncertainty in Artificial Intelligence 4*, pp. 169–198. North Holland, Amsterdam.*

Smith, W. (1957). A note on truncation and sufficient statistics. Annals of Mathematical Statistics 28, 247 – 252.

Spirtes, P., C. Glymour, y R. Scheines (1991). An algorithm for fast recovery of sparse causal graphs. *Social Science Computing Reviews 9, 62 – 72.*

Spirtes, P., C. Glymour, y R. Scheines (1993). Causation, prediction and search. En Lecture Notes in Statistics*, Volumen 81, New York. Springer Verlag.*

Tukey, J. (1949). Sufficiency, truncation and selection. Annals of Mathematical Statistics 20, 309 – 311.

Whittaker, J. (1990). *Graphical models in applied multivariate Statistics*. Chichester: Wiley.

Wu, C. (1983). On the convergence properties of the em algorithm. *The Annals of Statistics 11(1), 95 – 103.*

Zhang, N. y D. Poole (1996). Exploiting causal independence in Bayesian network inference. Journal of Artificial Intelligence Research 5, 301–328.