UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

TRABAJO FIN DE MÁSTER

Redes Bayesianas para predicción y descubrimiento de relaciones con señales procedentes de sensores industriales

Máster Universitario en Investigación e Innovación en Inteligencia Computacional y Sistemas Interactivos

Autor: Pablo Ramírez Hereza

Tutor: Daniel Ramos Castro

FECHA: SEPTIEMBRE 2020

TRABAJO DE FIN DE MÁSTER

REDES BAYESIANAS PARA PREDICCIÓN Y DESCUBRIMIENTO DE RELACIONES CON SEÑALES PROCEDENTES DE SENSORES INDUSTRIALES

AUTOR: Pablo Ramírez Hereza
DIRECTOR: Daniel Ramos Castro

AUDIAS
Dpto. de Tecnología electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
SEPTIEMBRE 2020

Resumen

Este Trabajo de Fin de Máster surge con la colaboración del grupo AUDIAS, de la Escuela Politécnica Superior (UAM), y una empresa proveedora de centrales eléctricas. Los principales objetivos del proyecto son, por un lado, la adquisición de un mayor conocimiento de las relaciones existentes entre las variables industriales implicadas en el proceso de generación de electricidad y, por otro lado, la predicción de determinadas variables no observables, y de mayor importancia, a lo largo del proceso en cuestión. Para la resolución de ambos objetivos, y partiendo de un trabajo previo realizado a lo largo de 2019, este trabajo se divide en dos tareas claramente diferenciadas.

En primer lugar, con el objetivo de realizar el descubrimiento de relaciones entre las variables involucradas, se propone la utilización de técnicas de aprendizaje estructural de redes bayesianas gaussianas. Para ello, se han evaluado las principales técnicas basadas en puntuación, como son *K2*, *Greedy Hill Climbing* y *GES*. Esta tarea finaliza con la generación de estructuras alternativas a las definidas por conocimiento experto.

En segundo lugar, este trabajo propone la utilización de un modelo completamente bayesiano gaussiano con el objetivo de obtener un modelo más robusto a la escasez de datos que el desarrollado en 2019. Para ello, se realiza un estudio sobre el impacto de la introducción de una distribución a priori *Normal-Wishart* en la predicción final de las variables no observables.

De esta forma, se consigue una transferencia de conocimiento a la empresa, cumpliendo finalmente con los objetivos iniciales del proyecto.

Palabras clave

Redes Bayesianas, gaussianización, aprendizaje estructural, modelo completamente bayesiano, distribuciones a priori, distribuciones a posteriori

Abstract

This Final Master in Science Thesis arises with the collaboration between the group AUDIAS, of the "Escuela Politécnica Superior" (UAM), and a provider power plant company. The main objectives of the project are, on the one hand, the acquistion of a greater knowledge of the existing relationships between the industrial variables involved in the electricity generation process and, on the other hand, the prediction of some unobservable variables, of greater importance, throughout the process. For the resolution of both objectives, and based on the previous work carried out throughout 2019, this work is divided into two clearly differentiated tasks.

Firstly, with the aim of discovering relationships between the variables, the use of Bayesian Gaussian networks structure learning techniques is proposed. To do this, the main scoring-based algorithms have been studied and evaluated, such as *K2*, *Greedy Hill Climbing* and *GES*. This task finalizes with the development of alternative structures to those defined by expert knowledge.

Secondly, this work proposes the use of a fully Bayesian Gaussian model with the aim of obtaining a more robust, model to the scarcity of data, than the one developed in 2019. For this, a study is carried out on the impact of the introduction of a Normal-Wishart prior on the final prediction of the unobservable variables.

Then, this work involves a transfer of knowledge to the company, finally fulfilling the initial objectives of the project.

Key words

Bayesian Networks, Gaussianization, structure learning, fully Bayesian model, prior, posterior

Agradecimientos

Tras la finalización de este trabajo, solo queda agradecer a todos aquellos que han hecho esto posible.

Primero, agradecer a la Escuela Politécnica Superior, y a todo el personal docente, por formarme como ingeniero de telecomunicaciones, investigador y persona. En concreto, agradecer al grupo de investigación AUDIAS por abrirme sus puertas y permitirme formar parte de un equipo como este. Mención de honor a mi tutor Daniel Ramos, por ser el gran maestro exigente y cercano que cualquier padawan necesita.

Por otra parte, agradecer a mi familia, tanto los que están como los que no, por depositar toda su confianza en mí, incluso cuando yo no lo hago. Sin vuestro apoyo hoy no estaría aquí. A mis amigos, por ayudarme a escapar del mundo en los momentos que más necesito.

Por supuesto, agradecer a mi novia y mejor amiga por el despliegue de paciencia (mucha), amor, locura y apoyo que me dedica día a día durante ya más de 6 años. A su familia, por abrirme en todo momento las puertas de su casa y hacerme sentir uno más.

Por último, a mi abuelo Juan, seguimos pensando en ti en todo momento.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

1.1. Motivación

Este trabajo de fin de máster se enmarca en la predicción de variables de interés, a partir de señales temporales de diferente naturaleza, mediante el desarrollo e implementación de modelos probabilísticos paramétricos.

El contexto en el que se desarrolla este trabajo es la colaboración del grupo AUDIAS de la Escuela Politécnica Superior (EPS), con una empresa proveedora de centrales eléctricas. Los objetivos principales de dicha colaboración son, por un lado, entender mejor las variables industriales que intervienen en el proceso de generación de electricidad y, por otro lado, realizar la predicción de determinadas variables de interés, denominadas variables *Medida*, a partir de un conjunto de variables observables, denominadas variables *Control*. Para ello, se dispone de una base de datos proporcionada por la empresa en cuestión.

Tras la realización de una serie de experimentos previos, se demuestra que la utilización de redes bayesianas gaussianas y redes bayesianas gaussianas dinámicas permite una predicción óptima de las variables de interés. Estas pruebas son recopiladas en formato de trabajo de fin de máster, *Algoritmos probabilísticos automáticos para predicción a partir de señales de centrales nucleares* (2019), por el mismo autor que el presente documento.

Como continuación del trabajo realizado anteriormente, la idea que subyace en este TFM es, en primer lugar, realizar el descubrimiento de relaciones entre las diferentes variables disponibles. El objetivo principal de esta tarea es obtener un modelo que represente mejor los datos que las redes implementadas con anterioridad, cuyas estructuras fueron definidas por el conocimiento experto de la empresa.

Para ello, se pretende estudiar y utilizar técnicas de aprendizaje estructural de redes bayesianas, como son los algoritmo *K2*, *GHC* y *GES*. El resultado esperado de esta tarea consiste en la obtención de modelos cuyas estructuras, a diferencia del *baseline*, sean aprendidas a partir de los datos. De esta manera, se pretende aportar un mayor conocimiento de las relaciones de independencia condicional existentes entre las variables.

En segundo lugar, este trabajo pretende evidenciar que los modelos completamente bayesianos o *fully bayesian* son útiles a la hora de incorporar la incertidumbre del problema y, por lo tanto, aportar mejores predicciones en situaciones en las que exista un número muy limitado de datos de entrenamiento.

Para ello, se propone la implementación de un modelo *fully bayesian* de las variables presentes en la base de datos, y su comparativa con modelos entrenados con estimadores de máxima verosimilitud (MLE) y máximo a posteriori (MAP). Como por ejemplo, el sistema *baseline*, cuyo entrenamiento fue realizado mediante MLE. En concreto, este trabajo se centra en el estudio e implementación del tratamiento completamente bayesiano de un modelo basado en una distribución gaussiana multivariada de los variables procedentes de la empresa, tras una transformación gaussianizante previa de las mismas.

De esta forma, el impacto científico de este proyecto es alto pues, hasta donde alcanza nuestro conocimimento, no existe ninguna aplicación de métodos probabilísticos gráficos de aprendizaje automático, y mucho menos métodos basados en un enfoque completamente bayesiano, que utilicen datos de esta naturaleza. Por otro lado, se espera que los resultados de este trabajo sean útiles para la mejora de los procedimientos de la empresa, lo cual implicará una alta innovación tecnológica en el ámbito del análisis estadístico y de la ciencia de datos en el contexto de esta aplicación.

1.2. Objetivos

El objetivo principal de este trabajo es doble. Por un lado, se pretende afrontar el desarrollo de un modelo *fully bayesian* para la predicción de variables de interés, a partir de un conjunto de variables conocidas. Por otra parte, se pretenden aprender estructuras alternativas a la que forma el *baseline* implementado anteriormente.

Para ello, se definen una serie de objetivos parciales:

1. Estudio teórico del impacto de la introducción de conocimiento *prior* en la distribución predictiva del modelo.
2. Aplicación de técnicas de procesado de señal para una mejor representación de los datos disponibles por el modelo.
3. Implementación del modelo *fully-bayesian* para las bases de datos disponibles e inferencia de las variables a predecir.
4. Comparativa de los resultados obtenidos con aquellos obtenidos por un modelo paramétrico entrenado mediante MLE o MAP.
5. Estudio teórico de algoritmos de aprendizaje estructural de redes bayesianas con variables gaussianas.
6. Generación de estructuras alternativas a la que forma el *baseline* implementado anteriormente.

1.3. Organización de la memoria

Esta memoria ha sido dividida de tal forma que el lector siga el flujo de trabajo del proyecto, entendiendo los conceptos clave en los que éste se basa. Para ello, este documento se encuentra dividido en varios capítulos. En el primero, se presenta una introducción del trabajo y sus principales objetivos. El segundo capítulo repasa los conceptos teóricos clave para el entendimiento de las soluciones a ambas tareas de este trabajo. El tercero, ofrece una descripción de la base de datos y del método de evaluación de ambas tareas. Por último, se presentan los resultados obtenidos y las principales conclusiones extraídas.

Es importante recalcar que, pese a que las tareas realizadas son independientes del trabajo definido en [1] por el mismo autor, el contexto en el que se desarrollan ambos trabajos es el mismo, por lo que, con el objetivo de que este documento sea autocontenido, se ha decidido mantener determinadas secciones de [1], relativas al estado del arte y al contexto del trabajo.

De esta forma las secciones 2.1 y 2.2, así como los anexos A y B, han sido directamente extraídos del documento en cuestión.

Capítulo 2

Estado del arte

En el siguiente capítulo se repasan los conceptos teóricos necesarios para la comprensión del trabajo realizado. Adicionalmente, en los anexos A y B se proporcionan conceptos básicos sobre la teoría de probabilidad y el proceso de gaussianización basado en ecualización de histogramas. En caso de que el lector no esté familiarizado con dichos conceptos, se recomienda su lectura.

2.1. Modelos gráficos probabilísticos

Se definen los modelos probabilísticos como aquellos que hacen uso de la teoría de la probabilidad (ver Anexo A) para cuantificar la incertidumbre existente en un determinado problema. Aunque la resolución de problemas probabilísticos complejos puede ser realizada de forma analítica, el uso de representaciones gráficas de las distribuciones de probabilidad es de gran utilidad dado que [2, 3]:

- Representan de una forma sencilla y altamente interpretable la estructura de un modelo probabilístico. De forma que, las propiedades del modelo, y en concreto, las propiedades de independencia condicional, pueden ser extraídas directamente a partir del grafo.
- Permiten la utilización de la teoría de grafos para la resolución de los cálculos complejos necesarios para el aprendizaje e inferencia del modelo.
- Permiten representar las relaciones entre las diferentes variables involucradas en el modelo independientemente de las distribuciones de probabilidad de dichas variables.

De esta forma, un modelo gráfico probabilístico es una representación de la distribución conjunta de las variables aleatorias del modelo. Dicha represetanción esta formada por *nodos*, que representan las variables, y *enlaces*, que expresan las relaciones probabilísticas entre dichas variables.

En función de dichos enlaces encontramos dos principales grupos de modelos gráficos probabilísticos:

1. **Modelos gráficos dirigidos**: Son los grafos en los cuales los enlaces se encuentran dirigidos de una variable a otra, indicando, de esta forma, la existencia de dependencia entre ambas. Los modelos más comunes, y objeto de este trabajo, son los grafos que carecen de caminos cerrados en su estructura, los grafos dirigidos acíclicos, *DAGs* de sus siglas en inglés (*Directed Acyclic graphs*) o redes bayesianas.

2. **Modelos gráficos no dirigidos** o también conocidos como *Markov Random Fields*. Como se verá más adelante, las redes bayesianas corresponden a un tipo especial de factorización de una distribución de probabilidad conjunta, en la cual cada factor es una distribución por si mismo, mientras que los *Markov Random Fields* representan una factorización alternativa, basada en factores o potenciales.

2.2. redes bayesianas

Como se ha visto anteriormente, las redes bayesianas son modelos gráficos probabilísticos dirigidos, en concreto, DAGs.

En un grafo dirigido, definimos una relación padre/hijo cuando existe un enlace entre dos variables. De esta forma, en la red bayesiana presentada en la figura 2.1, podemos decir que X1 es el nodo padre de X2, que X2 es padre de X3 y, a su vez, que X3 es hijo de X2 y X2 hijo de X1.

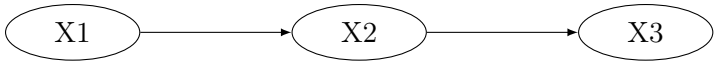


Figura 2.1: Ejemplo red bayesiana

Si aplicamos la regla de la cadena (A.3) para un problema genérico de 3 variables X1, X2 y X3, tenemos que:

$$\begin{aligned} P(X_3, X_2, X_1) &= P(X_3|X_2, X_1)P(X_2, X_1) \\ &= P(X_3|X_2, X_1)P(X_2|X_1)P(X_1) \end{aligned} \tag{2.1}$$

Ahora bien, asumiendo que la red anterior define el conjunto de independencias condicionales del problema, obtenemos directamente del grafo una simplifación de la expresión anterior:

$$(X_3X_1)|X_2 \rightarrow P(X_3, X_2, X_1) = P(X_3|X_2)P(X_2|X_1)P(X_1) \tag{2.2}$$

de esta forma, una red bayesiana puede ser parametrizada a partir de todas las distribuciones de probabilidad condicionales o CPD (*Conditional Probability Distributions*) $P(X_i|Pa_i)$, donde X_i representa el nodo i y Pa_i sus nodos padre. De esta forma, generalizando la ecuación 2.2, se define la relación entre un grafo y la función de probabilidad conjunta de un problema mediante la *Regla de la Cadena* del grafo:

La distribución conjunta definida por un grafo es dada por el producto, para todas sus variables, de las probabilidades condicionales de cada nodo con respecto a sus padres (CPDs). Teniendo, para un grafo de K nodos:

$$p(x) = \prod_{k=1}^{K} p(x_k|pa_k) \tag{2.3}$$

Esta expresión define la propiedad de *factorización* de la probabilidad conjunta de un problema, dado el conjunto de independencias condicionales definidas por una red bayesiana.

Una vez introducidas las redes bayesianas, en las siguientes secciones se realiza una introducción a la inferencia y aprendizaje en éstas, centrándose en redes bayesianas gaussianas, en las cuales, los nodos representan variables aleatorias continuas con una distribución gaussiana univariada.

2.2.1. Inferencia en redes bayesianas

Se denomina *Inferencia* al proceso de cálculo de la probabilidad una vez conocidos los valores que toman otras variables de la red, es decir, cuando se introduce una determinada *evidencia*.

Así pues, sea O el conjunto total de variables en la red, Q el conjunto de *variables observadas*, de las cuales se introduce una evidencia ($Q = \mathbf{q}$), R el conjunto de variables *no observadas o latentes* dentro del cual, se encuentra el subconjunto S que incluye las variables a inferir y T las variables carentes de interés. Cumpliéndose: $O = Q \cup R$ y $R = S \cup T$.

Podemos definir el proceso de inferencia como el cálculo de:

$$\begin{aligned} P(S|Q = \mathbf{q}) &= \frac{P(S, Q = \mathbf{q})}{P(Q)} \\ &= \frac{\sum_T P(S, Q = \mathbf{q}, T = t_i)}{P(Q)} \\ &= \frac{\sum_T P(S, Q = \mathbf{q}, T = t_i)}{\sum_R P(Q, R = r_j)} \end{aligned} \tag{2.4}$$

Así pues, como se puede ver en 2.11, el proceso de inferencia requiere el cálculo de dos marginalizaciones, lo cual, para modelos complejos, supone un alto coste computacional. Para la realización de la inferencia de forma eficiente existen numerosos algoritmos que se pueden dividir en:

1. **Inferencia exacta**: Los casos en los cuales se puede hacer una inferencia exacta son limitados, en especial, redes con todos los nodos latentes discretos o con distribuciones gaussianas univariadas. En este último caso, la inferencia se puede realizar algebráicamente al ser la red una parametrización de una distribución *gaussiana multivariada conjunta*. Los algoritmos utilizados en este caso se basan en "empujar sumas" en el cálculo de la inferencia, como por ejemplo *Variable Elimination* [4], o en el paso de mensajes a través de árboles, como el algoritmo *Junction Tree* [5] [6].

2. **Inferencia aproximada**: Incluso en los casos en los que la inferencia exacta es posible, puede ser computacionalmente demasiado lenta debido a la complejidad de la estructura de la red o a la complejidad de las distribuciones continuas de los nodos. Se utilizan técnicas de inferencia por *Monte-Carlo*, *Loopy Belief Propagation*, *inferencia variacional*, etc. En este documento no se realizará un estudio de este tipo de técnicas.

2.2.1.1. Inferencia en redes bayesianas Gaussianas

Tal y como se ve en [7], debido a las características estadísticas de la función normal, la inferencia, en este caso, se puede calcular matricialmente de una forma analítica:

Sea X el conjunto de N variables aleatorias del modelo, de las cuales q son latentes, gaussianas multivariadas a inferir. Podemos particionar μ y Σ tal que:

$$\begin{aligned} X &= \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \text{de tamaño} \begin{bmatrix} q \times 1 \\ (N-q) \times 1 \end{bmatrix} \\ \mu &= \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \text{de tamaño} \begin{bmatrix} q \times 1 \\ (N-q) \times 1 \end{bmatrix} \\ \Sigma &= \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \text{de tamaño} \begin{bmatrix} q \times q & q \times (N-q) \\ (N-q) \times q & (N-q) \times (N-q) \end{bmatrix} \end{aligned} \tag{2.5}$$

Entonces, dada la evidencia a, $P(x_1|x_2 = a)$ es una multivariada definida por:

$$\begin{aligned} \bar{\mu} &= \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(a - \mu_2) \\ \bar{\Sigma} &= \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} \end{aligned} \tag{2.6}$$

2.2.1.2. *Variable Elimination*

Como define [5], debido a la estructura de la red bayesiana, varias subexpresiones dentro de la expresión de la probabilidad conjunta (ver expresión 2.2) dependen de pocas variables. Es por esto que, calculando dichas expresiones una vez, y guardando los resultados, se puede evitar recalcularlas un número exponencial de veces.

Así pues, sea la red bayesiana descrita por:

$$P(A, B, C, D) = P(A)P(B|A)P(C|B)P(D|C) \tag{2.7}$$

La probabilidad marginal $P(D)$ se calcula, en el caso en el que las distribuciones de los nodos sea continua (nótese que si fuese discreta se reemplazarían las integrales por sumatorios):

$$P(D) = \int_A \int_B \int_C P(A)P(B|A)P(C|B)P(D|C) \tag{2.8}$$

Empujando las integrales, clave del algoritmo *Variable Elimination*, obtenemos:

$$\begin{aligned} P(D) &= \int_C P(D|C) \int_B P(C|B) \int_A P(A)P(B|A) \\ &= \int_C P(D|C) \int_B P(C|B)\tau(B) \\ &= \int_C P(D|C)\tau(C) \end{aligned} \tag{2.9}$$

Conllevando de esta forma una reducción en el coste computacional en el cálculo de la probabilidad marginal, clave para la inferencia. El algoritmo *Variable Elimination* se puede considerar como un algorítmo de paso de mensaje, en el cual, tras eliminar una variable marginalizándola, se envía el mensaje $\tau(.)$ al siguiente nodo a eliminar.

2.2.2. Aprendizaje en redes bayesianas

En una red bayesiana distinguimos dos tipos de aprendizaje: Aprendizaje de parámetros o *model fitting* y aprendizaje de la estructura también llamado *model selection*. Este trabajo aborda el aprendizaje estructural de redes bayesianas en la sección 2.5 y, por otro lado, en las secciones 2.3 y 2.4 se aborda el aprendizaje de parámetros en modelos probabilísticos paramétricos.

En ambos casos, el conjunto de datos disponible para el entrenamiento es considerado completamente observado, por lo que no se hará uso de técnicas como *Expectation-Maximization*[15] en ningún proceso de aprendizaje.

2.3. Aprendizaje de parámetros

En esta sección se describen las principales técnicas de aprendizaje de parámetros en un modelo probabilístico a partir de un conjunto de datos completo. En concreto, este trabajo se centra en el aprendizaje de parámetros en un modelo basado en una distribución gaussiana multivariada.

Los principales estimadores utilizados para definir los parámetros de un modelo, o de una red bayesiana, son el estimador de máxima verosimilitud y el estimador de máximo a posteriori.

El estimador de máxima verosimilitud (*Maximum Likelihood Estimator* o *MLE*) define los valores de los parámetros del modelo como aquellos que maximizan el logaritmo de la función de verosimilitud. Por otro lado, el estimador de máximo a posteriori (*Maximum a Posteriori o MAP*), define los valores de los parámetros como aquellos que maximizan el logaritmo de la distribución a posteriori de los parámetros.

2.3.1. Estimador de Máxima Verosimilitud

Sea $X = \{x_1, ..., x_M\}$ un conjunto de observaciones independientes de una misma variable, con una función de distribución de la que únicamente se sabe la familia a la que pertenece, y θ el conjunto de parámetros que la definen. El algoritmo *MLE* pretende asignar un valor a los parámetros θ^{MLE} que maximice la verosimilitud de los datos con el modelo. Es decir, busca un modelo que sea consistente con los datos, desde el punto de vista de su verosimilitud.

Para ello, se siguen los siguientes pasos:

1. Se parte de la función de densidad de probabilidad fijando los valores observados como fijos. Esta función es llamada *función de verosilimitud*:

$$l(\theta, x_1, x_2, ..., x_n) = \prod_{i=1}^{n} f(x_i|\theta) \tag{2.10}$$

2. Se calcula el logaritmo:

$$L(\theta, x_1, x_2, ..., x_n) = \sum_{i=1}^{n} \log(f(x_i|\theta)) \tag{2.11}$$

3. Se maximiza la función 2.11

$$\theta^{MLE} = \arg\max_{\theta} L(\theta, x_1, x_2, ..., x_M) \tag{2.12}$$

Dichos parámetros pueden ser obtenidos analíticamente a partir de los datos o, por lo contrario, necesitar aproximaciones.

Máxima Verosimilitud en distribuciones gaussianas multivariadas

Como ya ha sido mencionado anteriormente, en el caso en el que todas las variables del modelo presenten una distribución gaussiana univariada, la red es una representación de una distribución gaussiana multivariada, cuyos parámetros que la definen son un vector de medias, μ, y una matriz de covarianzas Σ, tal que:

$$f(x) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right) \tag{2.13}$$

Siguiendo el desarrollo previamente definido, tal como se describe en [8], obtenemos los valores de la matriz de covarianza Σ y μ, son:

$$\begin{aligned} \hat{\mu} &= \frac{1}{n}\sum_{i=1}^{n} x_i \\ \hat{\Sigma} &= \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{\mu})(x_i - \hat{\mu})^T \end{aligned} \tag{2.14}$$

Como se puede observar en las ecuaciones definidas en 2.14, los valores de los parámetros que definen la distribución gaussiana multivariada se pueden extraer directamente a partir de los datos de entrenamiento.

2.3.2. Selección de probabilidades a priori (*priors*)

Mientras que el estimador *MLE* maximiza la verosimilitud del modelo con los datos, $P(D|\theta)$, el estimador de máximo posteriori selecciona como parámetros del modelo aquellos que maximizan la distribución a posteriori o *posterior* de los parámetros, o lo que es equivalente, maximiza la expresión $P(D|\theta)P(\theta)$. Para ello es necesario la definición de las distribuciones a priori o *prior* de los parámetros, $P(\theta)$.

La selección de la distribución a priori de los parámetros se realiza frecuentemente

en base a una conveniencia matemática, en vez de realizarse en base a un conocimiento previo a la observación del conjunto de datos.

De esta forma, este trabajo se centrará en la utilización de *prior conjugados*, es decir, distribuciones a priori que, al multiplicarse con el *likelihood*, dan lugar a distribuciones *posterior* con la misma forma funcional que el *prior* en cuestión. Simplificando, de esta forma, el análisis bayesiano que se realizará en secciones posteriores.

Selección de prior para distribuciones gaussianas multivariadas

El documento [8] expone un análisis detallado de los diferentes *prior conjugados* utilizados para los parámetros de una gaussiana multivariada. Este documento sirve de guía para el desarrollo del estimador de MAP y el tratamiento completamente bayesiano del modelo gaussiano multivariado.

Los parámetros que definen una gaussiana multivariada son, como hemos visto anteriormente, el vector de medias (μ), y la matriz de covarianza (Σ) o la matriz de precisión (Λ). Por mayor simplicidad en los cálculos, se decide en este trabajo utilizar la matriz de precisión. Por lo tanto, el posterior de los parámetros se puede expresar tal que:

$$\begin{aligned} P(\mu, \Lambda|\mathcal{D}) &\propto P(\mathcal{D}|\mu, \Lambda)P(\mu, \Lambda) \\ &= P(\mathcal{D}|\mu, \Lambda)P(\mu|\Lambda)P(\Lambda) \end{aligned} \tag{2.15}$$

En [9] se demuestra que una distribución *Normal-Wishart* es una distribución *prior conjugada* de un *likelihood* gaussiano. De esta forma, la distribución *prior* conjugada de la matriz de precisión, $P(\Lambda)$, es una distribución Wishart, y la distribución *prior* del vector de medias dada la precisión, $P(\mu|\Lambda)$, corresponde a una distribución normal multivariada.

Pese a que un estudio más teórico y matemático de dicho *prior* queda fuera del alcance de este trabajo (para una mayor profundidad, se recomienda al lector la lectura de [9] y [2]), se realiza una introducción sobre ambas distribuciones y los *hiperparámetros* que la definen. Por un lado, la distribución Wishart es una distribución definida sobre matrices, tal que:

$$\begin{aligned} Wi_v(X|S) &= \frac{1}{Z}|X|^{\frac{v_0-p-1}{2}}\exp[-\frac{1}{2}\mathrm{tr}(T_0^{-1}X)] \\ Z &= 2^{v_0 p/2}\Gamma_p(v/2)|T_0|^{v/2} \end{aligned} \tag{2.16}$$

donde X es una matriz simétrica de dimensión p, y $\Gamma_p(\alpha)$ es la función gamma generalizada. Los *hiperparámetros* que definen esta distribución son el grado de libertad v_0 y la matriz de escala T_0.

Por otro lado, la distribución $P(\mu|\Lambda)$ corresponde a una gaussiana multivariada cuya matriz de precisión es una función lineal de Λ, definida por la distribución anterior.

$$\begin{aligned} P(\mu|\Lambda) &= \mathcal{N}(\mu|\mu_0, (k_0\Lambda)) \\ &= \frac{|k_0\Lambda|^{1/2}}{\sqrt{(2\pi)^n}} \exp\left(-\frac{1}{2}(\mu-\mu_0)^T(k_0\Lambda)(\mu-\mu_0)\right) \end{aligned} \tag{2.17}$$

Siendo, tal como se puede ver en la ecuación 2.17, μ_0 la media del parámetro μ, y k_0 el coeficiente de linealidad de la matriz de precisión, sus dos *hiperparámetros*.

De esta forma, la distribución a priori de los parámetros quedará definida con la siguiente notación:

$$P(\mu, \Lambda) = NWi(\mu, \Lambda|\mu_0, k_0, v_0, T_0)$$

2.3.3. Técnica de Máximo a Posteriori

Como su nombre indica, el estimador de *Maximum a Posteriori*, define como parámetros del modelo aquellos que maximizan la distribución *posterior*, y no la función de verosimilitud.

Así pues, sea $X = \{x_1, ..., x_n\}$ un conjunto de observaciones independientes de una misma variable, con una función de distribución, de la que únicamente se sabe la familia a la que pertenece, siendo θ el conjunto de parámetros que la definen y $P(\theta)$ el conocimiento a priori que tenemos sobre los parámetros del modelo. El algoritmo MAP pretende asignar un valor a los parámetros θ^{MAP} que maximice el *posterior* de los datos con el modelo.

Para ello, se siguen los siguientes pasos:

1. Partiendo del teorema de Bayes, se puede obtener el posterior de los parámetros tal que:

$$\begin{aligned} P(\theta|X) &= \frac{P(X|\theta)P(\theta)}{P(X)} \\ &\propto P(X|\theta)P(\theta) \end{aligned} \tag{2.18}$$

Nótese que, al considerarse $P(X)$ un factor de proporcionalidad, éste puede omitirse al realizar el proceso de maximización.

2. Se calcula el logaritmo:

$$\begin{aligned} l(\theta|X) &= \log P(X|\theta) + \log P(\theta) \\ &= \sum_{i=1}^{n} \log(f(x_i|\theta)) + \log P(\theta) \end{aligned} \tag{2.19}$$

3. Se maximiza la función anterior:

$$\theta^{MAP} = \arg\max_{\theta} l(\theta|X) \tag{2.20}$$

Como se puede comprobar en la expresión 2.19, cuanto mayor es el tamaño del conjunto de datos menor peso tiene el *prior*. Hasta que $n \to \infty$, caso en el cual se cumple que $\theta_{MLE} = \theta_{MAP}$

Máximo a Posteriori en distribuciones Gaussianas Multivariadas

Tal como se definió anteriormente, en este trabajo nos centraremos en el uso de una distribución normal-Wishart como distribución a priori de los parámetros μ y Λ, para una verosimilitud gaussiana multivariada de dimensión d.

De esta forma, siguiendo los desarrollos matemáticos de [9], tenemos que el *posterior* corresponde también a una distribución normal-Wishart, definida, para un conjunto de entrenamiento de tamaño n, tal que:

$$P(\mu, \Lambda) = N(\mu|\mu_n, (k_n\Lambda)^{-1}) Wi_{v_n}(\Lambda|T_n) \tag{2.21}$$

donde los parámetros que definen la función, son calculados:

$$\begin{aligned} \mu_n &= \frac{k\mu_0 + n\bar{x}}{k+n} \\ T_n &= T + S + \frac{kn}{k+n}(\mu_0 - \bar{x})(\mu_0 - \bar{x})^T \\ S &= \sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T \\ v_n &= v + n \\ k_n &= k + n \end{aligned} \tag{2.22}$$

Por lo tanto, siguiendo [9][8], se obtienen las expresiones finales de los parámetros en función de los *hiperparámetros*:

$$\begin{aligned}\theta^{MAP} &= \arg\max_{\theta} \left[P(D|\mu,\Lambda)NWi(\mu,\Lambda)\right] \\ \mu_{MAP} &= \sum_{i=1}^{n} x_i + k_0\mu_0 N + k_0 \\ \Sigma_{MAP} &= \frac{S + k_0(\mu_{MAP}-\mu_0)(\mu_{MAP}-\mu_0)^T + T_0^{-1}}{N + v_0 - d}\end{aligned} \tag{2.23}$$

2.4. Modelo completamente bayesiano

2.4.1. Introducción

A diferencia del punto de vista frecuentista, en el cual se define la probabilidad como frecuencia de repetición de un evento, el punto de vista bayesiano se basa en la utilización de la teoría de la probabilidad como marco formal para la cuantificación de la incertidumbre asociada a un problema. Como, por ejemplo, la incertidumbre asociada a la selección de los parámetros que definen un modelo, o la incertidumbre asociada a la propia selección de un modelo.

Mientras que desde un punto de vista frecuentista o clásico de la probabilidad los parámetros que definen un modelo se consideran parámetros fijos, cuyo valor es determinado por un estimador, desde un punto de vista completamente bayesiano, se realiza la integración o marginalización sobre todos los posibles valores de los parámetros que definen el modelo.

De esta forma, adoptando un punto de vista completamente bayesiano, se define la distribución *posterior predictiva* como:

$$P(x|\mathcal{D}) = \int P(x|\theta)P(\theta|D)d\theta \tag{2.24}$$

Donde $P(\theta|D)$ es la distribución a posteriori de los parámetros (ecuación 2.18), y $P(x|\theta)$ es el modelo paramétrico de los datos. Remarcar la diferencia existente con las aproximaciones basadas en estimadores, en las cuales la probabilidad de un dato x vendría determinado por el modelo definido con los parámetros estimados, $P(x|\theta^{MLE})$ o $P(x|\theta^{MAP})$.

La principal ventaja del uso de aproximaciones completamente bayesianas frente a aproximaciones basadas en estimadores reside en la robustez frente a la escasez de

datos. La incorporación de la incertidumbre en los parámetros reduce la probabilidad de encontrar valores de densidad predictiva muy cercanos a cero, con lo que el modelo gana estabilidad.

Por otro lado, la principal desventaja de estos modelos reside en el alto coste computacional necesario para el cálculo de la marginalización (integración) sobre todo el espacio de parámetros. Sin embargo, el desarrollo de métodos de muestreo, como el *Markov Chain Monte Carlo*, o esquemas de aproximación deterministas como *Variational Bayes* o *Expectation Propagation*, junto con las grandes mejoras en cuanto a velocidad y memoria de los ordenadores, han provocado el crecimiento del interés y del uso de métodos bayesianos en tareas de *machine learning*.

Tratamiento fully bayesian de un modelo gaussiano

Este trabajo se centra en el tratamiento completamente bayesiano de un modelo gaussiano. Para ello, al igual que se ha descrito en 2.3.2, se hace uso de una distribución a priori normal-Wishart de los parámetros. Esta sección se centra en la definición de la distribución predictiva obtenida en el proceso. Para ello, a modo de resumen, se recopilan las distribuciones involucradas en el cálculo de la distribución predictiva:

1. Verosimilitud o *likelihood* gaussiano multivariado:

$$p(D|\mu, \Lambda) = \prod_{i=1}^{n} \mathcal{N}(x_i|\mu, \Lambda) \tag{2.25}$$

2. Distribución a priori o *prior* de los parámetros:

$$p(\mu, \Lambda) = \mathcal{N}Wi(\mu, \Lambda|\mu_0, k_0, v_0, T_0) \tag{2.26}$$

   Descrito en la sección 2.3.2. Los hiperparámetros μ_0, k_0, v_0 y T_0 son seleccionados por el usuario.

3. Distribución a posteriori o *posterior* de los parámetros:

$$p(\mu, \Lambda) = \mathcal{N}Wi(\mu, \Lambda|\mu_n, k_n, v_n, T_n) \tag{2.27}$$

   Descrito en la sección 2.3.3. Los parámetros μ_n, k_n, v_n y T_n son calculados a partir de los hiperparámetros y las muestras presentes en $\mathcal{D}$.

4. Modelo gaussiano multivariado de las variables:

$$P(D|\mu,\Lambda) = \mathcal{N}(x|\mu,\Lambda) \tag{2.28}$$

La utilización de una distribución a priori de los parámetros conjugada, permite que el posterior tenga su misma forma funcional y que, en este caso, el cálculo de 2.29 tenga una solución cerrada.

$$P(x|\mathcal{D}) = \int \mathcal{N}(x|\mu,\Lambda) NWi(\mu,\Lambda|\mu_n,k_n,v_n,T_n)d\theta \tag{2.29}$$

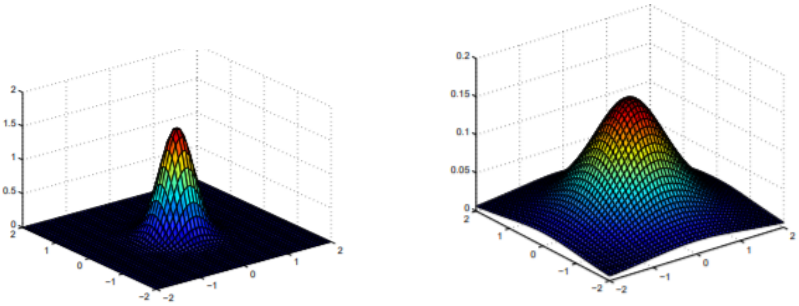
La solución corresponde a una distribución t de Student multivariada. Esta distribución es dada por la expresión:

$$t_v(x|\mu,\Sigma) = \frac{\Gamma(v/2+d/2)}{\Gamma(v/2)} \frac{|\Sigma|^{-1/2}}{v^{d/2}\pi^d} \times [1+\frac{1}{v}(x-\mu)^T\Sigma^{-1}(x-\mu)]^{-(\frac{v+d}{2})} \tag{2.30}$$

donde x es un vector d-dimensional, y el conjunto de parámetros que definen la distribución son, el vector de medias, μ, la matriz de escala, Σ y el grado de libertad o v.

Analizando las características de la distribución (2.31), se observa que tiene un comportamiento similar al definido por una distribución gaussiana multivariada, con la excepción de que presenta unas colas más largas.

$$\begin{aligned} E[x] &= \mu \ \text{ si } v > 1 \\ mode[x] &= \mu \\ cov[x] &= \frac{v}{v-2}\Sigma \ \text{ si } v > 2 \end{aligned} \tag{2.31}$$



(a) (b)

Figura 2.2: Gaussiana multivariada (a) y t-Student multivariada con $v = 2$

En el límite $v \to$ ínf la distribución t-Student multivariada tiende a una gaussiana multivariada de media μ y matriz de covarianza Σ.

Así pues, la distribución predictiva resultante de la ecuación 2.29 es [8]:

$$p(x|\mathcal{D}) = t_{v_n-d+1}(\mu_n, \frac{T_n(k_n+1)}{k_n(v_n-d+1)}) \tag{2.32}$$

A partir de la expresión 2.32 y del valor del parámetro v_n, que crece cuanto mayor sea el conjunto de entrenamiento, se deduce que, a medida que el conjunto de entrenamiento crece la distribución t de Student tiende a una gaussiana multivariada.

Para finalizar, de una manera similar a la gaussiana multivariada, se puede calcular algebraicamente las distribuciones de probabilidad condicional y marginal de la distribución t de Student multivariada. De forma que, sea $p(x)$ una t-Student multivariada tal que:

$$p(x) = t_v(\mu, \Sigma) \tag{2.33}$$

donde μ y Σ pueden particionarse tal que:

$$\begin{aligned} X &= \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \text{de tamaño} \begin{bmatrix} q \times 1 \\ (N-q) \times 1 \end{bmatrix} \\ \mu &= \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \text{de tamaño} \begin{bmatrix} q \times 1 \\ (N-q) \times 1 \end{bmatrix} \\ \Sigma &= \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \text{de tamaño} \begin{bmatrix} q \times q & q \times (N-q) \\ (N-q) \times q & (N-q) \times (N-q) \end{bmatrix} \end{aligned} \tag{2.34}$$

Se puede definir la distribución de probabilidad marginal de x_2 como:

$$p(x_1) = t_v(\mu_2, \Sigma_{22}) \tag{2.35}$$

y la distribución de probabilidad de x_1 condicionada a $x_2 = a$ [10]:

$$p(x_1|x_2 = a) = t_{v+N-q}\left(\mu_{1|2}, \frac{v+d_1}{v+N-q}\Sigma_{11|2}\right) \tag{2.36}$$

donde:

$$\begin{aligned} \mu_{1|2} &= \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(a - \mu_2) \\ \Sigma_{11|2} &= \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} \\ d_1 &= (a - \mu_2)^T \Sigma_{22}(a - \mu_2) \end{aligned} \tag{2.37}$$

Estas expresiones se utilizarán en la sección 4.2 de este documento, para realizar el proceso de inferencia en el modelo completamente bayesiano implementado.

2.5. Aprendizaje estructural

2.5.1. Introducción

Se define aprendizaje estructural como el entrenamiento, a partir de un conjunto de datos $\mathcal{D}$, de la estructura $\mathcal{G}$ que define las relaciones de independencia condicional entre las variables del modelo $\mathcal{X}$. Dicho de otra forma, el aprendizaje estructural pretende recuperar la estructura $\mathcal{G}'$ que factoriza la distribución de probabilidad conjunta que generó las variables, siempre desconocida, de las variables, $\mathcal{P}'(\mathcal{X})$.

Se identifican dos objetivos claros del aprendizaje estructural. Por una parte, el descubrimiento de relaciones, es decir, conocer las relaciones de dependencia que existen entre las variables. Y, por otro lado, la asignación de una densidad, es decir, entrenar una red capaz de generalizar y realizar predicciones para muestras no presentes en $\mathcal{D}$.

La complejidad de esta tarea reside en:

1. Los datos disponibles en $\mathcal{D}$ son ruidosos y, por lo tanto, no son totalmente representativos de la distribución $\mathcal{P}'(\mathcal{X})$. El ruido estadístico presente en los datos puede dar lugar a la inclusión u omisión de un enlace que provoque la generación de una estructura imprecisa ($\mathcal{G} \neq \mathcal{G}'$).

2. La existencia de varios *Perfect Maps*[1] asociados a una distribución $\mathcal{P}'(\mathcal{X})$ provoca que la estructura $\mathcal{G}'$ no se puede identificar a partir de los datos. Lo mejor que se puede esperar es que el algoritmo de aprendizaje estructural recupere un grafo $\mathcal{G}$ I-equivalente[2].

3. Como solución al problema del aprendizaje estructural, surgen algoritmos que generalmente requieren un alto coste computacional, siendo incluso intratables para un número determinado de variables.

En la literatura [4, 11] se distinguen principalmente dos aproximaciones al aprendizaje estructural:

1. *Constraint-based structure learning*: estos algoritmos dividen el aprendizaje estructural en dos etapas diferentes. En la primera, se realiza la evaluación de las relaciones de dependencia e independencia condicional de las variables. Una

[1] Se dice que un grafo G es un *Perfect map* de una distribución P, si el conjunto de independencias que representa, son las mismas que existen en P

[2] Se dice que dos grafos son I-equivalentes si representan ambos las mismas independencias

vez identificadas las relaciones, se busca una clase equivalente de estructuras que las cumpla. La principal desventaja de estos algoritmos reside en su alta sensibilidad a errores en las evaluaciones de independencia.

2. *Score-based structure learning*: estos métodos consideran una red bayesiana como un modelo estadístico y, por lo tanto, afrontan el problema de aprendizaje estructural como un problema de selección de modelo. El funcionamiento de los algoritmos basados en puntuación se basa en la definición de:

   - Un espacio de búsqueda de modelos, que define el conjunto de estructuras que van a ser evaluadas.
   - Una función de puntuación *score*, que puntúa como de bien se ajusta el modelo a los datos de entrenamiento.
   - Un algoritmo de búsqueda, que permite recorrer el espacio de búsqueda.

   Pese a que estos métodos son menos sensibles que los anteriores, que, en ocasiones, no resulta en la estructura deseada.

Considerando que las ventajas de los algoritmos basados en *scoring* superan sus inconvenientes, la primera tarea de este trabajo se centra en el estudio y uso de aproximaciones basadas en funciones de puntuación para el aprendizaje estructural de redes bayesianas gaussianas, a partir de un conjunto de entrenamiento completo.

2.5.2. Aprendizaje estructural basado en puntuación

Los algoritmos basados en una función de *score* afrontan el aprendizaje estructural como un problema de optimización, en el que se define una función de puntuación. Ésta es evaluada para cada estructura candidata perteneciente a un espacio de interés, mediante la utilización de un algoritmo de búsqueda.

En esta sección se describen, primero, algunos de los *scores* más relevantes en la literatura y, después, los tres métodos utilizados en este trabajo en términos de los algoritmos y espacios de búsqueda que utilizan.

Likelihood Score

El primer *score* se basa en función de verosimilitud o *likelihood function*, que mide la probabilidad del conjunto de entrenamiento dado el modelo.

Al igual que en la sección 2.3.1, maximizar la función de verosimilitud permite encontrar el modelo que mejor se ajusta a los datos. Sin embargo, en este caso, el

modelo se define como la dupla compuesta por la estructura del modelo, $\mathcal{G}$, y el conjunto de parámetros que lo definen dado el grafo, $\theta_{\mathcal{G}}$. En la literatura [4] esta dupla se encuentra representada como $\langle \mathcal{G}, \theta_{\mathcal{G}} \rangle$.

Desarrollando la máxima verosimilitud del modelo:

$$\begin{aligned} \max_{\mathcal{G},\theta_{\mathcal{G}}} L(\langle \mathcal{G}, \theta_{\mathcal{G}} \rangle : \mathcal{D}) &= \max_{G} [\max_{\theta_{\mathcal{G}}} L(\langle \mathcal{G}, \theta_{\mathcal{G}} \rangle : \mathcal{D})] \\ &= \max_{\mathcal{G}} [L(\langle \mathcal{G}, \theta_{\mathcal{G}}^{MLE} \rangle : \mathcal{D})] \end{aligned} \tag{2.38}$$

se concluye que, la estructura $\mathcal{G}$ final, perteneciente al espacio de búsqueda, es aquella que maximiza la verosimilitud utilizando los parámetros obtenidos mediante *MLE*.

La principal limitación de este *score* es que generalmente no disminuye al introducir un enlace. Por lo tanto, la red obtenida con máxima verosimilitud va a tener una independencia condicional solo cuando se encuentra de forma exacta en el conjunto de entrenamiento, hecho que no suele ocurrir debido al ruido estadístico de los datos.

De esta forma, la utilización de este *score* provoca un sobreajuste u *overfit* de la estructura a los datos de entrenamiento, en el cual se beneficia la red más compleja con respecto a la red más sencilla.

Score Bayesiano

Como función alternativa, basada en una perspectiva bayesiana, se propone en la literatura la utilización de un *score bayesiano*, el cual considera la incertidumbre asociada a la selección de $\langle \mathcal{G}, \theta_{\mathcal{G}} \rangle$.

Por lo tanto, definiendo una distribución a priori asociada a la variable $\mathcal{G}$ y aplicando el teorema de Bayes, se tiene:

$$P(\mathcal{G}|\mathcal{D}) = \frac{P(\mathcal{D}|\mathcal{G})P(\mathcal{G})}{P(\mathcal{D})} \tag{2.39}$$

Se define el *score bayesiano* a partir del logaritmo de la distribución a posteriori como:

$$score_B = \log P(\mathcal{D}|\mathcal{G})P(\mathcal{G}) + \log P(\mathcal{G}) \tag{2.40}$$

donde $P(\mathcal{G})$ es el *prior* de las estructuras. La definición de esta distribución nos permite "penalizar" o "beneficiar" determinadas estructuras. Como por ejemplo, las estructuras más complejas.

Al término $P(\mathcal{D}|\mathcal{G})$ se le denomina *marginal likelihood* de los datos dada la estructura, y considera la incertidumbre en los parámetros asociados a la estructura $\mathcal{G}$ mediante la marginalización sobre todo el espacio de posibles parámetros.

$$P(\mathcal{D}|\mathcal{G}) = \int_{\theta_\mathcal{G}} P(\mathcal{D}|\theta_\mathcal{G}, \mathcal{G}) P(\theta_\mathcal{G}|\mathcal{G}) d\theta_\mathcal{G} \tag{2.41}$$

De esta forma, existen varios *scores bayesianos* basados en la selección de *priors* de la dupla $\langle \mathcal{G}, \theta_\mathcal{G} \rangle$. Los más utilizados son los basados en la utilización de una distribución de *Dirichlet* como prior de los parámetros de la red, como por ejemplo *BD*, *BDe*, *BDeu* y *K2*.

Bayesian Information Criterion

El alto coste computacional, así como el hecho de que la existencia de una solución cerrada de 2.41 dependa de las distribuciones *prior* de $\langle \mathcal{G}, \theta_\mathcal{G} \rangle$, provoca que generalmente se utilicen aproximaciones del *score bayesiano*.

Si se hace uso de una distribución de dirichlet como prior de $\theta_\mathcal{G}$ y el número de muestras disponibles, N, tiende a infinito, de la ecuación 2.41 se obtiene:

$$\begin{aligned} \log P(\mathcal{D}|\mathcal{G}) &= L(\langle \mathcal{G}, \theta_\mathcal{G}^{MLE} \rangle : \mathcal{D}) - \frac{\log N}{2} |\mathcal{G}| \\ &= BIC(\mathcal{G} : \mathcal{D}) \end{aligned} \tag{2.42}$$

Donde se define $|\mathcal{G}|$ como la dimensión del modelo o el número de parámetros independientes asociados al grafo, y $L(\langle \mathcal{G}, \theta_\mathcal{G}^{MLE} \rangle : \mathcal{D})$ es el coeficiente de máxima verosimilitud.

A esta aproximación se le denomina *Bayesian Information Criterion o BIC*. De la ecuación 2.42 se concluye que *BIC* penaliza las estructuras de mayor complejidad mediante el factor $-\frac{\log N}{2}|\mathcal{G}|$ con respecto al *score* de máxima verosimilitud. Por esta razón, se le clasifica como un *score* basado en penalización.

Akaike Information Criterion

Otra función de puntuación basada en penalización muy utilizada es *Akaike Information Criterion* o *AIC*:

$$AIC(\mathcal{G} : \mathcal{D}) = L(\langle \mathcal{G}, \theta_\mathcal{G}^{MLE} \rangle : \mathcal{D}) - |\mathcal{G}| \tag{2.43}$$

Como se puede apreciar en la ecuación 2.43, *AIC* difiere de *BIC* en el factor de penalización. Esto hace que este *score* favorezca la generación de estructuras más complejas, que describen mejor las relaciones entre las variables que *BIC*, pero cuya capacidad generalizadora es inferior.

En este trabajo se evaluarán tres algoritmos diferentes de aprendizaje estructural de redes bayesianas gaussianas haciendo uso del *score BIC*. La razón por la cual se ha seleccionado este *score*, es por su capacidad de obtener estructuras, que describen las relaciones existentes entre las variables, pero que también sean capaces de realizar predicciones correctas ante nuevos datos no presentes en $\mathcal{D}$. Para la comprensión de los algoritmos en cuestión, es imprescindible la descripción de las principales características que hacen que este *score* sea utilizado:

1. Equivalencia: sean $\mathcal{G}_1$ y $\mathcal{G}_2$ dos estructuras I-equivalentes cualesquiera, un *score* cumple la propiedad de equivalencia si $score(\mathcal{G}_1 : D) = score(\mathcal{G}_2 : D)$ para cualquier $\mathcal{D}$.

2. Consistencia: un *score* cumple la propiedad de consistencia si, a medida que el número de datos tiende a infinito:

   - Es máximo para la estructura generadora de los datos, $\mathcal{G}'$.
   - Todas las estructuras no I-equivalentes a $\mathcal{G}'$ tienen valores inferiores.

3. Descomposición: un *score* cumple la propiedad de descomposición si, para cualquier estructura $\mathcal{G}$ se puede expresar como una suma de *scores* locales.

$$score(\mathcal{G} : \mathcal{D}) = \sum_k \text{FamScore}(X_k|\text{Pa}_{X_k}^{\mathcal{G}} : \mathcal{D}) \tag{2.44}$$

   Siendo FamScore una medida del *score* realizada sobre la estructura formada por un nodo y sus padres.

2.5.3. Algoritmos de búsqueda utilizados

Una vez descritos algunos de las funciones de puntuación más utilizadas en la literatura, se realiza una descripción de los diferentes algoritmos de aprendizaje estructural utilizados en este trabajo. Estos algoritmos difieren los uno de los otros en función del espacio de búsqueda sobre el que se va a evaluar la función de coste, y el algoritmo de búsqueda que se va a emplear para recorrer dicho espacio.

Búsqueda exhaustiva

La aproximación más intuitiva al aprendizaje estructural basado en *scores* se basa en establecer como espacio de búsqueda todas las estructuras existentes para un conjunto de nodos, evaluar la función de puntuación sobre todos los grafos posibles, y escoger finalmente aquel que maximice dicha puntuación. Sin embargo, para un total de n variables, el número de posibles estructuras $r(n)$ que definen todas las relaciones posibles de independencia condicional, varía de forma súper-exponencial, tal que:

$$r(n) = \sum_{i=1}^{n} (-1)^{i+1} \binom{n}{i} 2^{i(n-1)} r(n-1) = n^{2^{O(n)}} \tag{2.45}$$

De esta forma, para un conjunto de 10 nodos tenemos un espacio de $r(10) \simeq 4{,}2 \times 10^{18}$ posibles estructuras. Por lo tanto, este algoritmo es poco eficiente computacionalmente, hasta que, para un número de variables igual o mayor a 7, resulta intratable.

Por esta razón, las técnicas de aprendizaje estructural basados en *scores* se basan en limitar el espacio de búsqueda y la utilización de algoritmos de búsqueda más óptimos.

Greedy Hill climbing

Con el objetivo de reducir el coste computacional, se diseñan algoritmos de búsqueda que recorren el espacio de estructuras hasta obtener aquella que maximiza la función de puntuación. La mayoría de los procedimientos de búsqueda que aparecen en la literatura se basan en procedimientos de búsqueda local, como por ejemplo el *Greedy Hill Climbing*[12]. d El algoritmo *Greedy Hill Climbing* se basa en las siguientes etapas:

1. Inicialización: se define una estructura inicial y se evalúa la función de puntuación. Esta red suele ser inicialmente una red sin enlaces, una red aleatoria o una red previamente construida.

2. Obtención de los grafos vecinos: se definen los vecinos como el conjunto de DAGs idénticos a la estructura en cuestión, salvo por una modificación u operación, O. Dichas operaciones son: incorporación de un enlace, eliminado de un enlace y cambio de orientación de un enlace. En la imagen 2.4 se representan los vecinos de la estructura representada en 2.3.

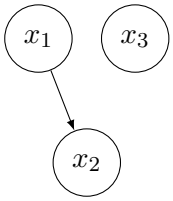


Figura 2.3: Estructura aleatoria inicial para 3 variables

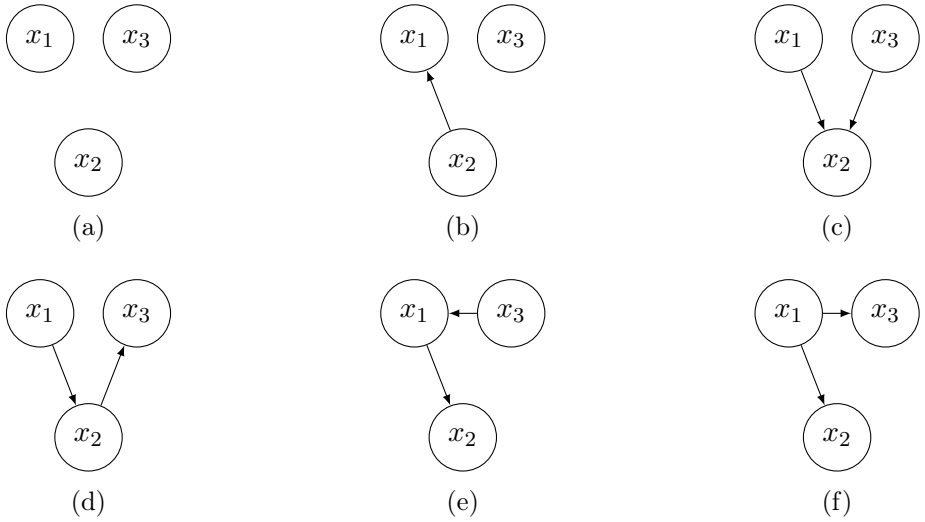


Figura 2.4: Vecinos de 2.3. Obtenidas por: (a) eliminado de enlace. (b) cambio de orientación. (c)-(f) inclusión de enlace.

3. Evaluación de los vecinos: se calcula el *score* de todas estructuras vecinas y se selecciona como estructura actual la estructura cuyo *score* es el más alto.

4. Se repiten los puntos 2 y 3 hasta que se alcance una estructura actual cuyos vecinos no tengan *scores* más elevados.

Este algoritmo hace uso de la propiedad de descomposición de la función de puntuación, para reducir el coste computacional causado por la evaluación de los grafos vecinos. De tal forma que, tras la evaluación de la estructura actual $\mathcal{G}_1$, la evaluación de una estructura vecina $\mathcal{G}_2$ puede realizarse tal que:

$$score(\mathcal{G}_2 : \mathcal{D}) = score(\mathcal{G}_1 : \mathcal{D}) + \delta(\mathcal{G}_1 : O) \tag{2.46}$$

donde $\delta(\mathcal{G} : O)$ es un factor que se calcula localmente, y no sobre la totalidad del grafo. Dicho factor toma como valor δ_1 si el operador corresponde a la inclusión o eliminado de un enlace dirigido a un nodo X, y toma un valor δ_2 si el operador corresponde a un cambio de orientación de un enlace de X a Y. Ambas expresiones se representan

en la ecuación 2.47.

$$\begin{aligned}\delta_1(\mathcal{G}_1 : O) &= \text{FamScore}(X, Pa_X^{\mathcal{G}_2}) - \text{FamScore}(X, Pa_X^{\mathcal{G}_1}) \\ \delta_2(\mathcal{G}_1 : O) &= \text{FamScore}(X, Pa_X^{\mathcal{G}_2}) + \text{FamScore}(X, Pa_Y^{\mathcal{G}_2}) \\ &\quad - \text{FamScore}(X, Pa_X^{\mathcal{G}_1}) - \text{FamScore}(X, Pa_X^{\mathcal{G}_1})\end{aligned} \tag{2.47}$$

De esta forma, la evaluación de cada uno de los vecinos se reduce al cálculo de la función de puntuación de la familia que ha cambiado con respecto a la estructura actual.

Las principales limitaciones de este algoritmo se basan en su tendencia a converger en dos situaciones no deseadas:

1. Presencia de un máximo local: caso en el que ningún vecino tiene un *score* superior al de la red actual. Sin embargo, dicha estructura no corresponde a la estructura real, generadora de los datos.

2. Presencia de un *plateau*: caso en el que las estructuras vecinas son *I-equivalentes* a la red actual y, por lo tanto, por la propiedad de equivalencia, todas tienen el mismo *score*.

K2

El algoritmo *K2* [13], es el ejemplo más utilizado basado en la reducción del espacio de búsqueda en función de la definición de un orden de las variables que conforman el modelo. Es decir, se limita el espacio de búsqueda a las estructuras $\mathcal{G}$ que cumplen que $X_i \in \text{Pa}_{X_j}^{G}$ solo si $X_i \prec X_j$, es decir un determinado nodo sólo puede tener como padres aquellos nodos que le precedan en el orden establecido.

De esta forma, y con la utilización de un *score* que cumpla la propiedad de descomposición, la búsqueda de $\mathcal{G}$ se simplifica considerablemente al poder dividirse en subtareas, que tratan de encontrar los padres de cada nodo X_i independientemente del resto.

Los padres de una variable son aquellos que maximicen la función de *score* local de la familia. Por lo tanto, la estructura $\mathcal{G}$ final es aquella que cumple:

$$\text{Pa}_{X_i}^{\mathcal{G}} = \arg\max_{Y_i} \text{FamScore}(X_i | Y_i : \mathcal{D}) \tag{2.48}$$

para cada uno de las variables del modelo, donde Y es el conjunto de posibles padres de X_i siguiendo el orden $\prec$.

De esta forma, el algoritmo *K2* se puede dividir en las siguientes etapas:

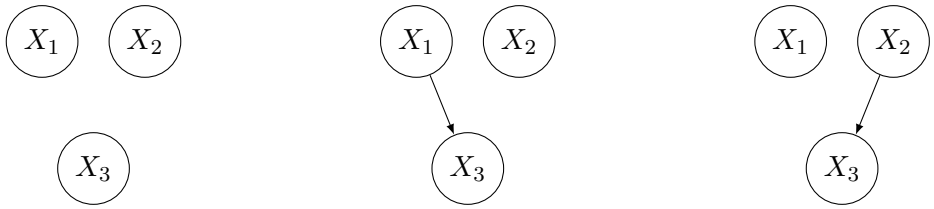
1. Definición del orden topológico de las variables del problema.

$$X_1 \prec X_2 \prec X_3 \tag{2.49}$$

2. Se recorren las variables según el orden topológico. Para una mayor comprensión del proceso iterativo realizado para cada nodo, se desarrolla para el nodo X_3 del orden 2.49.

   a) Detección de los padres potenciales de la variable en función del orden topológico. En este ejemplo, X_1 y X_2. Remarcar que el primer nodo nunca tiene padres.

   b) Evaluación del *score* local, sobre la familia formada por el nodo actual y cada uno de los padres potenciales según el orden. Se selecciona la familia que maximiza el *score*. En el ejemplo propuesto en la figura 2.5, la introducción del padre X_1 maximiza el *BIC*.



(a) $BIC = -325{,}35$ (b) $BIC = -324{,}25$ (c) $BIC = -328{,}11$

Figura 2.5: Evaluación de los padres potenciales de X_3

   c) De la misma forma, se evalúa la incorporación de nuevos padres a la familia actual. Se repite el proceso hasta que la incorporación de ningún padre potencial mejore el *score*.

(a) $BIC = -324{,}25$ (b) $BIC = -323{,}02$

Figura 2.6: Evaluación de los padres potenciales de X_3

En el ejemplo propuesto, la estructura que maximiza el *score* es la presentada en 2.6(b).

3. Se repite el proceso anterior para cada uno de los nodos.

Como consecuencia, este algoritmo representa una solución muy rápida y ampliamente utilizada en los problemas en los que el conocimiento experto permite identificar un orden topológico de los nodos de la red. Sin embargo, tiene como principal debilidad su sensibilidad al orden establecido inicialmente por el usuario. En el caso de que este orden sea establecido por conocimiento experto, dicho problema se reduce considerablemente.

GES

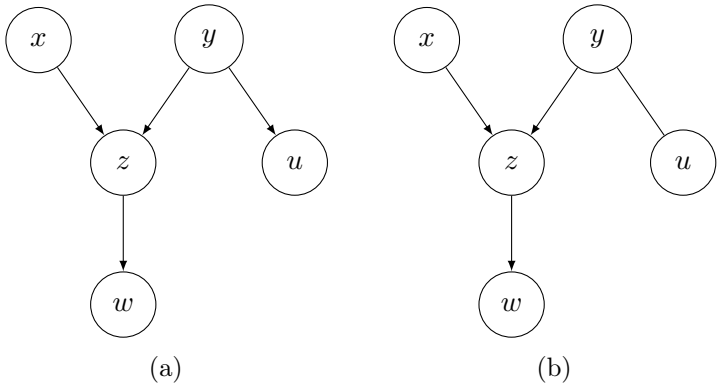
Pese a que en la literatura se presentan diferentes alternativas para la solución de las limitaciones del algoritmo *Greedy Hill Climbing*, una de las principales soluciones consiste en la modificación del espacio de búsqueda. Mientras que *GHC* utiliza como espacio de búsqueda el conjunto total de DAGs, otros algoritmos proponen la utilización de un espacio de *I-equivalencia*, en el cual, cada estado, o clase, es una representación de un conjunto de estructuras *I-equivalentes*.

Dicha representación se denomina *PDAG*, y se basa en un grafo parcialmente dirigido en el que:

- Los enlaces dirigidos son aquellos que todas las estructuras tienen que tener en la misma dirección para pertenecer a la misma clase. Es decir, aquellos cuyo cambio de orientación puede generar o eliminar estructuras en V (Como por ejemplo, la figura 2.7(a), presenta una estructura en V en $x \rightarrow z \leftarrow y$).
- Los enlaces no dirigidos son aquellos que toda estructura tiene que tener, independientemente de su dirección, para formar parte de la misma clase.

Como ejemplo de *PDAG*, se presenta la figura 2.7 correspondiente al famoso problema del estudiante.

En la figura 2.7, se representa un grafo y el *PDAG* que representa el conjunto de estructuras I-equivalentes. Como se puede apreciar, independientemente de la dirección del enlace entre u e y, las estructuras resultantes son *I-equivalentes* a (a), por lo dicho enlace no tiene dirección en el *PDAG* que representa dicha *I-equivalencia*.



(a) (b)

Figura 2.7: (a) Grafo original (b) *PDAG*

En este caso, los vecinos de un *PDAG* son aquellos que difieren en un único operador de suma o eliminado de enlaces. Cabe destacar que, en este caso, el cambio de orientación de enlaces no es considerado un operador, pues puede originar estructuras pertenecientes a la misma clase.

Por último, la evaluación de una clase *PDAG*, se realiza la evaluación de un DAG cualquiera perteneciente a la clase en cuestión. Debido a la propiedad de equivalencia del *score* todas las estructuras pertenecientes a una clase constan del mismo *score*.

El algoritmo más utilizado es el algoritmo de *Greedy Equivalence Search* o *GES* [14]. Éste se divide en las siguientes etapas:

1. Inicialización: el algoritmo se inicializa con un grafo sin enlaces.
2. Introducción de enlaces: se incluyen enlaces hasta que la introducción de ningún enlace restante mejore el *score*.
3. Eliminado de enlaces: se realiza el proceso inverso al anterior, se quitan enlaces hasta que en ningún caso mejore el *score*.

De esta forma, el algoritmo *GES* garantiza, en caso de tener un alto número de muestras, generar una estructura *I-equivalente* a la estructura real de los datos.

Capítulo 3

Diseño

Tras el estudio de las bases teóricas de este trabajo, en esta sección se describen todas las consideraciones tenidas en cuenta en la etapa de diseño del proyecto.

Esta etapa incluye una descripción de la base de datos proporcionada por la empresa, la cual será utilizada para la realización de ambas tareas del presente trabajo. Tras esto, se presenta una descripción del *baseline* realizado en 2019 y las principales consideraciones en su diseño. Adicionalmente, se presenta una descripción del entorno de trabajo sobre el que se han realizado los experimentos. Para finalizar, se especifican las métricas utilizadas para la evaluación de los experimentos del capítulo 4 de este documento.

3.1. Base de datos

La empresa colaboradora es una empresa proveedora y gestora de varias centrales de generación de electricidad. Entre sus actividades, se encuentra la generación de protocolos y recomendaciones sobre el procedimiento de generación de electricidad. Para ello, se consta de una base de datos, proporcionada por la empresa, que se basa en un libro Excel, de 118 669 entradas, con el siguiente formato:

Planta	**Ciclo**	**Fecha**	**Campo**	**Medida**
Planta 1	22	21/07/2011	*Control 1*	6.87379
Planta 1	22	22/07/2011	*Control 1*	6.89552
Planta 1	22	21/07/2011	*Control 2*	25.08

Donde:

Planta es el nombre de una central generadora de electricidad gestionada por la empresa. Han sido consideradas 5 centrales, las cuales denominaremos: *Planta 1, Planta 2, Planta 3, Planta 4 y Planta 5*. Dichas centrales pueden ser divididas en dos grupos, en función de los métodos y protocolos que usan. De esta forma, el *Grupo 1* está formado por las centrales *Planta 1* y *Planta 2*, y el *Grupo 2* por las centrales *Planta 3*, *Planta 4* y *Planta 5*.

Ciclo es la numeración que indica el ciclo de trabajo del generador de electricidad en el que han sido tomadas las medidas. Es decir, el ciclo es la división temporal que engloba el funcionamiento completo del generador de electricidad. El número de ciclos proporcionados por cada central es diferente.

Fecha corresponde a la fecha en la que se ha realizado la medida. La periodicidad a la hora de captar una medida depende de la variable a medir y de la central.

Campo es el nombre de la variable cuyo valor ha sido guardado. Pese a que en la base de datos entregada hay un total de 27 campos diferentes, solo serán de interés en este proyecto 11. Dichas variables se pueden dividir en:

- **Señales *Control***: corresponden a las variables controlables directamente desde cada central. Serán denominados como *Control 1, Control 2, Control 3, Control 4* y, en el caso de las centrales pertenecientes al *Grupo 2*, *Control 5*.

- **Señales *Medida***: corresponden a subproductos del proceso de generación de electricidad cuyo valor no es observable a lo largo de todo el ciclo de trabajo y, por lo tanto, que se quieren predecir. Estas señales se encuentran en la base de datos divididas en dos componentes diferentes. Dichas componentes serán denominadas *Medida 1s, Medida 1i, Medida 2s, Medida 2i, Medida 3s, Medida 3i, Medida 4s y Medida 4i*.

Por último, el campo **Medida** es *el valor de la señal "Campo", adquirido en el ciclo "Ciclo" de la central "Planta", el día "Fecha"*. Es importante tener en cuenta que una variable puede tener diferentes rangos dinámicos en función de la central, y, que puede haber medidas erróneas debido algún problema en el sistema de captación de las centrales.

El estudio de las señales presentes en la base de datos queda fuera del alcance de este trabajo. Sin embargo, en caso de que sea de interés para el lector, se recomienda la lectura de [1].

3.2. Baseline

Una vez descrita la base de datos utilizada en ambas tareas, se realiza la descripción del modelo desarrollado en 2019, y que es considerado como el *Baseline* de este trabajo. Para una descripción más detallada del desarrollo del modelo, se recomienda la lectura de los capítulo 3 y 4 de [1].

De esta forma, el *baseline* se basa en un modelo gráfico probabilístico para cada una de las centrales presentes en la base de datos. Estos modelos han sido utilizados para la predicción de las señales *Medida* a partir de las señales de *Control* en cada central. En concreto, todos se basan en una red bayesiana gaussiana, cuya estructura ha sido definida por el conocimiento experto de la empresa y el grupo AUDIAS. Dicha estructura es representada en la figura 3.1.

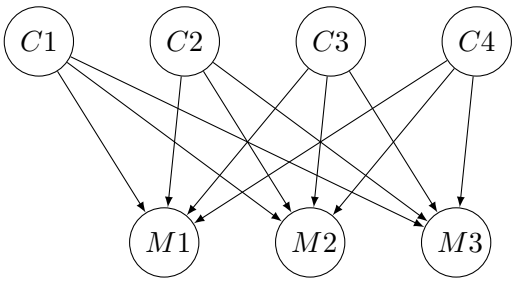


Figura 3.1: Red bayesiana gaussiana para las centrales del *Grupo 1*. Las variables *Control* son representadas con C y las variables *Medida* con M

Los modelos de las centrales pertenecientes al *Grupo 2* han sido recientemente modificados, incluyendo la variable *Control 5*, tal como se representa en la figura 3.2.

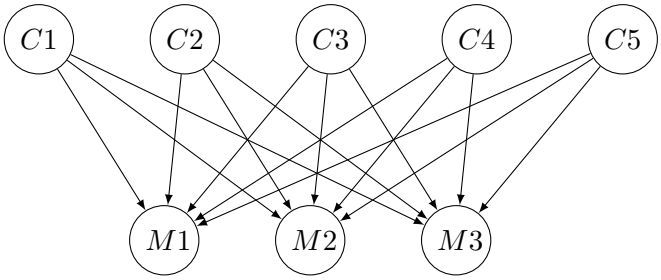


Figura 3.2: Red bayesiana gaussiana para las centrales del *Grupo 2*. Las variables *Control* son representadas con C y las variables *Medida* con M

Analizando la estructura de ambos modelos, supondremos, por indicación de la empresa, que las variables controlables a lo largo del ciclo, *Control 1, Control 2, Control 3 y Control 4*, son independientes entre sí cuando no se conocen las variables

Medida. Por otro lado, las variables *Medida*, cuyo valor a lo largo del ciclo se pretende predecir, son: *Medida 1, Medida 2, y Medida 3*. Éstas son independientes entre sí cuando se conocen todas las variables de control. Adicionalmente, se puede observar que las todas las variables *Medida* dependen directamente de todas las variables *Control*.

A modo de resumen, se exponen a continuación las principales consideraciones en el diseño de los modelos:

1. Para el entrenamiento de los modelos, solo se utilizan instantes temporales en los todas las señales de interés han sido capturadas. De esta forma, se evitan errores por *data imputation*, causados principalmente por la interpolación de las variables *Medida*, debido a que tienen un periodo de captación y una gran cantidad de valores atípicos, en comparación con las variables *Control*.

2. Las variables *Medida* son obtenidas mediante la suma de sus dos componentes en los instantes temporales en los que ambas son adquiridas.

3. Debido a su naturaleza, y de acuerdo a las especificaciones de la empresa, se realiza una transformación previa de las variables *Control 1* y *Control 2*. De forma que:

   - A la señal *Control 1* se le aplica un eliminado de valores, considerados erróneos, superiores e inferiores a 5'5 y 7'5 respectivamente.
   - Se realiza la diferenciación matemática de *Control 2*, restando al valor en un día concreto su valor anterior.

4. Se realiza una interpolación lineal de las variables *Control*, en los días en los que no se tienen algunos de sus valores.

5. Con el objetivo de que los datos se ajusten mejor al modelo, se aplica una gaussianización marginal de cada variable basada en ecualización de histogramas (ver Anexo B).

6. El entrenamiento de la red se realiza mediante el algoritmo de máxima verosimilitud o *MLE* y la inferencia se realiza mediante *Variable Elimination*. Tras la inferencia, se realiza el proceso inverso de gaussianización para obtener las variables en el espacio químico-físico, interpretable por la empresa.

De esta forma, el proceso completo desarrollado en 2019 queda descrito por el diagrama representado en la figura 3.3.

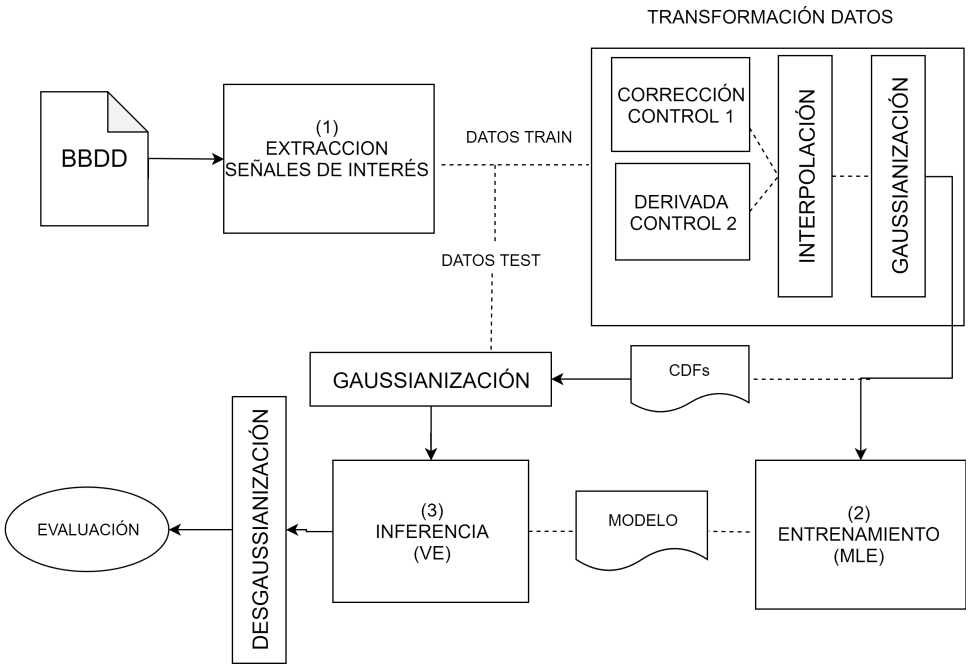


Figura 3.3: Diagrama final del proceso de desarrollo

3.3. Entorno de trabajo

Tanto el *baseline*, como los experimentos relacionados con ambas tareas de este trabajo, han sido implementadas en Matlab^{TM} 2019b. Mientras que para la implementación del modelo *fully bayesian* de la tarea 2 no se ha utilizado ningún toolbox, la tarea de aprendizaje estructural se ha realizado mediante la utilización del paquete externo *Bayesian Network Toolbox* o *BNT*[15]. *BNT* es un paquete *open-source* desarrollado en Matlab, entre 1997 y 2002, por Kevin Murphy, para modelos gráficos dirigidos. Este paquete soporta diferentes distribuciones de probabilidad para los nodos de las redes bayesianas, diferentes algoritmos de inferencia exacta, inferencia aproximada, aprendizaje de parámetros, y aprendizaje estructural. En concreto, para el aprendizaje estructural se hace uso de la extensión de BNT propuesta por Olivier Chapel y Philippe Leray, *Structure Learning Package* o *SLP*[16]. Sobre dicho paquete se han realizado unos breves cambios para adaptar los algoritmos *GHC* y *GES* a variables gaussianas.

La razón por la cual este trabajo ha sido implementado en Matlab se resume en los siguientes puntos:

1. Es un lenguaje de programación previamente conocido por la empresa colaboradora.

2. La existencia de un paquete tan potente, con extensa documentación, licencia permisiva (GNU GPL) y ampliamente utilizado como es *BNT*, ha priorizado el uso de Matlab^{TM} frente a otros lenguajes más utilizados en investigación, como Python, en los cuales no se han encontrado alternativas con funcionalidades similares a *BNT*.

3.4. Métricas de evaluación

Tras la descripción de la base de datos utilizada, el sistema *baseline* y el entorno de trabajo, en esta sección se describen las métricas utilizadas para la evaluación de los experimentos descritos en el capítulo 4 de este documento.

Así pues, las métricas que van a ser utilizadas se pueden dividir en 2 clases:

- Métricas para la comparativa de estructuras: por un lado, se va a utilizar la propia función de puntuación *BIC* y, por otro, se va a hacer uso de la denominada *"editing measure"*, definida en [16].

  La medida "*editing measure*", que en este documento vamos a denominar *em*, se define como la longitud de la secuencia de operaciones mínima, para transformar el grafo real de un conjunto de variables, en un grafo determinado.

- Métricas para evaluación de predicción: para medir los resultados en la predicción de las variables *Medida*, se va a hacer uso de la métrica *RMSE*, definida al que:

$$RMSE(x) = \sqrt{\frac{\sum_{t=1}^{T}(\widehat{x} - x)^2}{T}} \tag{3.1}$$

  Donde $\hat{x}$ es el valor real de la señal a predecir, x es el valor medio predicho por el modelo en cada instante y T es el número de predicciones realizadas. Esta medida tiene en cuenta el error de exactitud del valor predicho con respecto al valor real.

  Por otro lado, se va a hacer uso del logaritmo de la función de verosimilitud de los datos con el modelo, como medida alternativa al *RMSE*.

$$LL(D|\theta) = \prod_{i=1}^{T} p(x_i|\theta) \tag{3.2}$$

Esta medida tiene en cuenta la consistencia del modelo generado con los datos de prueba. Es una medida de ajuste del modelo a los datos. Nótese que, aunque en un entorno bayesiano es necesario introducir la incertidumbre presente, y no solo basarse en los datos de entrenamiento; LL mide el ajuste puramente a los datos. La idea subyacente es que, al ser los datos de test "no vistos" por el modelo, esta medida contempla en cierto modo la capacidad de de generalización del modelo entrenado.

Capítulo 4

Experimentos

En este capítulo se describen los experimentos, y los resultados obtenidos, relacionados con las dos tareas de este TFM.

4.1. Tarea 1: Aprendizaje estructural

El principal objetivo definido para esta tarea (ver sección 1.2) se basa en el aprendizaje de estructuras alternativas a las definidas en las figuras 3.1 y 3.2, que describan las relaciones de independencia condicional existentes en los datos.

Para ello, esta tarea se ha dividido en tres experimentos diferentes:

1. Primero, se realiza una comparativa de los tres algoritmos utilizados, con un conjunto de datos gaussianos cuya estructura generadora ya es conocida.

2. Tras esto, se comprueba la capacidad de generalización de los modelos entrenados por estos algoritmos.

3. Por último, se generan las estructuras finales para cada una de las centrales de interés.

4.1.1. Comparativa de algoritmos

En esta sección se evalúan los tres algoritmos de aprendizaje estructural de redes bayesianas estudiados: *Greedy Hill-Climbing*, *K2*, y *Greedy Equivalent Search*. Para ello, se realiza una comparativa de las redes generadas por los tres algoritmos, utilizando un conjunto de datos gaussianos cuya estructura real es conocida.

Tras una búsqueda de bases de datos que cumplan las anteriores condiciones, no se encuentra ninguna acorde a lo que necesitamos. Por lo tanto, para esta comparativa inicial se decide utilizar un dataset formado por la extracción de muestras del modelo *baseline* de la central *Planta 1*, cuya estructura queda definida por la figura 3.1. Así, el modelo *Baseline* de la primera central se utiliza como "generador de datos sintéticos" para el entrenamiento del aprendizaje estructural, y por lo tanto, las etiquetas "ground-truth" (es decir, el grafo "real" generador de los datos) son conocidas. En consecuencia, se puede evaluar adecuadamente el rendimiento del aprendizaje estructural en este conjunto de datos. De esta forma, se generan:

1. Conjunto de entrenamiento para el aprendizaje de las estructuras: la longitud de este conjunto de entrenamiento es variable, lo que nos va a permitir evaluar el rendimiento de los algoritmos para diferentes números de datos.

2. Conjunto de test: utilizado para la comparativa de las estructuras generadas, usando para ello las métricas *BIC* y *em*. El tamaño de este conjunto es fijo, de 1000 muestras.

Así pues, los resultados obtenidos son representados en la tabla 4.1. En dicha tabla, se pueden observar las estructuras generadas y las métricas obtenidas en test, utilizando conjuntos de entrenamiento de diferentes tamaños, y diferentes algoritmos de aprendizaje estructural.

En el caso del algoritmo *Greedy Hill Climbing* (*GHC* en la tabla) se registran los resultados para dos estructuras "semilla" o iniciales diferentes. Mientras que GHC_1 corresponde a la utilización de un grafo sin enlaces como estructura semilla, GHC_2 corresponde a la utilización de un grafo aleatorio[1].

Por otro lado, las pruebas con el algoritmo *K2* han sido realizadas utilizando dos órdenes topológicos definidos por la empresa. La entrada $K2_1$ en la tabla corresponde a los resultados obtenidos utilizando el orden 4.1, y la entrada $K2_2$ corresponde a los resultados obtenidos utilizando el orden 4.2.

$$\begin{aligned} &\textit{Control 1} \prec \textit{Control 3} \prec \textit{Control 4} \prec \textit{Control 2} \ ... \\ &... \prec \textit{Medida 3} \prec \textit{Medida 2} \prec \textit{Medida 1} \end{aligned} \tag{4.1}$$

[1]Este grafo se mantiene para todas las pruebas realizadas

$$\begin{aligned} &\textit{Control 3} \prec \textit{Control 1} \prec \textit{Control 4} \prec \textit{Control 2} \ ... \\ &... \ \prec \textit{Medida 3} \prec \textit{Medida 2} \prec \textit{Medida 1} \end{aligned} \tag{4.2}$$

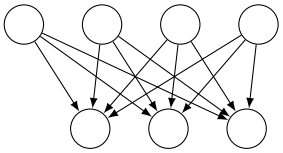
Adicionalmente, en la tabla 4.2 se presentan los tiempos de ejecución de los tres algoritmos utilizados, para diferentes tamaños de conjunto de entrenamiento.

De los resultados obtenidos en las tablas 4.1 y 4.2 se extraen diferentes conclusiones. Primero, se observa que los resultados obtenidos con los tres algoritmos son similares, para cada conjunto de entrenamiento utilizado. En este caso, todos reconstruyen la estructura generadora de los datos cuando se dispone de un conjunto de entrenamiento de 5000 muestras.

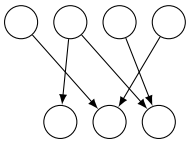
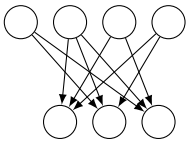
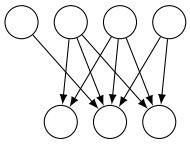
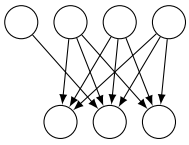
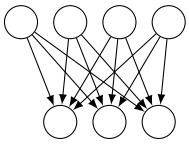
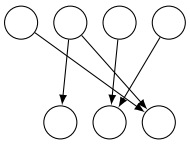
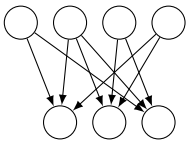
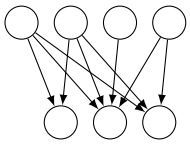
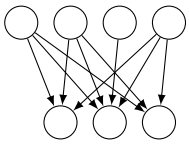
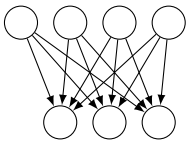
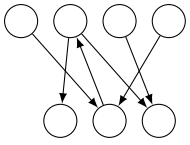
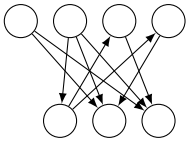
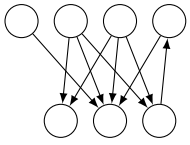
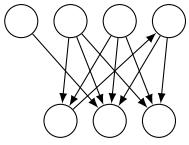
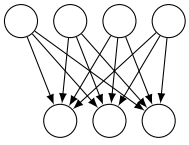
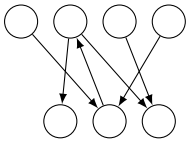
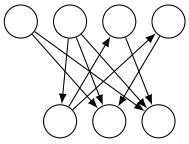
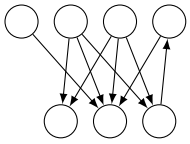
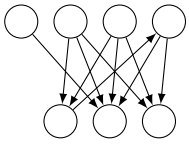
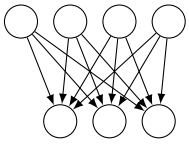
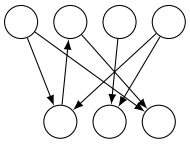
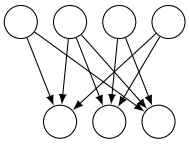
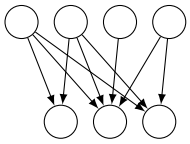
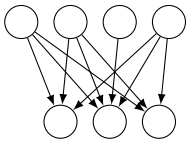
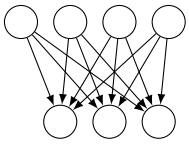
El algoritmo *K2*, es considerablemente el algoritmo más rápido con el que se han realizado pruebas, siendo hasta 33 veces más rápido que *GHC* y 25 veces más que *GES*. Además, resulta ser un algoritmo bastante insensible al tamaño del conjunto de entrenamiento, alcanzando un valor bajo de la métrica *em* para un conjunto de entrenamiento de 250 datos. Sin embargo, la principal limitación de éste algoritmo reside en su sensibilidad al orden topológico definido. Hay que tener en cuenta que, en estas pruebas, la estructura generadora no infringe los órdenes definidos por la empresa. En el caso de que esto no ocurriese, la estructura generadora no podría ser reconstruida con este algoritmo. Por último, se puede ver que, al ser los órdenes definidos por la empresa (4.1 y 4.2) muy parecidos, las estructuras generadas, y las métricas obtenidas, son muy similares para ambos experimentos, $K2_1$ y $K2_2$.

Por otra parte, pese a que el algoritmo *Greedy Hill Climbing* reconstruye la estructura generadora para $N = 5000$, es el algoritmo que peores resultados obtiene para el experimento en cuestión. Para cualquier número de muestras de entrenamiento, reconstruyen las redes con mayor diferencia con respecto a la estructura real. Además, se puede comprobar que el algoritmo converge en las mismas estructuras para las dos inicializaciones diferentes.

Por último, el algoritmo *GES* es el algoritmo con el que mejores resultados se han obtenido para conjuntos de entrenamiento pequeños ($N = 100$ y $N = 250$). Adicionalmente, se comprueba que la búsqueda por clases de equivalencia es más rápido que el tradicional *Hill Climbing*.



Estructura generadora

Algoritmo	$N_{train} = 100$	$N_{train} = 250$	$N_{train} = 500$	$N_{train} = 1000$	$N_{train} = 5000$
$K2_1$	 $BIC = -234{,}82$ $em = 7$	 $BIC = -233{,}73$ $em = 3$	 $BIC = -233{,}03$ $em = 3$	 $BIC = -232{,}36$ $em = 2$	 $BIC = -231{,}87$ $em = 0$
$K2_2$	 $BIC = -234{,}82$ $em = 7$	 $BIC = -233{,}73$ $em = 3$	 $BIC = -233{,}03$ $em = 3$	 $BIC = -232{,}36$ $em = 2$	 $BIC = -231{,}87$ $em = 0$
GHC_1	 $BIC = -234{,}55$ $em = 7$	 $BIC = -233{,}75$ $em = 5$	 $BIC = -233{,}03$ $em = 4$	 $BIC = -232{,}38$ $em = 3$	 $BIC = -231{,}87$ $em = 0$
GHC_2	 $BIC = -234{,}55$ $em = 7$	 $BIC = -233{,}75$ $em = 5$	 $BIC = -233{,}03$ $em = 4$	 $BIC = -232{,}38$ $em = 3$	 $BIC = -231{,}87$ $em = 0$
GES	 $BIC = -234{,}13$ $em = 6$	 $BIC = -232{,}70$ $em = 2$	 $BIC = -233{,}03$ $em = 3$	 $BIC = -232{,}36$ $em = 2$	 $BIC = -231{,}87$ $em = 0$

Cuadro 4.1: Para cada algoritmo y tamaño del conjunto de train se representa: (1) Estructura (2) BIC dividido entre 100 (3) *editing measure*

Algoritmo	N=100	N=250	N=500	N=1000	N=5000
K2	0.6	1.0	2.2	4.4	20.5
GHC	20.3	42.7	63.9	134.7	786.3
GES	12.5	35.45	54.9	103.5	580.3

Cuadro 4.2: Tiempos de ejecución (en segundos) de los algoritmos de aprendizaje estructural

Tras la finalización de este experimento, utilizado para el estudio del aprendizaje estructural basado en *scores*, para la familiarización con el entorno de *SPL* de *BNT*, y para comprobar la adaptación de los algoritmos anteriores a variables gaussianas, se dispone a comprobar la capacidad de generalización de los modelos cuyas estructuras son aprendidas directamente de los datos.

4.1.2. Inferencia con las estructuras alternativas

Tras comprobar el funcionamiento de los tres algoritmos de aprendizaje estructural estudiados, en esta sección se presentan los experimentos realizados para comprobar la capacidad de generalización de las estructuras aprendidas mediante *GHC*, *K2* y *GES*. Con este objetivo, en este experimento, al igual que en 2019, se propone la predicción de las variables *Medida* a partir de las variables *Control*. De este modo, si las estructuras aprendidas predicen similarmente bien que el modelo *baseline*, se puede concluir que el procedimiento, y las estructuras generadas, son correctos.

Para ello, se propone la utilización del algoritmo de evaluación *K-Fold Cross Validation*, siendo K el número de ciclos disponibles para cada central en la base de datos. Así pues, el experimento llevado a cabo para cada una de las centrales se basa en:

1. Obtener los ciclos con valores de todos los elementos del diseño.
2. De los ciclos obtenidos, se utiliza uno para la evaluación del modelo final, y el resto para el entrenamiento de la estructura y de los parámetros que definen el modelo.
3. La evaluación consiste en el cálculo del *RMSE* entre las predicciones de cada variable *Medida* y su valor real, y el cálculo del logaritmo de la verosimilitud con el modelo previamente entrenado.
4. Se repiten los pasos 2 y 3, de tal forma que se realice la predicción de todos los ciclos.

Cabe destacar que, en cada iteración de la validación cruzada, los datos de entrenamiento han sido transformados siguiendo el proceso descrito en la sección 3.2, para el *baseline*. De esta forma el proceso llevado a cabo para cada iteración, queda reflejado en la figura 4.12

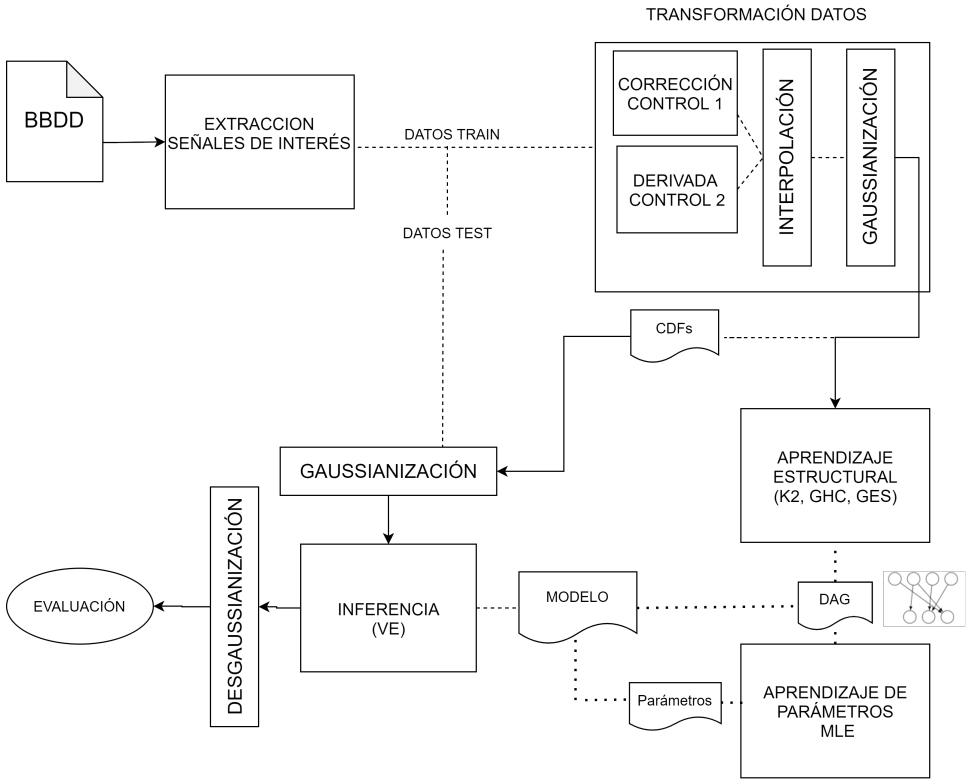


Figura 4.1: Diagrama final del proceso realizado para cada iteración del algoritmo *K-Fold Cross Validation*

Para simplificar el análisis por parte del lector, en las tablas 4.3 - 4.4[2] y 4.5 se adjuntan los valores medios por central de las métricas evaluadas en cada iteración de la validación cruzada.

[2]Se adjuntan exclusivamente las tablas correspondientes a las variables *Medida 1* y *Medida 2*. La variable *Medida 3* ya no es de importancia para la empresa.

Algoritmo	Planta 1	Planta 2	Planta 3	Planta 4	Planta 5
baseline	**6.23**	31.6	7.36	4.38	**15.48**
$K2_1$	6.26	**31.25**	**7.23**	**4.28**	**20.35**
$K2_2$	6.26	31.34	**7.23**	**4.28**	**20.35**
GHC_1	6.30	31.52	6.76	8.73	28.64
GHC_2	6.31	33.03	7.72	7.08	30.01
GES	**6.23**	**31.2**	**6.78**	**4.27**	27.81

Cuadro 4.3: RMSE medio de la validación cruzada en la predicción de *Medida 1* para cada central y algoritmo de aprendizaje estructural

Algoritmo	Planta 1	Planta 2	Planta 3	Planta 4	Planta 5
baseline	0.95	**1.94**	**1.25**	1.01	**2.67**
$K2_1$	0.94	1.96	1.29	**0.93**	2.69
$K2_2$	**0.93**	1.97	1.29	**0.93**	2.69
GHC_1	**0.93**	1.96	1.57	0.95	3.15
GHC_2	**0.93**	1.95	1.29	0.95	3.16
GES	**0.93**	1.95	1.29	0.95	3.18

Cuadro 4.4: RMSE medio de la validación cruzada en la predicción de *Medida 2* para cada central y algoritmo de aprendizaje estructural

Algoritmo	Planta 1	Planta 2	Planta 3	Planta 4	Planta 5
baseline	-442.52	-485.05	-634.49	-719.32	**-557.54**
$K2_1$	-410.17	-455.48	-619.85	**-656.55**	-583.07
$K2_2$	**-410.10**	-466.8	-634.52	-663.84	-566.8
GHC_1	-416.39	-462.21	-636.11	-691.15	-573.39
GHC_2	-410.68	-449.91	-634.01	-694.01	-573.97
GES	-416.41	**-453.15**	**-616.88**	-660.34	-589.10

Cuadro 4.5: Media del logaritmo de la verosimilitud de los datos de test con el modelo entrenado

Como se puede observar, los resultados en cuanto a RMSE de los algoritmos de aprendizaje estructural son similares a los resultados obtenidos con la red definida por conocimiento experto, exceptuando los resultados obtenidos para la central *Planta 5*. En esta central, la red del modelo *baseline* obtiene considerablemente mejores resultados que el resto de alternativas propuestas.

Sin embargo, pese a que la métrica *RMSE* nos permite comparar de una forma muy intuitiva y sencilla los resultados obtenidos, ésta solo considera el valor medio de

la distribución condicional gaussiana que representa la predicción final. Por lo tanto, esta métrica no es óptima para comprobar cómo se ajusta un modelo a un conjunto de test. Esta es la razón por la que se hace uso de la medida del logaritmo de la verosimilitud.

Observando la tabla 4.5, se puede concluir que el *baseline*, en la mayoría de los casos, se ajusta peor a los datos de test que algunos de los modelos cuyas estructuras han sido aprendidas mediante aprendizaje estructural. De esta forma, los mejores resultados en cuanto a la verosimilitud son generalmente obtenidos por las estructuras generadas mediante $K2$, utilizando el orden definido en (1), y GES, exceptuando la última central, en la cual, el sistema *baseline* se ajusta mejor que el resto de modelos. En cuanto a GHC, exceptuando en la *Planta 2*, da lugar a los modelos que menos se ajustan a los datos de test.

Tras la realización de este experimento, y a la vista de los resultados obtenidos, se concluye que los modelos generados por los algoritmos de aprendizaje estructural probados tienen una capacidad de generalización similar al modelo definido por conocimiento experto.

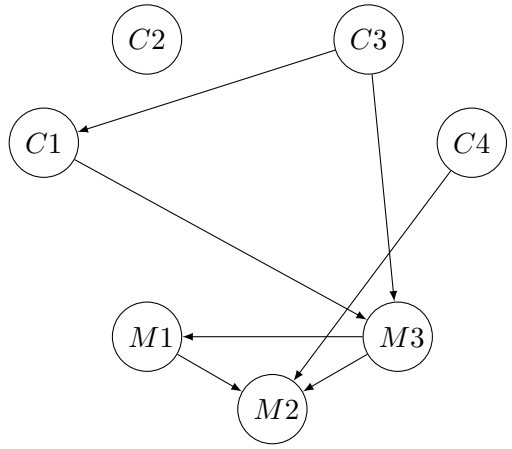
4.1.3. Generación de estructuras alternativas finales

Con el experimento anterior se comprueba la capacidad generalizadora de los modelos generados para la predicción de cada uno de los ciclos de cada una de las centrales de interés. Por lo tanto, para cada iteración del proceso de validación cruzada se generan estructuras diferentes, que en algunos casos, pertenecen a diferentes clases de I-equivalencia. La figura 4.2 representa un caso en el que esto ocurre.

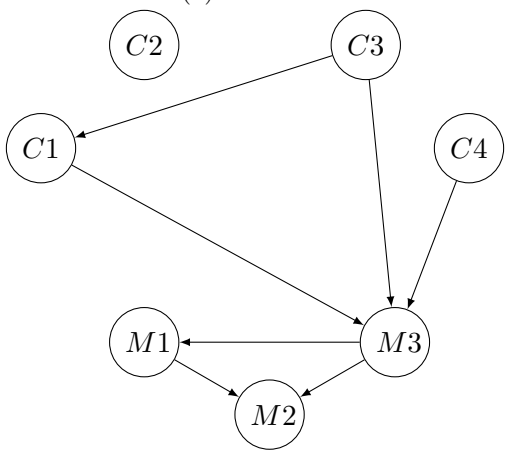
Por esta razón, se podría concluir que las relaciones entre variables varían en función del ciclo de trabajo en el que se capturan. Sin embargo, esta hipótesis es cuestionada por la empresa, que asegura que las relaciones entre las variables tienen que ser fijas para cada una de las centrales. Por lo que se concluye que, las diferencias existentes entre las estructuras de las centrales son debidas a:

1. Una cantidad insuficiente de datos de entrenamiento. El conjunto total de datos disponibles para cada una de las centrales viene determinado en la tabla 4.6. A la vista del número de datos y los resultados obtenidos en el primer experimento de esta tarea, estos datos son limitados para la reconstrucción de la estructura real de las variables.

2. Ruido estadístico presente en los datos proporcionados por la empresa.



(a)



(b)

Figura 4.2: Grafos entrenado mediante $K2_2$ (a) Predicción ciclo 1 (b) Predicción ciclo 2

	Planta 1	Planta 2	Planta 3	Planta 4	Planta 5
N	324	414	217	318	310

Cuadro 4.6: Conjunto total de muestras disponibles para cada central

3. Existencia de valores "atípicos" en la base de datos, producidos posiblemente por un error en la captura o por un acontecimiento inusual en la central.

Por esto, y con el objetivo de cerrar la tarea, se propone la generación de las estructuras finales con todos los datos de entrenamiento disponibles. Aunque el número de datos parece insuficiente, se espera la generación de estructuras más fiables que las generadas para cada uno de los ciclos en la tarea anterior. Para la generación de

estas estructuras se mantienen las transformaciones sobre los datos definidas en 3.2.

Para no sobrecargar esta sección con todas las estructuras generadas, se decide representar como ejemplo las estructuras generadas para las centrales *Planta 1* y *Planta 4* en las figuras 4.3 y 4.4 respectivamente. El resto de las estructuras pueden ser visualizadas en el Anexo C.

Analizando las estructuras generadas, son varias las conclusiones que se extraen. Primero, se observa que las estructuras generadas para cada una de las centrales son diferentes, dando lugar a estructuras que ni siquiera son I-equivalentes. Esto confirma la hipótesis inicial de la empresa, según la cual las relaciones entre las variables dependen de la central en la que se miden. Como excepción, el algoritmo *K2* da lugar a exactamente las mismas estructuras para las centrales *Planta 1* y *Planta 2*.

Por otra parte, se destaca que la variable *Control 2*, en ningún caso presenta ninguna dependencia con el resto de las variables del modelo. Lo contrario ocurre con el resto de variables *Control*, las cuales, generalmente están relacionadas por un *active trail*, o camino activo, con las variables *Medida*, siempre que estas no sean observadas. Dentro de estas variables de control, se destaca la relevancia de las variables *Control 1* y *Control 5*, que en todos los casos, presentan dependencias directas con una, o varias, variables *Medida*; y además, en el las centrales pertenecientes al *Grupo 1*, la variable *Control 3*.

Por otro lado, la estructura del *baseline* define las variables de control como independientes entre sí cuando no se conocen las variables *Medida*, y a su vez, define las variables *Medida* como independientes entre sí si se conocen las variables *Control*. Sin embargo, las estructuras aprendidas a partir de los datos revelan que, en todas las centrales, existen dependencias directas entre variables pertenecientes a la misma clase (*Control* y *Medida*).

Por último, tras el análisis de determinadas estructuras con la empresa, se determina que ciertas dependencias que establecen las variables *Medida* como nodos padre de las variables *Control* carecen de sentido químico o bien imposibilitan la correcta interpretabilidad de la red por parte de la empresa. Estos enlaces aparecen con la utilización de los algoritmos *Greedy Hill Climbing* y *GES*. Esto se debe en parte a que, los expertos de la empresa interpretan estas relaciones como relaciones de causalidad y no de relación estadística, ocasionando la divergencia entre el conocimiento experto y el aprendizaje estructural.

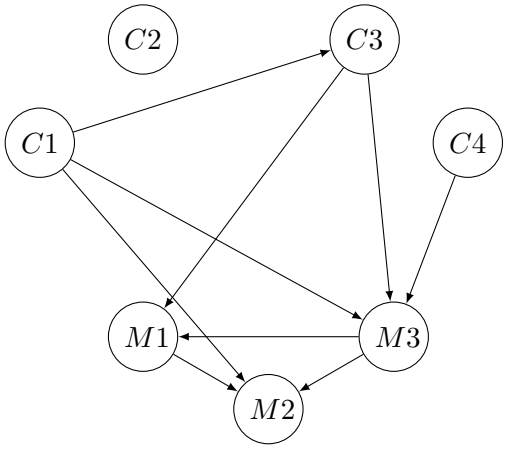
Software de laboratorio de aprendizaje estructural

Tras el análisis de las estructuras generadas, y para finalizar la tarea de aprendizaje estructural, se ha proporcionado a la empresa un software de laboratorio con el que realizar el aprendizaje estructural con todos los datos que ellos tengan disponibles fuera de la colaboración. El algoritmo seleccionado para esta herramienta ha sido el algoritmo K2. La selección de dicho algoritmo resulta razonable vistos los resultados obtenidos a lo largo de toda la tarea:

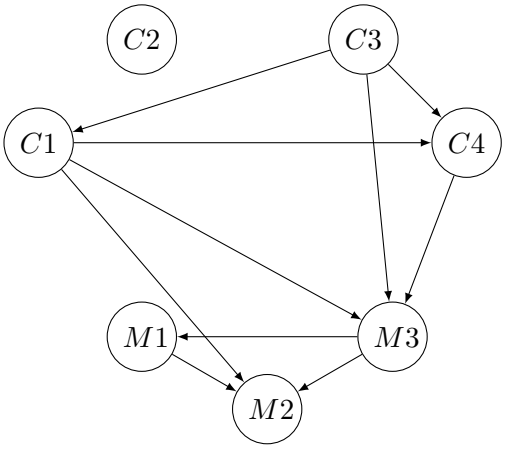
1. Es el algoritmo más rápido de todos los que han sido evaluados. El tiempo de ejecución más largo que se ha obtenido es de apenas 20 segundos para un dataset notablemente superior (5000 muestras) que el número de datos disponibles para la empresa.

2. Por otra parte, los modelos cuyas estructuras han sido aprendidas mediante *K2* han resultado obtener resultados en cuanto a verosimilitud de los datos de test, mejores o similares a GES.

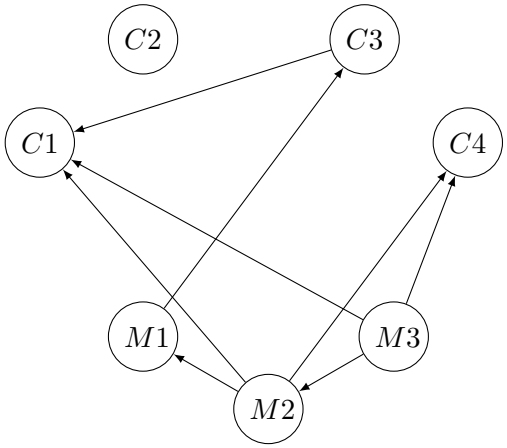
3. Por último, la introducción de un orden topológico de las variables, permite aprovechar el conocimiento experto de la empresa y así descartar posibles enlaces que químicamente o físicamente no tienen sentido.



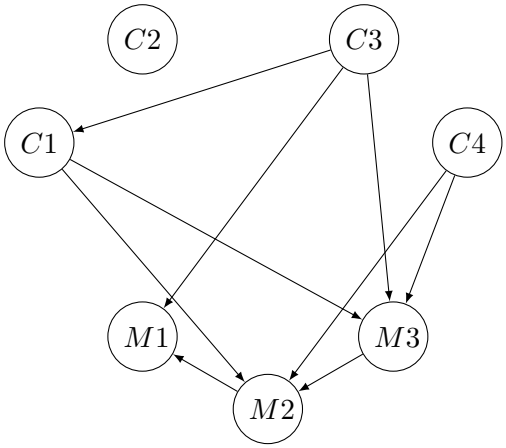
(a) $K2_1$



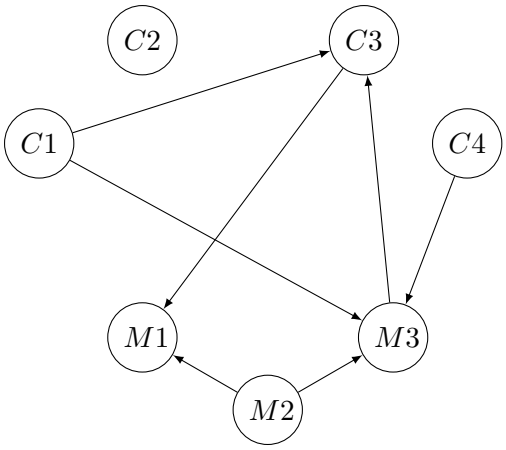
(b) $K2_2$



(c) GHC_1



(d) GHC_2



(e) *GES*

Figura 4.3: Estructuras obtenidas para la central *Planta 1* utilizando los diferentes algoritmos

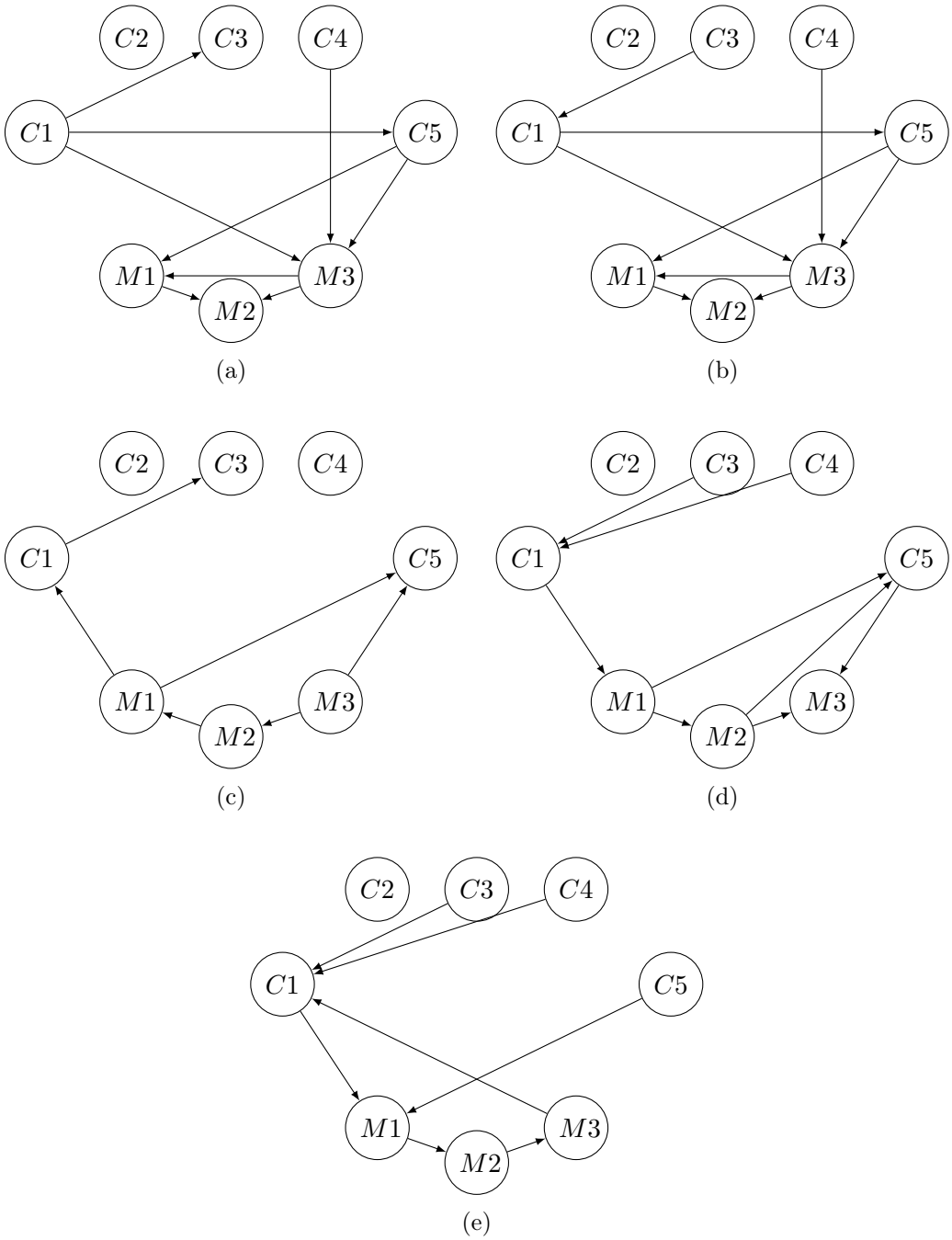


Figura 4.4: Estructuras obtenidas para la central *Planta 4* utilizando los diferentes algoritmos

4.2. Tarea 2: Generación de un modelo *fully Bayesian*

Hasta ahora, en el contexto de la colaboración del grupo AUDIAS con la empresa proveedora de centrales eléctricas, todos los modelos en los que se ha profundizado son modelos entrenados mediante estimadores, en concreto, el entrenamiento de todos los modelos ha sido realizado mediante el estimador de máxima verosimilitud.

La principal desventaja de esta técnica, como se ha introducido en 2.4.1, es que, ante la escasez de datos, la capacidad de generalización de los modelos entrenados mediante *MLE* es baja, produciéndose normalmente un sobreajuste u *overfitting* del modelo a los datos de entrenamiento.

Por lo tanto, al considerarse el entorno de experimentación de la colaboración un escenario desafiante, caracterizado por tener un número de datos limitado, ser una tarea en la que el control de la incertidumbre es crítico, resulta lógico considerar como el siguiente paso de esta colaboración el tratamiento completamente bayesiano de los modelos previamente implementados. Aunque este proceso queda fuera del alcance de este trabajo, se propone como tarea introductoria, la implementación de un modelo completamente bayesiano gaussiano de los datos proporcionados por la empresa. De esta forma, el objetivo principal de esta tarea es la comparación de los resultados en la predicción de las variables *Medida* a partir de las variables *Control*, con el modelo completamente bayesiano, y modelos paramétricos gaussianos cuyos parámetros son entrenados mediante *MLE* y *MAP*.

El proceso de evaluación con el que se realiza esta comparación es similar al descrito en la sección 4.1.2. Según el cual, se realiza un *K-Fold Cross Validation*, donde K es el número de ciclos de cada central y , por lo tanto, en cada iteración de la validación cruzada se realiza la predicción de un ciclo diferente. Sin embargo, en este caso, el tamaño del conjunto de entrenamiento N, utilizado en cada iteración de la validación cruzada, no será el conjunto total de datos de los ciclos de entrenamiento, sino que será variable. De esta forma, se evaluará el rendimiento de los modelos paramétricos en función del tamaño del conjunto de entrenamiento.

Este conjunto de entrenamiento de N muestras se obtendrá mediante la aleatorización de todos los datos de los ciclos disponibles, y tras la aplicación de las transformaciones descritas en 3.2. Adicionalmente, con el objetivo de representar mejor el comportamiento del modelo completamente bayesiano, y al ya carecer de interés para la empresa, se decide extraer de las variables de interés, la variable *Medida 3*.

Por lo tanto, el conjunto de variables que van a ser modeladas es:

$\hat{x} = \{Control1, Control2, Control3, Control4, Control5, Medida1, Medida2\}$

Así pues, mediante el estimador de máxima verosimilitud, el modelo obtenido quedaría definido tal que:

$$p(\hat{x}|\mu_{MLE}, \Sigma_{MLE}, \mathcal{D}) = \mathcal{N}(\hat{x}|\mu_{MLE}, \Sigma_{MLE}) \tag{4.3}$$

donde los parámetros son calculados mediante las ecuaciones 2.14.

Para la generación del modelo completamente bayesiano y para la estimación de los parámetros mediante *MAP*, tal y como se introduce en la sección 2.3.3, se selecciona como distribución *prior* de los parámetros del modelo gaussiano (μ y Λ) una distribución normal-Wishart. En concreto, para esta tarea se ha decidido la utilización de un *prior* poco informativo, seleccionando los hiperparámetros, descritos en [8]:

$$k_0 = 0;\ v_0 = -1;\ |T_0| = 0;\ \mu_0 = 0; \tag{4.4}$$

De esta forma, se obtiene el *posterior* definido como:

$$p(\mu, \Lambda) = \mathcal{N}Wi(\mu, \Lambda|\mu_n, k_n, v_n, T_n) \tag{4.5}$$

donde

$$\mu_n = \frac{1}{N}\sum_{n=1}^{N} x_n; \quad v_n = n - 1 \quad ; T_n = S \quad ; k_n = n \tag{4.6}$$

Tras definir la distribución *posterior* de los parámetros, y, de la misma forma que para *MLE*, el modelo obtenido mediante *MAP* es definido tal que:

$$p(\hat{x}|\mu_{MAP}, \Sigma_{MAP}, \mathcal{D}) = \mathcal{N}(\hat{x}|\mu_{MAP}, \Sigma_{MAP}) \tag{4.7}$$

donde los parámetros son obtenidos sustituyendo 4.4 en las ecuaciones 2.18.

Por su parte, mediante un tratamiento completamente bayesiano del modelo, tras la marginalización descrita en la ecuación 2.29, se obtiene la siguiente distribución t de Student predictiva:

$$p(\hat{x}|\mathcal{D}) = t_{n-d}\left(\mu_n, \frac{S(n+1)}{n(n-d)}\right) \tag{4.8}$$

De esta forma, quedan definidos y entrenados los tres modelos que se van a comparar.

La comparativa de los tres modelos se realiza en términos de la predicción de las variables $\hat{x_1}$ = { *Medida 1, Medida 2* } a partir de $\hat{x_2}$ = { *Control 1, Control 2, Control 3, Control 4, Control 5* }. Para ello, adicionalmente a las tablas 4.7 - 4.11, que representan el valor medio del *log likelihood* de las variables $\hat{x_1}$ con respecto al modelo, se proponen dos representaciones diferentes:

1. Representación de la distribución de probabilidad condicionada de las variables *Medida* con respecto a las variables *control*, $P(\hat{x_1}|\hat{x_2} = a)$, obtenida para la predicción a partir de una muestra a del conjunto de test, para cada uno de los modelos. Adicionalmente, para una mejor comparativa de los modelos, se adjuntan sus curvas de contorno y sus curvas de contorno en escala logarítmica. Estas figuras son numeradas como 4.5, 4.6, 4.7, 4.9, 4.10 y 4.11.

2. Representación de las distribuciones $P(Medida1|\hat{x_2})$ y $P(Medida2|\hat{x_2})$, obtenidas tras la marginalización de la distribución anterior. Estas distribuciones son las predicciones finales del modelo para cada una de las variables *Medida*. Estas son las figuras 4.8 y 4.9.

Remarcar que, en este caso, los valores medios predichos por los tres modelos coinciden, por lo que, se excluye la métrica *RMSE* para la comparación de los modelos.

De los resultados obtenidos se extraen las siguientes conclusiones:

Primero, a la vista de los resultados de las tablas 4.7 a 4.11. Se comprueba que, para todas las centrales de interés, el modelo *fully bayesian* (denominado "FBM") tiene una mayor capacidad de generalización que el resto de modelos, pues obtiene los valores más altos de *log likelihood*.

Por otra parte, se comprueba que la diferencia existente entre los resultados obtenidos por los modelos entrenados con estimadores, y los obtenidos por el modelo *fully bayesian*, es mayor cuanto menor es el tamaño del conjunto de entrenamiento.

Adicionalmente, estos resultados permiten comprobar lo previamente estudiado referente a los modelos completamente bayesianos.

De esta forma, analizando las figuras 4.5 a 4.7, se comprueba la diferencia existente entre las distribuciones condicionadas $P(\hat{x_1}|\hat{x_2})$ obtenidas con los modelos basados en estimadores y el modelo completamente bayesiano, para un conjunto muy limitado de entrenamiento, $N = 10$. A partir de las curvas de contorno, se constata que el modelo completamente bayesiano genera una distribución que decrece de forma menos abrupta que los modelos paramétricos gaussianos basados en estimadores. Esto provoca que los estimadores de parámetros, y especialmente, *MLE*, sobreajusten el

modelo a los datos de entrenamiento. Sin embargo, tal como se puede ver en la figura 4.8 (b), la incorporación de la incertidumbre en los parámetros reduce la probabilidad de que un valor, no presente en el conjunto de entrenamiento, tenga una densidad de probabilidad nula. De esta forma, se genera un modelo con un margen de credibilidad mayor que el generado mediante *MLE* y, por lo tanto, más robusto frente a la escasez de datos.

Por otro lado, a medida que el conjunto de entrenamiento aumenta en tamaño, la verosimilitud del modelo con los datos aumenta con respecto al conocimiento *prior*. Esto provoca que, ante la presencia de un conjunto de entrenamiento más grande, el comportamiento de los modelos *fully bayesian* sea similar al de los modelos entrenados con *MLE*. Lo mismo ocurre con los modelos entrenados con *MAP*. Como prueba de ello, se repiten las representaciones anteriores, para un tamaño del conjunto de entrenamiento $N = 200$. De esta forma, tal como se aprecia en las figuras 4.9, 4.10, 4.11 y 4.12, con un conjunto de entrenamiento de este tamaño, las distribuciones finales obtenidas por los tres modelos son prácticamente iguales.

Así pues, con los resultados y observaciones presentados anteriormente, se finaliza la segunda tarea de este trabajo, cumpliendo los objetivos previamente establecidos, y que abre una posible vía de investigación relacionada con el tratamiento completamente bayesiano de problemas en el ámbito en el que se desarrolla este TFM.

Modelo	N=10	N=20	N=50	N=100	N=150	N=N_{MAX}
MLE	-588.1	-181.2	-178.4	-182.9	-179.8	-176.48
MAP	-290.7	-171.7	-173.7	-180.5	-178.3	-175.9
FBM	**-238.9**	**-170.1**	**-172.7**	**-179.6**	**-177.6**	**-175.7**

Cuadro 4.7: Media del logaritmo de la verosimilitud de los datos de test para *Planta1*

Modelo	N=10	N=20	N=50	N=100	N=150	N=N_{MAX}
MLE	-639.1	-219.3	-183.7	-183.8	-187.2	-191.6
MAP	-285.8	-197.9	-179.1	-182.1	-186.1	-190.95
FBM	**-240.5**	**-192.7**	**-179.0**	**-182.1**	**-185.9**	**-190.84**

Cuadro 4.8: Media del logaritmo de la verosimilitud de los datos de test para *Planta2*

Modelo	N=10	N=20	N=50	N=100	N=N_{MAX}
MLE	-1659.1	-407.9	-223.9	-218.1	-194.12
MAP	-410.0	-291.49	-209.67	-211.28	-190.0
FBM	**-318.1**	**-243.5**	**-201.5**	**-206.1**	**-186.9**

Cuadro 4.9: Media del logaritmo de la verosimilitud de los datos de test para *Planta3*

Modelo	N=10	N=20	N=50	N=100	N=N_{MAX}
MLE	-8054.8	-345.8	-194.1	-199.7	-213.0
MAP	-1838.1	-258.75	-185.9	-195.19	-209.4
FBM	**-472.7**	**-236.95**	**-179.7**	**-189.4**	**-202.5**

Cuadro 4.10: Media del logaritmo de la verosimilitud de los datos de test para *Planta4*

Modelo	N=10	N=20	N=50	N=100	N=N_{MAX}
MLE	-1235.7	-203.5	-191.8	-173.9	-179.3
MAP	-345.0	-176.9	-181.8	-176.6	-176.3
FBM	**-277.6**	**-170.9**	**-174.2**	**-167.3**	**-173.1**

Cuadro 4.11: Media del logaritmo de la verosimilitud de los datos de test para *Planta5*

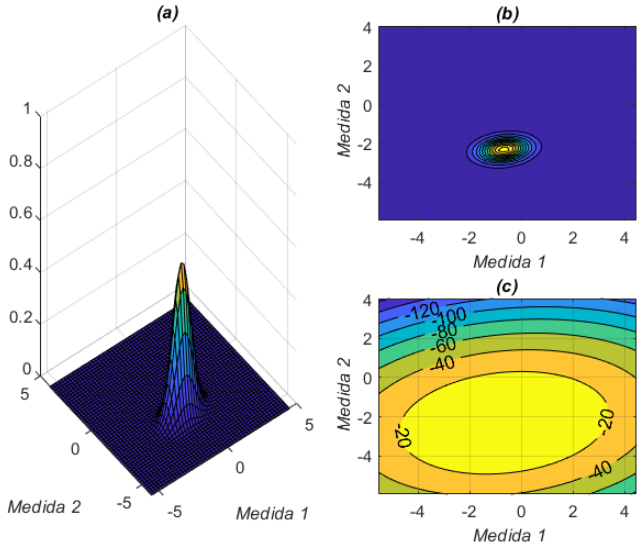


Figura 4.5: Ejemplo de distribución $P(\hat{x_1}|\hat{x_2})$ (a), su contorno (b) y su contorno en escala logarítmica (c), para la predicción de las variables *Medida* en la central *Planta 1*, por el modelo entrenado mediante MLE y $N = 10$

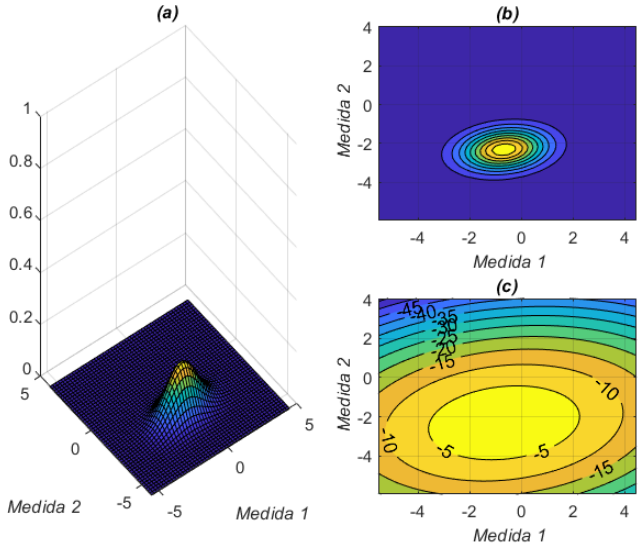


Figura 4.6: Ejemplo de distribución $P(\hat{x_1}|\hat{x_2})$ (a), su contorno (b) y su contorno en escala logarítmica (c), para la predicción de las variables *Medida* en la central *Planta 1*, por el modelo entrenado mediante MAP y $N = 10$

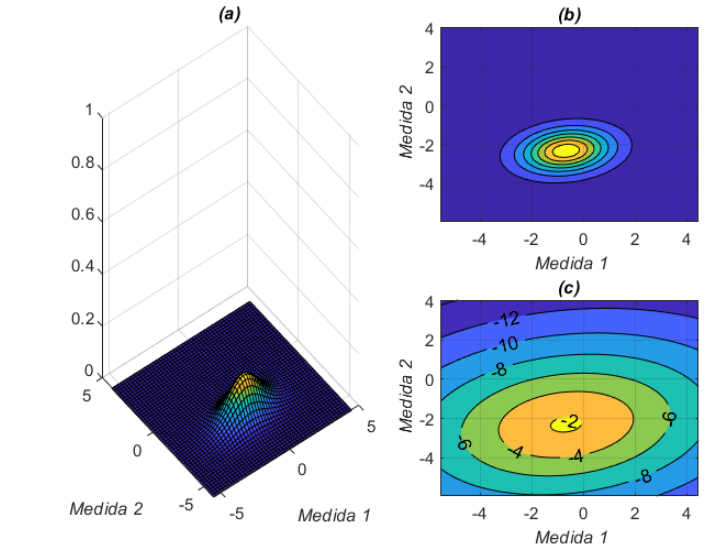


Figura 4.7: Ejemplo de distribución $P(\hat{x_1}|\hat{x_2})$ (a), su contorno (b) y su contorno en escala logarítmica (c), para la predicción de las variables *Medida* en la central *Planta 1*, por el modelo entrenado *fully bayesian* y $N = 10$

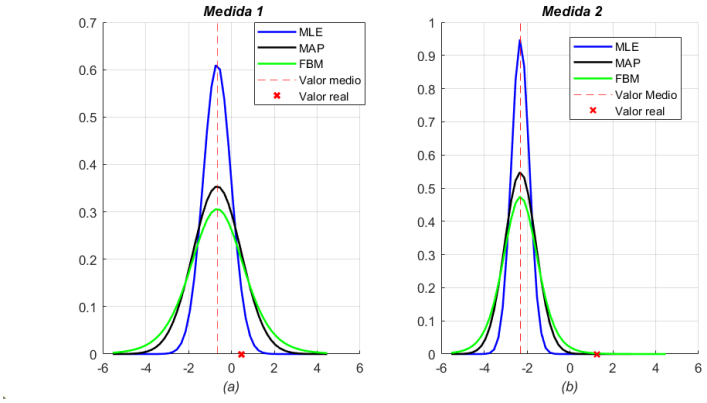


Figura 4.8: Ejemplo de distribución $P(\hat{x_1}|\hat{x_2})$ (a), su contorno (b) y su contorno en escala logarítmica (c), para la predicción de las variables *Medida* en la central *Planta 1*, por el modelo entrenado *fully bayesian* y $N = 10$

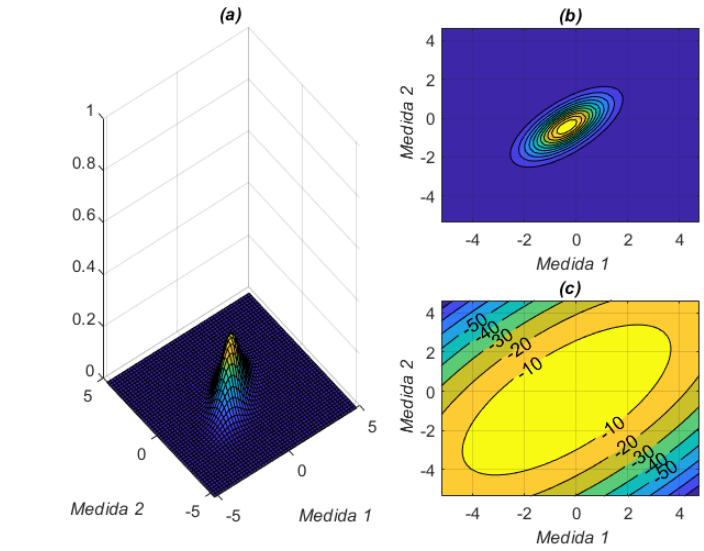


Figura 4.9: Ejemplo de distribución $P(\hat{x_1}|\hat{x_2})$ (a), su contorno (b) y su contorno en escala logarítmica (c), para la predicción de las variables *Medida* en la central *Planta 1*, por el modelo entrenado mediante MLE y $N = 200$

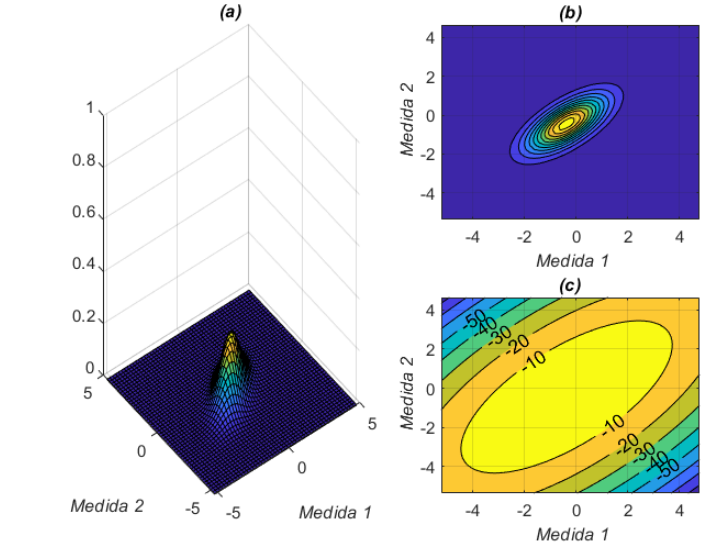


Figura 4.10: Ejemplo de distribución $P(\hat{x_1}|\hat{x_2})$ (a), su contorno (b) y su contorno en escala logarítmica (c), para la predicción de las variables *Medida* en la central *Planta 1*, por el modelo entrenado mediante MAP y $N = 200$

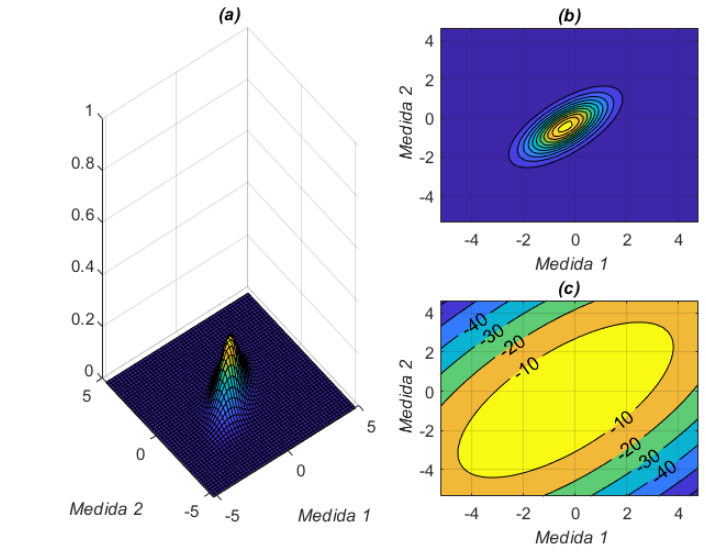


Figura 4.11: Ejemplo de distribución $P(\hat{x_1}|\hat{x_2})$ (a), su contorno (b) y su contorno en escala logarítmica (c), para la predicción de las variables *Medida* en la central *Planta 1*, por el modelo entrenado *fully bayesian* y $N = 200$

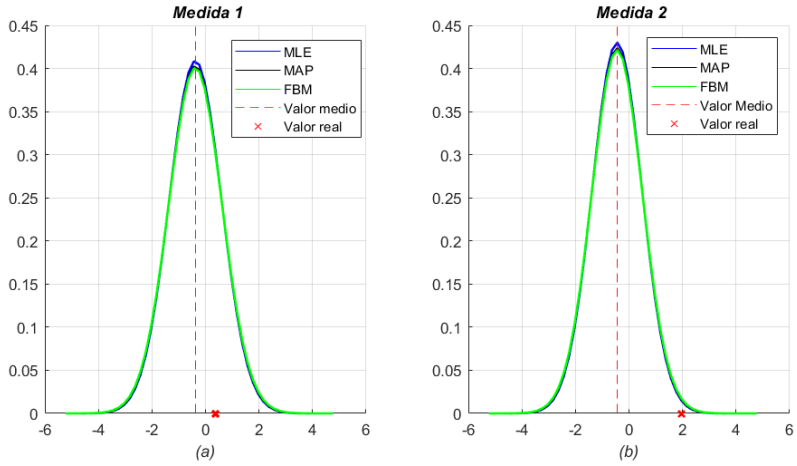


Figura 4.12: Ejemplo de distribución $P(\hat{x_1}|\hat{x_2})$ (a), su contorno (b) y su contorno en escala logarítmica (c), para la predicción de las variables *Medida* en la central *Planta 1*, por el modelo entrenado *fully bayesian* y $N = 200$

Capítulo 5

Conclusiones

Este Trabajo de Fin de Máster presenta dos objetivos claramente diferenciados, que se han traducido en la realización de dos tareas independientes.

Por un lado, con el objetivo de descubrir las relaciones existentes entre las variables de interés para la empresa, se ha realizado un análisis de diferentes algoritmos de aprendizaje estructural de redes bayesianas, basados en puntuación. Los resultados obtenidos en esta tarea han permitido a la empresa aumentar y consolidar su conocimiento, respecto a las relaciones existentes entre las variables presentes en el proceso de generación de electricidad. Adicionalmente, esta tarea ha concluido con el aprendizaje de estructuras que, en la mayoría de los casos, han demostrado representar mejor los datos disponibles que la red generada por conocimiento experto, manteniendo a su vez una capacidad de generalización similar a la del sistema *baseline*. Para finalizar, además de proporcionar a la empresa las estructuras finales obtenidas, se ha proporcionado un software de laboratorio que realiza el aprendizaje estructural mediante K2 para cualquier variable de interés.

Por otro lado, se ha implementado un modelo completamente bayesiano de las variables de interés, con el objetivo de comprobar si la cuantificación de la incertidumbre presente en los parámetros permite una mejor descripción de los datos por parte del modelo y, por lo tanto, una mejor predicción de las variables *Medida* a partir de las variables *Control*. Los resultados de esta tarea han sido satisfactorios, pues se ha demostrado que el tratamiento completamente bayesiano proporciona una mayor robustez ante la escasez de datos. Y, por lo tanto, abre posibles líneas de investigación en colaboración con la empresa, para profundizar en estos algoritmos y adaptarlos al contexto del proyecto.

De esta forma, se considera que ambos objetivos han sido cumplidos y, como principal trabajo futuro, se propone la investigación del tratamiento completamente bayesiano del modelo definido por la red bayesiana del sistema *baseline*. Adicionalmente, ante la escasez de paquetes de Python semejantes a BNT, se propone el desarrollo en Python de una herramienta que permita realizar el entrenamiento de parámetros y estructuras de redes bayesianas gaussianas, y la utilización de algoritmos de inferencia exacta.

Bibliografía

[1] P. Ramirez Hereza tech. rep.

[2] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[3] W. M. Bolstad and J. M. Curran, *Introduction to Bayesian statistics*. John Wiley & Sons, 2016.

[4] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[5] D. Barber, *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

[6] M. I. Jordan *et al.*, "Graphical models," *Statistical Science*, 2004.

[7] M. L. Eaton, "Multivariate statistics: a vector space approach.," *Jhon Wiley & Sons, Inc*, 1983.

[8] K. P. Murphy, "Conjugate bayesian analysis of the gaussian distribution," *def*, vol. 1, no. $2\sigma 2$, p. 16, 2007.

[9] M. H. DeGroot, *Optimal statistical decisions*, vol. 82. John Wiley & Sons, 2005.

[10] P. Ding, "On the conditional distribution of the multivariate t distribution," *The American Statistician*, vol. 70, no. 3, pp. 293–295, 2016.

[11] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[12] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, "The max-min hill-climbing bayesian network structure learning algorithm," *Machine learning*, vol. 65, no. 1, pp. 31–78, 2006.

[13] G. F. Cooper and E. Herskovits, "A bayesian method for the induction of probabilistic networks from data," *Machine learning*, vol. 9, no. 4, pp. 309–347, 1992.

[14] D. M. Chickering, "Optimal structure identification with greedy search," *Journal of machine learning research*, vol. 3, no. Nov, pp. 507–554, 2002.

[15] K. Murphy *et al.*, "The bayes net toolbox for matlab," *Computing science and statistics*, 2001.

[16] P. Leray and O. Francois, "Bnt structure learning package: Documentation and experiments," *Laboratoire PSI, Universitè et INSA de Rouen, Tech. Rep*, 2004.

[17] S. S. Chen and R. A. Gopinath, "Gaussianization," in *Advances in neural information processing systems*, pp. 423–429, 2001.

[18] G. Saon, S. Dharanipragada, and D. Povey, "Feature space gaussianization," in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. I–329, IEEE, 2004.

Apéndice A

Teoría de la probabilidad

En este anexo se proporciona un breve repaso de las reglas fundamentales de la teoría de la probabilidad [2][3], base de los modelos probabilísticos presentados en el trabajo.

Dadas las variables aleatorias $A = \{a_1, a_2, ..., a_N\}$ y $B = \{b_1, b_2, ..., b_M\}$:

1. Definimos la **regla de la suma** como:

$$P(X) + P(\bar{X}) = 1 \tag{A.1}$$

2. Definimos la **probabilidad conjunta** de ambos eventos mediante la que es denominada la **regla del producto** como:

$$P(A, B) = P(A|B)P(B). \tag{A.2}$$

Dadas las dos reglas fundamentales de la teoría de la probabilidad podemos definir:

1. **Probabilidad marginal** de A, en ocasiones presentada como la regla de la suma:

$$P(A) = \sum_{b} P(A, B) = \sum_{i}^{M} P(A|B = b_i)P(B = b_i) \tag{A.3}$$

2. **Teorema de Bayes**: A partir de la regla del producto y de la propiedad de simetría $P(X, Y) = P(Y, X)$ obtenemos directamente la relación:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{A.4}$$

3. **Regla de la cadena**: Podemos definir la probabilidad conjunta de múltiples variables como:

$$P(X_1, X_2, .., X_R) = P(X_1)P(X_2|X_1)P(X_3|X_2, X_1)...P(X_R|X_1, X_2, ..., X_3) \tag{A.5}$$

Independencia marginal e independencia condicional Dadas dos variables aleatorias $A = \{a_1, a_2, ..., a_N\}$ y $B = \{b_1, b_2, ..., b_M\}$:

Se dice que A *es independiente marginalmente de* B si se cumple:

$$P(A|B) = P(A) \tag{A.6}$$

Sustituyendo en (2.2), obtenemos:

$$P(A, B) = P(A)P(B) \tag{A.7}$$

Dada una tercera variable aleatoria $Z = \{z_1, z_2, ..., z_O\}$. Se dice que A *es independiente de* B *conociendo* Z, es decir condicionalmente independiente, si se cumple que:

$$P(A|B, Z) = P(A|Z) \tag{A.8}$$

Sustituyendo en (2.2), obtenemos:

$$P(A, B|Z) = P(A|Z)P(B|Z) \tag{A.9}$$

Apéndice B

Gaussianización de datos

Este anexo proporciona una descripción del proceso de gaussianización utilizado sobre las variables de interés, para una mejor descripción de los datos a partir de los modelos probabilísticos gaussianos estudiados.

Gaussianización basada en ecualización de histogramas

Tal y como se describe en [17] [18], sea un vector X de variables aleatorias con una función de distribución de probabilidad conjunta $f(X)$. Suponemos que cada variable aleatoria x_i tiene una función de distribucion $f(x_i)$ y una correspondiente función de distribución acumulada o CDF (*Cummulative Distribution Function*) $F(x_i)$. Se supone además que $\phi(.)$ es la cdf de de una variable Gaussiana unidimensional de media cero y varianza unidad, tal que:

$$\phi(\epsilon) = \int_{-\infty}^{\epsilon} \frac{1}{\sqrt{2\pi}} \exp -\frac{\alpha^2}{2} d\alpha \tag{B.1}$$

se puede demostrar que $Y = \phi^{-1}(F(x_i))$ es una variable aleatoria de media cero y varianza unidad. Siendo $\phi^{-}1$ la función inversa de la normal, también denominada *probit*. Mediante la gaussianización de las variables con media y varianza unidad, obtenemos la gaussianización total del conjunto de variables.

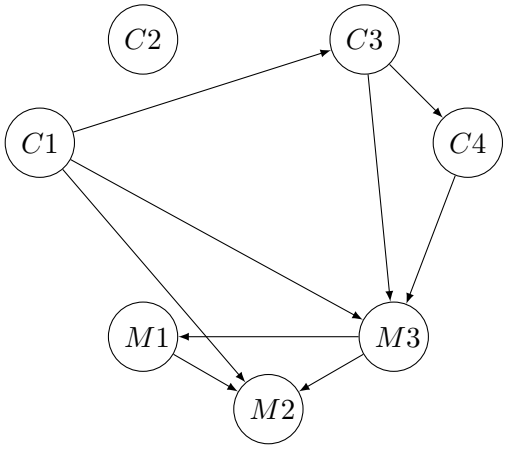
Apéndice C

Estructuras finales aprendidas

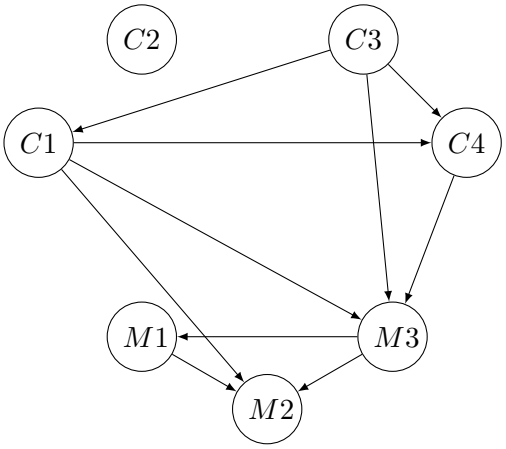
En este anexo se presentan las estructuras finales aprendidas a partir de los datos proporcionados por la empresa. Estas estructuras corresponden a los resultados obtenidos en la sección 4.1.3 de este documento.

En estas estructuras, las variables *Control* aparecen representadas por los nodos Cx y las variables *Medida* por los nodos Mx.

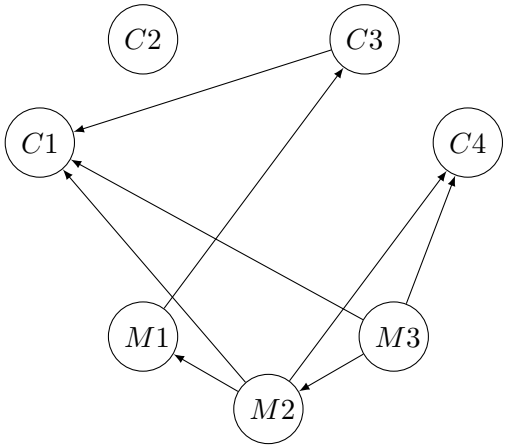
De esta forma, cada página corresponde a las estructuras generadas para cada central de interés. Éstas han sido generadas mediante el uso de las técnicas de aprendizaje estructural: *K2*, *Greedy Hill Climbing* nombrado en este documento como *GHC* y *Greedy Equivalence Search* o GES.



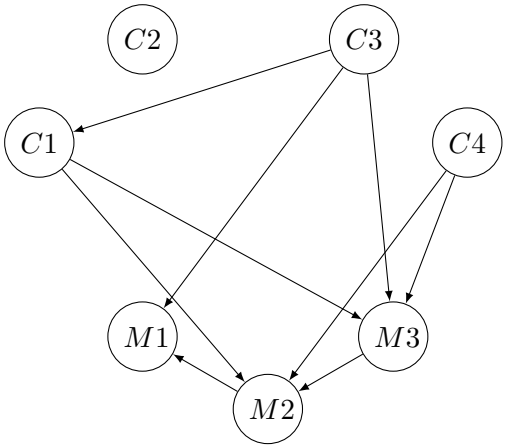
(a) $K2_1$



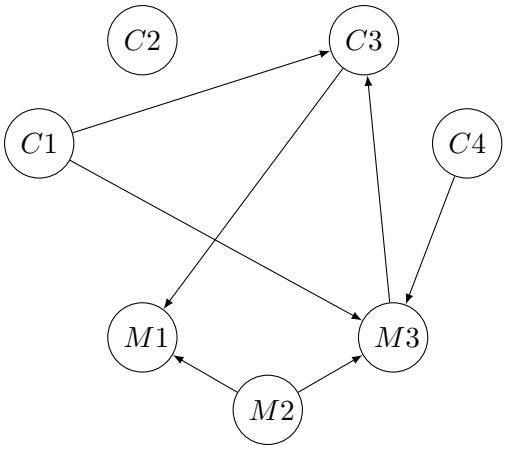
(b) $K2_2$



(c) GHC_1

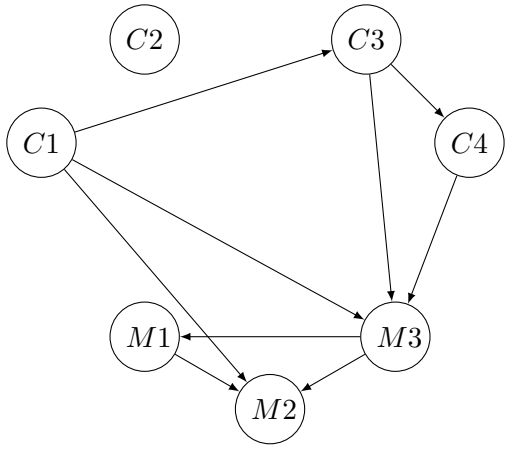


(d) GHC_2

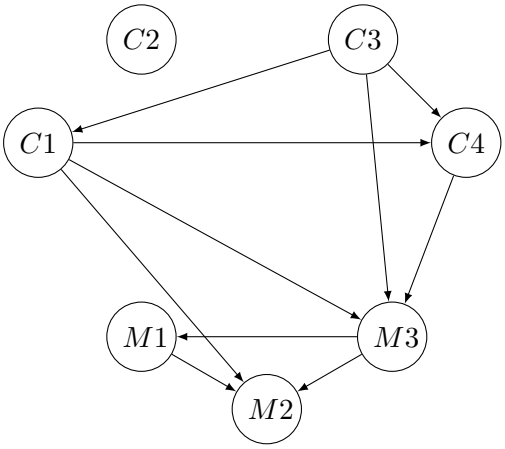


(e) *GES*

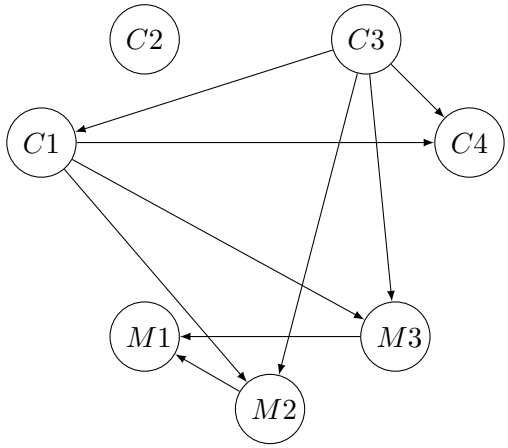
Figura C.1: Estructuras obtenidas para la central *Planta 1* utilizando los diferentes algoritmos



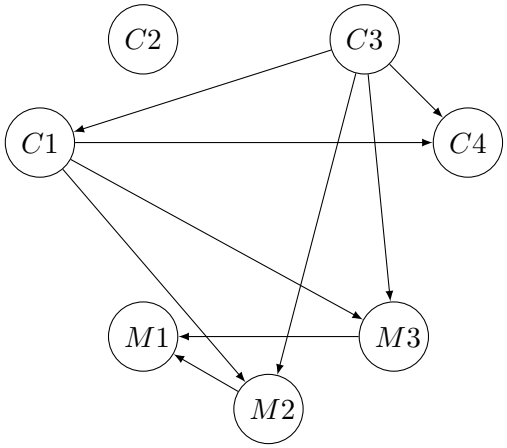
(a) $K2_1$



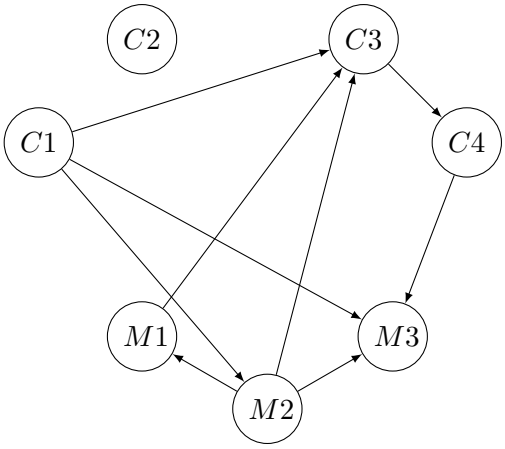
(b) $K2_2$



(c) GHC_1



(d) GHC_2



(e) *GES*

Figura C.2: Estructuras obtenidas para la central *Planta 2* utilizando los diferentes algoritmos

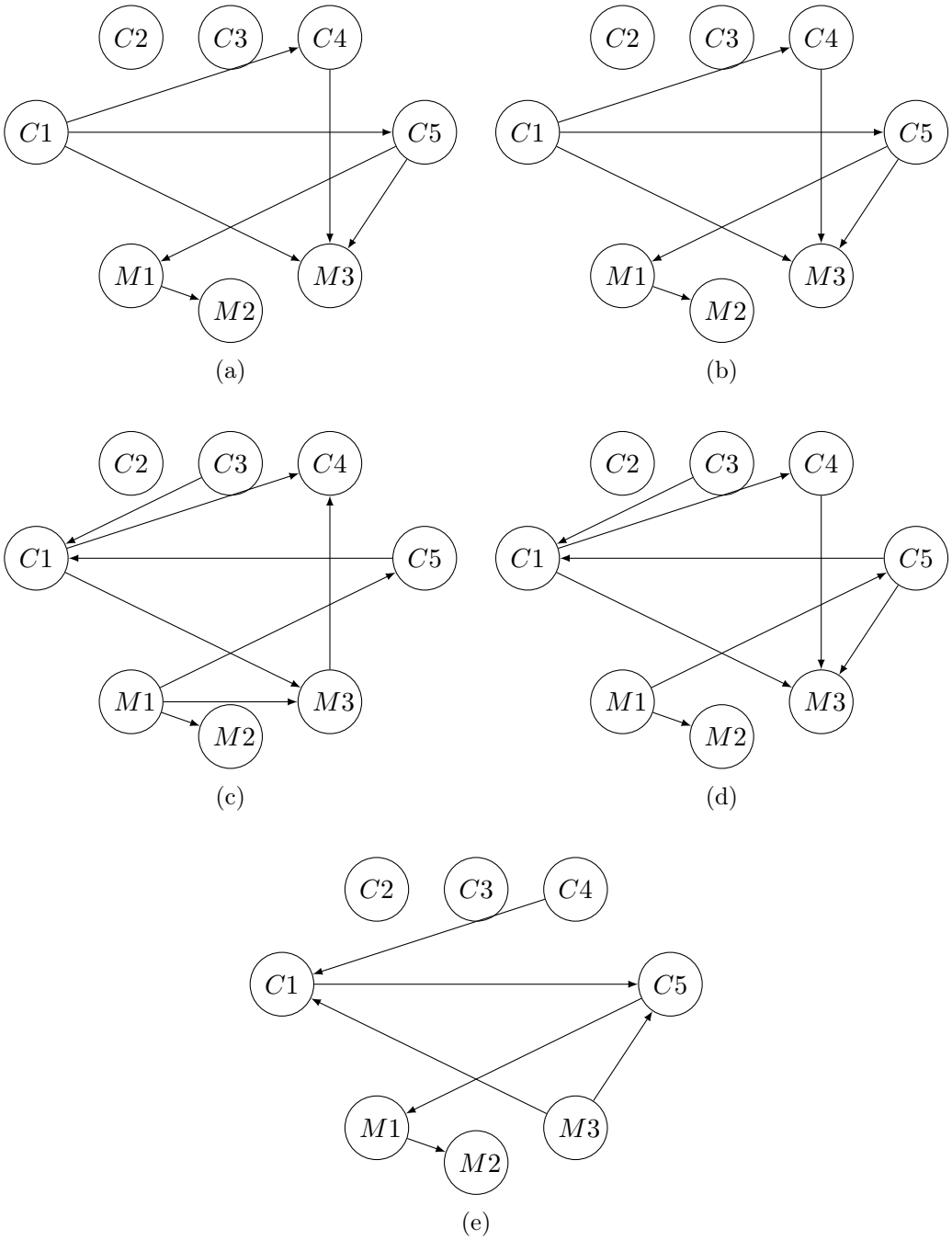


Figura C.3: Estructuras obtenidas para la central *Planta 3* utilizando los diferentes algoritmos

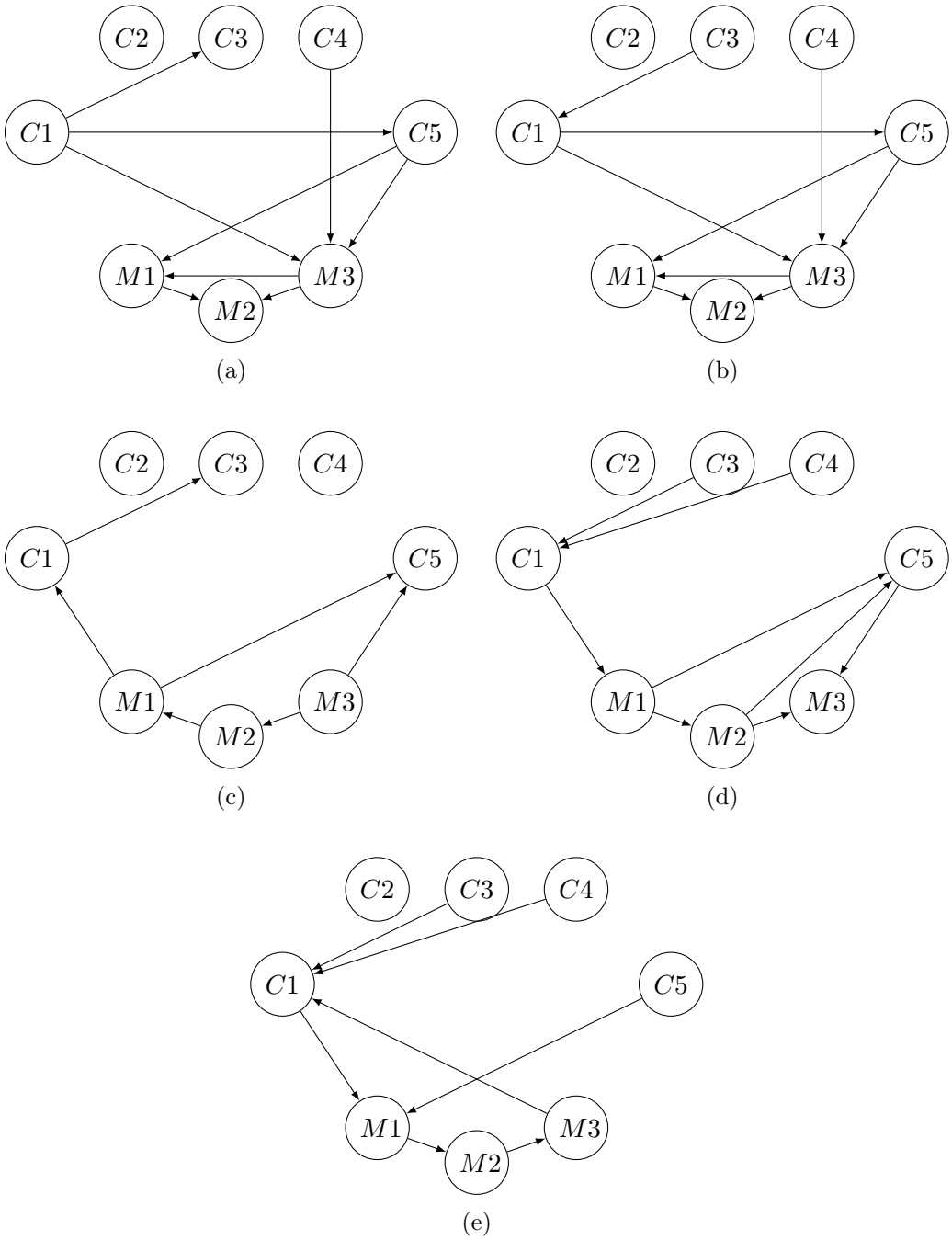


Figura C.4: Estructuras obtenidas para la central *Planta 4* utilizando los diferentes algoritmos

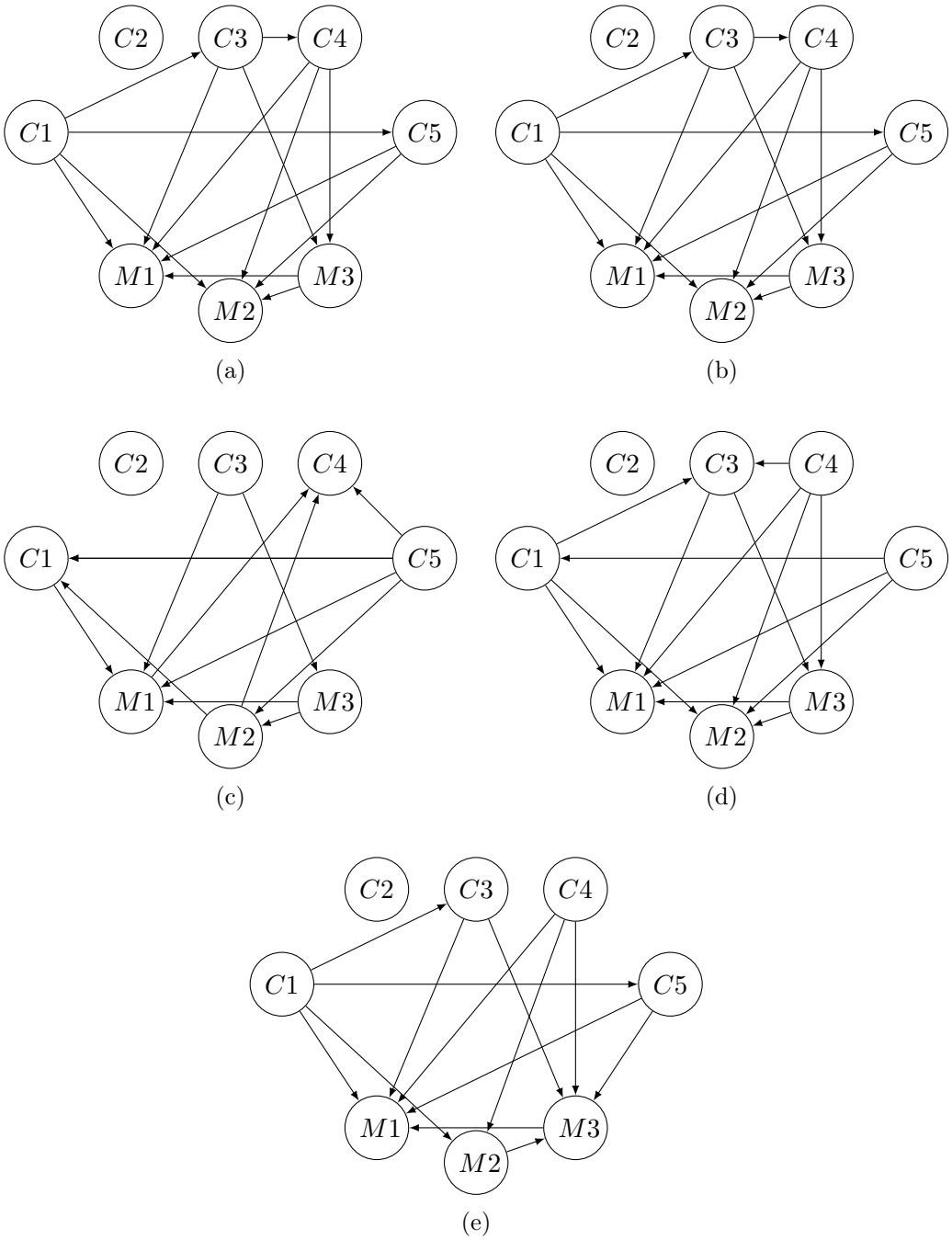


Figura C.5: Estructuras obtenidas para la central *Planta 5* utilizando los diferentes algoritmos