INSTITUTO POLITECNICO NACIONAL

CENTRO DE INVESTIGACION EN COMPUTACION

HERRAMIENTA PARA LA EXTRACCION DE REDES BAYESIANAS PREDICTIVAS A PARTIR DE BASES DE DATOS TEMPORALES

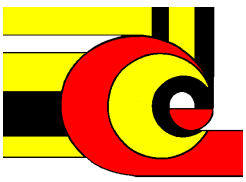
T E S I S

QUE PARA OBTENER EL GRADO DE
MAESTRO EN CIENCIAS DE LA COMPUTACION

P R E S E N T A

ING. JUAN MANUEL MEDINA APODACA

DIRECTOR DE TESIS
DR. JESUS FIGUEROA NAZUNO



MEXICO D. F.

INDICE

ABSTRACT. This thesis presents a tool for the extraction of graph-type models from data, where input data consists of a set of time series. The constructed model expresses each time series as a node, and direct relations between them as directed arcs in the graph. Besides the graphical result, the model holds information about the behavior of each relation in the form of a conditional probability density.

The kind of model obtained is commonly known as a Bayesian Network, and has been widely studied in the field of Machine Learning. Traditionally, Bayesian Networks learning techniques have been developed aiming to their use on discrete data, forcing the discretization of continuous time series. As the correct discretization of data is an important problem, a new time series discretization method was developed in order to represent as much information as possible, considering both its amplitude and its variations. This discretization method requires a parameter (σ) which specifies the importance given to the slope at each point in the time series. Distinct values for σ produce different discrete sequences. When trying to discover relations among time series, it's desirable to use those values of σ which produce the greatest coincidences among them

Some time is expected to elapse between the occurrence of an event and its corresponding effect. This is reflected as time delays between the time series which represent such events. Discretized time series must be aligned in order to discover the correct delay between them.

As distinct values of σ may produce distinct time delays, these parameters must be studied as a whole. A simulated annealing algorithm has been applied to find the best configuration for this values. This process facilitates the discovery of subtle relations between pairs of time series. The Bayesian Network was extracted from those aligned discrete sequences using well known learning algorithms.

The traditional Bayesian Networks model was extended in order to represent time delays between pairs of nodes. This way, every arc linking two nodes has an associated integer number, obtained from the alignment of the underlying time series, which represents the difference in time between the occurrence of an event in the "cause node" and the occurrence of the corresponding event in the "consequence node".

Finally, some time series can be recovered from the Bayesian Network for evaluation and short-term prediction proposes. Known values must be given to some other nodes in order to get the probability density in the target node and sample over it.

The tool has been implemented as a distributed environment, allowing for collaboration among users at different places. The distribution was achieved using CORBA objects.

RESUMEN. Esta tesis presenta una herramienta para la extracción automática de modelos de tipo grafo a partir de datos, en donde los datos de entrada consisten en un conjunto de series de tiempo. El modelo construido expresa cada serie de tiempo como un nodo, y las relaciones directas entre ellas como arcos dirigidos en el grafo. Además del resultado gráfico, el modelo mantiene información acerca del comportamiento de cada relación en forma de una densidad de probabilidad condicional.

El tipo de modelo obtenido se conoce comúnmente como Red Bayesiana, y ha sido estudiado ampliamente en el área de Machine Learning. Tradicionalmente, las técnicas para el aprendizaje de Redes Bayesianas han sido desarrolladas con miras a su uso en datos discretos, forzando así la discretización de series de tiempo continuas. Dado que la correcta discretización de las series de tiempo es un problema importante, se ha desarrollado un nuevo método con el objetivo de representar tanta información como sea posible, tomando en cuenta tanto la amplitud como las variaciones de la serie de tiempo. Este método de discretización requiere un parámetro (σ) que especifica la importancia dada a la pendiente en cada punto de la serie de tiempo. Distintos valores de σ producen diferentes secuencias discretas. Cuando se intenta descubrir relaciones entre series de tiempo, es deseable utilizar aquellos valores de σ que produzcan la mayor coincidencia entre ellas.

Se espera que pase algún tiempo entre la ocurrencia de un evento y su efecto correspondiente. Esto se refleja como retrasos en tiempo entre las series que representan tales eventos. Las series de tiempo discretizadas deben ser alineadas para descubrir el retraso correcto entre ellas.

Debido a que distintos valores de σ pueden producir distintos retrasos en tiempo, estos parámetros deben ser vistos como una unidad. Se aplicó un algoritmo de recocido simulado para encontrar la mejor configuración para estos valores. Este proceso facilita el descubrimiento de relaciones sutiles entre pares de series. La Red Bayesiana fue entonces extraída a partir de las secuencias discretas alineadas utilizando algoritmos de aprendizaje bien conocidos.

Se realizó una extensión al modelo de Redes Bayesianas para representar retrasos de tiempo entre pares de nodos. De este modo, cada arco que une dos nodos tiene un número entero asociado, obtenido de la alineación de las series de tiempo subyacentes, que representa la diferencia de tiempo entre la ocurrencia de un evento en el "nodo causa" y la ocurrencia del evento correspondiente en el "nodo consecuencia".

Por último, algunas series de tiempo pueden ser recuperadas a partir de la Red Bayesiana para fines de evaluación y predicción a corto plazo. Para esto se deben asignar valores conocidos a algunos otros nodos, con el objetivo de obtener la densidad de probabilidad en el nodo objetivo y muestrear sobre ella.

La herramienta ha sido implementada como un ambiente distribuido, permitiendo la colaboración entre usuarios que se encuentran en diferentes lugares. La distribución se llevó a cabo utilizando objetos CORBA.

PROLOGO

Objetivo

El objetivo de este trabajo es presentar una alternativa para el estudio de las relaciones existentes entre varias series de tiempo, mediante la extracción automática de modelos basados en Redes Bayesianas. Los modelos obtenidos cuantifican la información referente a retrasos de tiempo en relaciones de causalidad, es decir, muestran la cantidad de unidades de tiempo que deben transcurrir antes de que el cambio en una serie de tiempo afecte a otra que depende de ella. Dada esta información y las densidades de probabilidad invariablemente presentes en las Redes Bayesianas, un modelo permite reconstruir e incluso predecir una serie de tiempo a partir de otras.

Básicamente, el procedimiento utilizado consiste en discretizar y desplazar las series de tiempo a fin de que compartan el máximo de información, extraer una Red Bayesiana a partir de las secuencias discretas alineadas, evaluar la calidad del modelo recuperando alguna serie de tiempo a partir del mismo y, opcionalmente, utilizar el modelo para predicción a corto plazo.

De manera puntual, los objetivos de esta tesis son:

- Definir un método para la extracción automática de Redes Bayesianas a partir de series de tiempo.
- Agregar información temporal a las Redes Bayesianas.
- Permitir la recuperación de una serie de tiempo a partir de una Red Bayesiana.
- Permitir, cuando sea posible, la utilización de las Redes Bayesianas para predicción de series de tiempo a corto plazo.

Motivación

El modelado de relaciones entre variables por medio de grafos se ha desarrollado en dos sentidos. Por una parte, se encuentran las técnicas orientadas a la representación de relaciones entre variables temporales, cuyo resultado es un grafo con arcos dirigidos y/o no dirigidos. Aun cuando este tipo de modelo tiene la ventaja de que se puede extraer a partir de variables continuas, es muy limitado debido a que no contiene más información que la explícitamente mostrada por el grafo.

Por otra parte, existen los modelos de Redes Bayesianas que contienen toda la información necesaria para reconstruir el modelo probabilístico de las relaciones entre las variables, pero cuyo desarrollo se ha enfocado a las bases de datos con registros independientes y valores discretos. Debido a esto, este tipo de modelo no considera ninguna característica proveniente del tiempo, tal como desplazamientos entre series de tiempo.

La motivación principal de este trabajo es la manifiesta utilidad de los modelos de relaciones con grafos, aunada a la necesidad de modelos ricos en información para series de tiempo.

Aportaciones

La aportación más directa es la adaptación de las Redes Bayesianas para su explotación con variables temporales. Esto supone la adición de información, principalmente en los arcos del grafo, con el fin de representar relaciones temporales.

Además, gracias a la información temporal añadida a la Red Bayesiana, este trabajo plantea su utilización para predicción de series de tiempo. Aun cuando la propuesta de que es posible utilizar las Redes Bayesianas para predicción a corto plazo pudiera resultar una perogrullada para el conocedor de estos modelos, su utilización para predicción de series de tiempo requiere consideraciones adicionales al simple hecho de muestrear u obtener la media de una densidad de probabilidad. Asimismo, al combinarse con el método de discretización utilizado, este tipo de predicción permite obtener datos adicionales, tales como una estimación del error para cada punto.

Una aportación colateral importante ha sido el desarrollo de un nuevo método de discretización capaz de tomar en cuenta, al mismo tiempo, la magnitud y las variaciones de la serie de tiempo. Además, este método ha probado introducir una menor cantidad de ruido que los métodos de discretización más comunes provenientes de comunicaciones digitales. Por último, al realizar una búsqueda sobre el único parámetro de la discretizacón y sobre los retrasos de tiempo entre las series, es posible obtener relaciones difíciles de detectar entre dos o más series de tiempo.

En síntesis, las aportaciones de este trabajo son:

- Un nuevo método para la discretización de series de tiempo.
- Un método para relacionar dos o más series de tiempo.
- Extensión de las Redes Bayesianas para incluir información temporal.
- Un método para la recuperación de una serie de tiempo a partir de una Red Bayesiana.
- Utilización de las Redes Bayesianas para predicción de series de tiempo a corto plazo.
- El procedimiento completo para la extracción automática de Redes Bayesianas a partir de series de tiempo.

Es importante resaltar que en este trabajo no se hace ni se pretende hacer aportación alguna referente a algoritmos para el aprendizaje de Redes Bayesianas tradicionales. Acerca de este tema existe abundante investigación alrededor del mundo, de modo que aquí solo se hace uso de algunos algoritmos ya publicados.

Estructura de la tesis

En el capítulo 1 se plantea el contexto en el que se estudian las Redes Bayesianas, y se da un panorama general sobre el proceso de extracción de Redes Bayesianas a partir de Bases de Datos Temporales. En el capítulo 2 se explica el concepto de serie de tiempo y se

presentan algunas de las técnicas más conocidas para su modelado y predicción. En el capítulo 3 se extractan los conceptos más importantes referentes al modelado de relaciones mediante la utilización de grafos y se da una breve explicación acerca de Redes Bayesianas. El capítulo 4 presenta el estado del arte referente a discretización de variables continuas, algoritmos de agrupamiento (clustering), extracción de modelos de grafos a partir de series de tiempo y extracción de Redes Bayesianas a partir de Bases de Datos. En el capítulo 5 se expone, de manera tanto intuitiva como semiformal, el método de discretización creado para mantener información acerca de la amplitud y variaciones de la serie de tiempo. En el capítulo 6 se describen los algoritmos utilizados para la obtención de los parámetros de discretización y alineación de las series de tiempo, presentándose algunas pruebas y resultados. En el capítulo 7 se explica la extracción de la Red Bayesiana a partir de las secuencias discretas alineadas, así como la recuperación de las series de tiempo a partir del modelo. Posteriormente se presentan algunas pruebas y resultados utilizando las secuencias discretas obtenidas en el capítulo 6. En el capítulo 8 se presenta una discusión acerca del valor correcto de algunos parámetros de acuerdo al tipo de información que se desee obtener, así como de la confiabilidad de los resultados. También se presentan opciones para la función utilizada durante la discretización. En el capítulo 9 se presenta de manera sucinta la arquitectura de la herramienta, misma que permite el trabajo de manera distribuida, así como algunas instrucciones y ejemplos para su uso. Finalmente, en el capítulo 10 se presentan las conclusiones obtenidas a lo largo de este trabajo. El apéndice A muestra los resultados obtenidos al aplicar otros métodos de modelado a las series de tiempo utilizadas.

El lector que cuente con conocimiento suficiente acerca de los temas básicos para esta tesis puede referirse únicamente a los capítulos 1, 5, 6, 7 y 8. Si solo se desea conocer el método de discretización basta con leer el capítulo 5. Los capítulos 1 y 5 son indispensables para comprender la extracción de Redes Bayesianas a partir de Bases de Datos temporales. La figura P.1 muestra la dependencia entre los capítulos de esta tesis. Las líneas punteadas indican la dependencia entre capítulos para cualquier persona que no conozca a fondo los antecedentes de los temas estudiados.

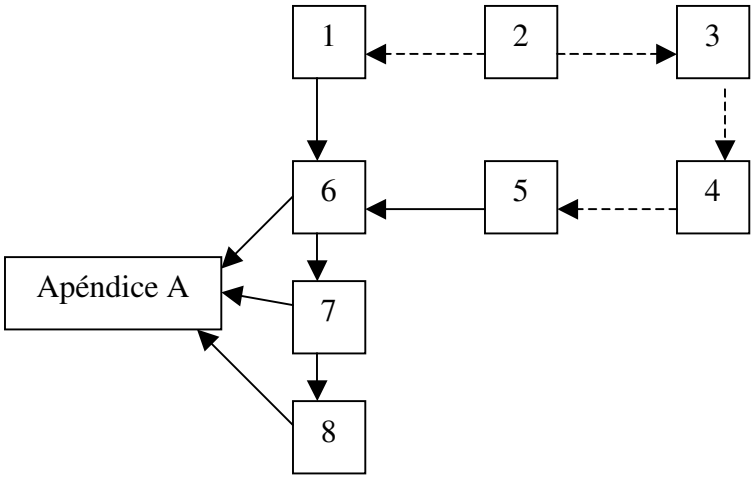


Figura P.1. Dependencia entre los capítulos

1. INTRODUCCIÓN

La obtención de información ha sido, durante largo tiempo, una de las tareas más importantes para el ser humano. La adquisición de datos a partir de observaciones y la inferencia de información a partir de los mismos es uno de los métodos por medio de los cuales se ha logrado un mayor avance en diversas áreas de conocimiento.

En los últimos años, el uso de la computadora ha permitido la manipulación automática de datos, facilitando el análisis exhaustivo de los mismos aún cuando se presentan en grandes cantidades. Tareas tan humanas como el conocimiento y trato individualizado del cliente se confían de manera cada vez más frecuente a una computadora.

Debido a la manera en que las bases de datos están organizadas, gran parte del esfuerzo para extracción automática de información ha dejado de lado su búsqueda en bases de datos temporales, es decir, bases de datos en las que la sucesión de registros obedezcan algún orden en su aparición. Aunque se han logrado avances importantes prescindiendo de esta característica, en la actualidad es posible retomar su estudio para obtener conocimiento que ha quedado oculto para la mayoría de las técnicas de aprendizaje automatizado.

1.1 "Machine Learning" y Redes Bayesianas

Dentro del área de la computación se ha desarrollado un campo conocido como *Machine Learning*, dedicado a la extracción automática de información a partir de datos. Esta información puede ser utilizada para mejorar el desempeño de un programa, apoyar la toma de decisiones, realizar diagnósticos, etc. Entre los principales temas estudiados por Machine Learning se encuentran [Mitchell, 1997]:

- ***Arboles de Decisión.*** Es un método para aproximar funciones con valores discretos. Clasifican instancias ordenándolas de forma descendente en el árbol, desde la raíz hasta algún nodo hoja. Cada nodo en el árbol especifica algún tipo de atributo de la instancia, y cada rama que desciende de ese nodo corresponde a uno de los posibles valores para ese atributo.
- ***Redes Neuronales Artificiales.*** Es un método para aproximar funciones en dominios reales, discretos o vectoriales.
- ***Evaluación de Hipótesis.*** Enfocada principalmente a los problemas de estimar la exactitud de una hipótesis y elegir la mejor entre un conjunto.
- ***Aprendizaje Bayesiano.*** Provee un método probabilístico para inferencia. Se basa en la suposición de que las variables de interés están gobernadas por distribuciones de probabilidad, y de que es posible tomar decisiones óptimas en base al razonamiento sobre tales probabilidades unidas a los datos observados.
- ***Métodos basados en instancias.*** A diferencia de los métodos de aprendizaje que construyen una descripción general de la función objetivo a partir de instancias, estos métodos simplemente almacenan las instancias de entrenamiento. La generalización se lleva a cabo cuando se presenta una nueva instancia. Entre los métodos más importantes dentro de ésta clase se encuentra el de *Vecinos Cercanos*.

- ***Algoritmos genéticos.*** Proveen un método de aprendizaje motivado por una analogía con la evolución biológica. Estos algoritmos generan nuevas hipótesis mutando y recombinando partes de hipótesis conocidas.

Dentro de los métodos de aprendizaje bayesiano se encuentran las *Redes Bayesianas*, también conocidas como *Redes de Creencia*. Estas redes describen de manera compacta la distribución de probabilidad que gobierna a un conjunto de variables, especificando suposiciones de independencia condicional entre las mismas, así como distribuciones de probabilidad condicional que expresan de manera cuantitativa las relaciones entre ellas.

1.2 Extracción de Redes Bayesianas predictivas

Si se tienen secuencias de mediciones de fenómenos expresadas como series de tiempo (ver capítulo 2), es posible expresar cada una de éstas en forma de un nodo perteneciente a una Red Bayesiana, como se muestra en la figura 1.1. Las relaciones de dependencia entre las series de tiempo están representadas por los arcos de la red.

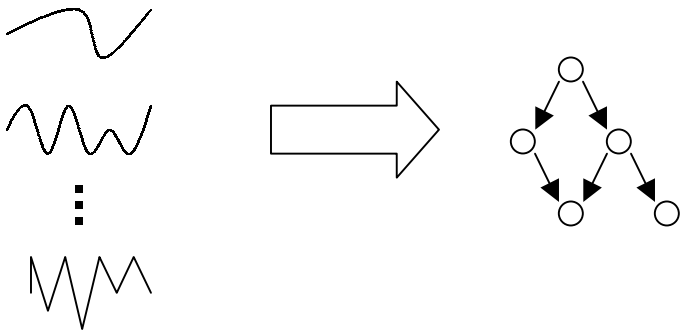


Figura 1.1 Extracción de una Red Bayesiana a partir de un conjunto de series de tiempo

Al contar con un modelo de este tipo es posible conocer, de manera cuantitativa, el efecto que sobre una variable tiene la modificación del entorno, es decir, la modificación de otras variables relacionadas. Esta es una herramienta útil para la toma de decisiones, así como para la predicción a corto plazo.

1.2.1 Extracción clásica de Redes Bayesianas

En la actualidad existe un gran desarrollo en la extracción de Redes Bayesianas a partir de Bases de Datos. Los métodos actuales consideran a la Base de Datos como un conjunto de *casos*, en donde cada caso es una instancia de un conjunto de variables. Ver figura 1.2.

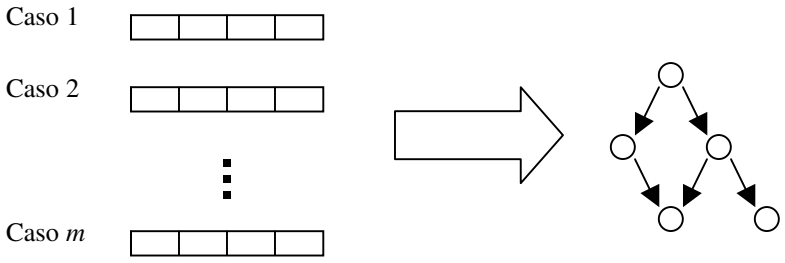


Figura 1.2 Extracción de Redes Bayesianas a partir de Bases de Datos

Generalmente se considera que las variables en los casos presentan valores discretos. Esto permite expresar las distribuciones de probabilidad de manera mucho más sencilla, al tiempo que facilita la implantación de algoritmos y medidas tales como aquellas provenientes de la Teoría de la Información. Así, si se desea extraer la Red Bayesiana a partir de un conjunto de series de tiempo, es necesario utilizar algún método de discretización como etapa de preprocesamiento. En esta tesis se utilizó un método especial que toma en cuenta el valor de cada punto de la serie de tiempo y su variación respecto al punto anterior.

1.2.2 Alineación de series de tiempo

Cuando se desea aplicar los algoritmos existentes para la obtención de Redes Bayesianas utilizando series de tiempo en lugar de los casos tradicionales, se hace necesario considerar diferencias que aparecen debido a la intervención del tiempo. Una de las consideraciones más importantes es que, si existen dos variables X y Y, y se observa que el valor de X tiene influencia sobre el valor de Y, es probable que transcurra un lapso de tiempo τ antes de que un cambio en la variable X se vea reflejado por un cambio en Y, como se muestra en la figura 1.3.

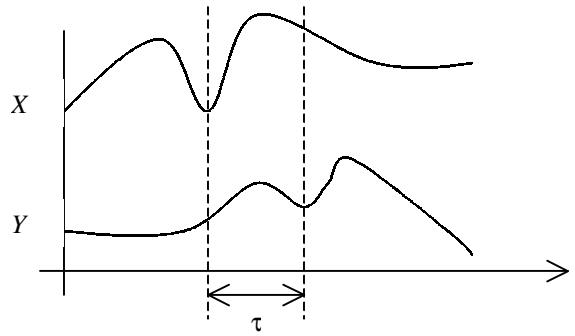


Figura 1.3 Lapso de tiempo entre causa y efecto

Al intentar extraer relaciones entre las variables, si se considera el valor de las series en un instante de tiempo t, la relación obtenida no será significativa debido a que el verdadero

valor de *Y* causado por *X* se encuentra en $t+\tau$. Esto se puede solucionar desplazando la serie de tiempo de la variable *X* un número τ de unidades de tiempo hacia adelante. Nótese que al hacer esto, los valores de las series de tiempo aumentarán su dependencia. En adelante, llamaremos a este proceso *alineación* de las series de tiempo.

1.2.3 Proceso completo

Una vez que las series de tiempo han sido discretizadas y alineadas, es posible utilizar algún método conocido para la extracción de Redes Bayesianas a partir de Bases de Datos, tomando como casos aquellos instantes de tiempo en los que se cuente con valores para todas las variables. El proceso completo se muestra en la figura 1.4.

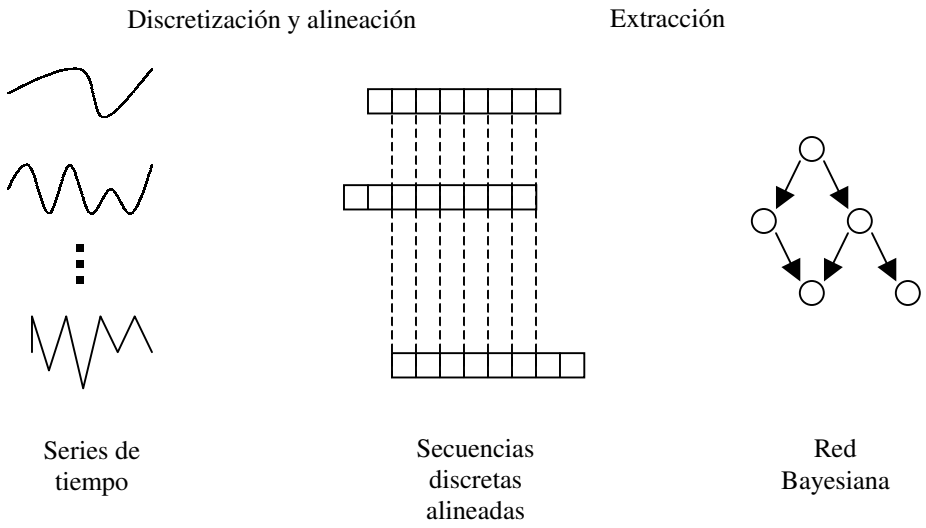


Figura 1.4 Proceso completo para la extracción de Redes Bayesianas a partir de Series de Tiempo

1.3 Recuperación de Series de Tiempo a partir de la Red Bayesiana

Una vez construída la Red Bayesiana, es posible determinar la distribución de probabilidad para una variable determinada utilizando reglas tales como la Regla de la Cadena y la Regla de Bayes [Russell & Norving, 1995]. Esta distribución puede ser calculada tomando en cuenta el conocimiento acerca del valor de otra variable, como se muestra en la figura 1.5.

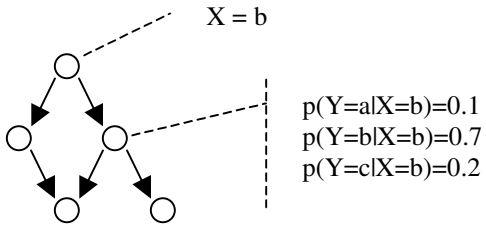


Figura 1.5 La distribución de probabilidad de *Y* es calculada dado que *X=b*

Si a cada variable de un conjunto no vacío $P = \{X_1, X_2, ...,X_n\}$ se le asignan los valores de su respectiva serie de tiempo, al calcular la distribución de probabilidad para algún otro nodo $Y \notin P$ conectado con algún nodo $X_i \in P$, ésta debe tender a producir los valores de la serie de tiempo correspondiente a Y. Así, si la estructura y parámetros obtenidos para la Red Bayesiana son correctos, se pueden recuperar valores similares a los de la serie original muestreando sobre la distribución resultante.

Al muestrear sobre la distribución de probabilidad para una variable se obtiene un valor discreto. Si al convertir este valor discreto a un valor continuo se toma en cuenta la pendiente que tenía la serie de tiempo original, el valor continuo recuperado dependerá del valor continuo anterior. Por ejemplo, suponga que al muestrear sobre la distribución de la variable Y se obtiene el valor discreto b_t. Suponga ahora que el valor b_t representa un aumento de cierto número de unidades en la serie de tiempo, entonces el valor calculado para la serie de tiempo correspondiente a Y dependerá del valor b_{t-1} de la misma. De este modo, si se calcula una secuencia de valores, lo que se obtendrá será una de varias trayectorias que podría tomar la serie de tiempo recuperada.

Dado que las distribuciones de probabilidad están expresadas en base a símbolos, y dado que estos símbolos no necesariamente representan valores de amplitud o de pendiente que sigan una distribución normal, no se debe esperar que las trayectorias obtenidas tengan una distribución de este tipo. Sin embargo, es posible seleccionar la trayectoria más adecuada calculando varias trayectorias posibles y obteniendo el punto óptimo para cada instante de tiempo, como se muestra en la figura 1.6.

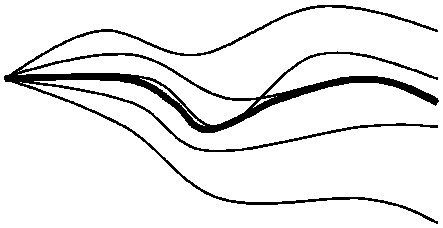


Figura 1.6 Obtención de la trayectoria más adecuada a partir de un conjunto de trayectorias

Es posible calcular una estimación del error que tendrá la reconstrucción de la serie midiendo la divergencia de las trayectorias calculadas. Así, la estimación del error se puede obtener en base a las diferencias entre cada una de las trayectorias y la trayectoria elegida.

2. SERIES DE TIEMPO

Existen fenómenos naturales y sociales cuyas variables pueden ser estudiadas registrando su comportamiento en forma de series de tiempo. Una serie de tiempo es una colección de valores:

$$\{x_t : t=1...n\}$$

En donde t indica el tiempo en el cual el dato x_t es observado [Chan, 2002]. Estos datos son comúnmente mediciones de una variable, natural o artificial, que puede servir para comprender o predecir el comportamiento del fenómeno al que pertenece. Generalmente un fenómeno se ve afectado por m variables v_1, v_2, ..., v_m. Si $m>1$ se dice que el fenómeno es multivariado. Dado que la serie de tiempo es generada por el fenómeno en cuestión, ésta depende implícitamente de dichas variables.

Es común caracterizar una serie de tiempo como un *proceso estocástico* definido a intervalos discretos. Básicamente, un proceso estocástico es una función aleatoria del tiempo, definida sobre algún intervalo de observación, con la característica de que, antes de hacer un experimento, no es posible conocer su valor.

Sea $\{X_j(t)\}$ un proceso estocástico definido en tiempos discretos t_1, t_2, ..., t_n. Para caracterizar a este proceso es necesario conocer la densidad de probabilidad conjunta de todas las variables aleatorias X. La densidad de probabilidad conjunta expresa la probabilidad de ocurrencia de cada posible configuración de las variables, es decir, las probabilidades de que cada variable tome un valor determinado.

Sea θ la densidad de probabilidad de un proceso estocástico, y sea $D=\{x_1, x_2, ..., x_n\}$ un conjunto de datos generados por dicho proceso. El *likelihood* de la distribución de probabilidad θ dado un conjunto de datos D es igual a la probabilidad de los datos dada la distribución, y se denota:

$$L(\theta : D) = p(D|\theta)$$

Una *estadística suficiente* es una función que sintetiza, a partir de los datos, la información relevante para el cálculo de probabilidad. Así, el likelihood se puede calcular a partir de una estadística suficiente. Por ejemplo, si se tiene una distribución binomial en donde n_e representa el número de éxitos y n_f el número de fracasos, estas dos cantidades forman una estadística suficiente. De éste modo, el likelihood se puede calcular como:

$$L(\theta : D) = \theta^{n_e}\left(1-\theta^{n_f}\right)$$

Se dice que un proceso estocástico es *estacionario en el sentido estricto* si su densidad conjunta es invariante bajo cambios de posición en tiempo respecto al origen, es decir, si la

densidad de probabilidad solo depende de las diferencias t_i-t_j entre instantes de tiempo t_1, t_2, ..., t_n, pero no directamente de estos valores [Hyvärinen et. al., 2001].

2.1 Modelado y predicción

Dada una serie de tiempo, es posible construir un modelo $Y(v_1, v_2, ..., v_m)$ que describa su comportamiento dados los parámetros v_1, v_2, ..., v_m. Es decir, dados n valores de la serie de tiempo, se construye el modelo

$$Y\left(v_1(t), v_2(t), \text{K}\ , v_m(t)\right) = \hat{x}(t) \qquad \text{para } t=1...n \qquad \textbf{(2.1)}$$

en donde $v_1(t)$, $v_2(t)$, ..., $v_m(t)$ son los valores de los parámetros en el tiempo t, y $\hat{x}(t)$ es la estimación de x para el instante t. En varios modelos, Y es una función del tiempo, por lo que el modelo queda expresado como

$$Y(t) = \hat{x}(t)$$

Se define el error de modelado como

$$e(t) = x(t) - \hat{x}(t)$$

Dado el modelo Y, en ocasiones se desea predecir un valor $x(n+h)$, en donde h se conoce como el *horizonte de predicción*. Así, se define la función de predicción $Y(n,h)$ como

$$Y(n,h) = E(x(n+h)\,|\,x(n), \text{K}\,) = \hat{x}(n+h)$$

Y el error de predicción se define como

$$e_n(h) = x(n+h) - \hat{x}(n+h)$$

Existe un gran desarrollo en el modelado y predicción de series de tiempo mediante distintas técnicas, entre las que podemos destacar "Autoregresión" [Chan, 2002], Redes Neuronales [Sánchez & Figueroa, 2001], Predicción Bayesiana [Pole et. al, 1994], etc. Existen trabajos en los que se ha estudiado el comportamiento de varias técnicas de modelado y predicción para distintos tipos de series de tiempo [Bautista & Figueroa, 2002].

2.1.1 Modelado de fenómenos

Dado que la mayoría de los fenómenos son multivariados y presentan un comportamiento no lineal, los modelos que se construyen deberían, en principio, conservar éstas características. Sin embargo, es común que al resolver problemas se asuman simplificaciones del comportamiento real, lo que lleva en ocasiones a la obtención de modelos lineales o que dependen de un menor número de variables.

Clásicamente, un fenómeno se puede modelar utilizando ecuaciones diferenciales [Morrison, 1991], que se convierten en ecuaciones diferenciales parciales cuando el fenómeno es multivariado, y en no lineales cuando el fenómeno es no lineal. Además, es usual agregar un componente que modele el comportamiento estocástico del fenómeno. Así, los modelos más generales se construyen utilizando ecuaciones diferenciales parciales, no lineales y estocásticas de la forma:

$$g\left[f(v_1, v_2, \text{K } v_m, t), \frac{\partial f(v_1, v_2, \text{K } v_m, t)}{\partial t}, \text{K }, \frac{\partial^p f(v_1, v_2, \text{K } v_m, t)}{\partial t^p}, \gamma(t), t \right] = 0 \quad \textbf{(2.2)}$$

En donde $\gamma(t)$ es una función estocástica. Resolviendo la ecuación (2.2) para la variable v_i se obtiene un modelo de la forma (2.1) para la serie de tiempo producida por esta variable.

A pesar de ser muy útiles y representativos, estos modelos tienen la desventaja de ser muy difíciles de construir para la mayoría de los fenómenos reales, y aún en aquellos casos en que se pueden construir, su solución es frecuentemente demasiado complicada.

2.1.2 Modelos clásicos para Series de Tiempo

De forma general, existen tres modelos clásicos para series de tiempo: promedio variable (moving average, MA), autoregresión (AR) y autoregresión con promedio variable (autoregresive moving average, ARMA).

Sea $\{Z(t)\}$ una secuencia de variables no correlacionadas con distribución idéntica, con media cero y varianza σ^2. A esta secuencia se le conoce como *secuencia de ruido blanco*.

Un modelo de promedio variable es básicamente un promedio pesado de las variables $\{Z(t)\}$. El modelo de promedio variable de orden q, MA(q), se expresa como:

$$\hat{x}(t) = Z(t) + \theta_1 Z(t-1) + \Lambda + \theta_q Z(t-q)$$

Un modelo de regresión múltiple se basa en la suposición de que los valores de una variable dependiente $\hat{y}$ están determinados de manera lineal por diversas variables $\hat{x}_1, \hat{x}_2, \text{K }, \hat{x}_m$ y un componente de ruido u [Mirer, 1983]:

$$\hat{y}(t) = \phi_o + \phi_1 x_1(t) + \phi_2 x_2(t) + \Lambda + \phi_n x_n(t) + u(t)$$

Un modelo de autoregresión es un modelo de regresión en el que la variable dependiente $\hat{y}$ es igual al valor actual de la serie de tiempo $\hat{x}(t)$, y las variables independientes $\hat{x}_1, \hat{x}_2, \text{K }, \hat{x}_m$ son valores anteriores de la serie $x(t\text{-}1)$, $x(t\text{-}2)$, ..., $x(t\text{-}p)$. El modelo de autoregresión AR(p) se puede escribir como $\phi(B)\hat{x}(t) = Z(t)$, en donde $\phi(B) = (1 - \phi_1 B - \Lambda - \phi_p B^p)$, $B\hat{x}(t) = \hat{x}(t-1)$, de modo que

$$\hat{x}(t) = \phi_1 \hat{x}(t-1) + \Lambda + \phi_p \hat{x}(t-p) + Z(t)$$

En ocasiones es necesario utilizar modelos MA o AR relativamente grandes para capturar la estructura de una serie de tiempo. Para obtener modelos más simples, es posible combinar AR y MA, obteniendo el modelo de autoregresión con promedio variable ARMA. Este modelo se expresa como:

$$\hat{x}(t) + \phi_1 \hat{x}(t-1) + \Lambda + \phi_p \hat{x}(t-p) = Z(t) + \theta_1 Z(t-1) + \Lambda + \theta_q Z(t-q)$$

2.1.3 Modelos con Redes Neuronales

Las Redes Neuronales Artificiales son modelos computacionales formados por unidades de procesamiento densamente conectadas. Dichas redes son implantaciones paralelas de grano fino de sistemas dinámicos o estáticos [Hassoun, 1995].

Una de las características más importantes de este tipo de modelo es su capacidad de generalización, es decir, que a partir de un conjunto de entradas de entrenamiento son capaces de procesar entradas desconocidas. Además, debido a que el aprendizaje se lleva a cabo a través de ejemplos y no de manera algorítmica, las Redes Neuronales son capaces de producir resultados aceptables aun para problemas cuyo dominio no es completamente conocido.

Existe un tipo especial de Red Neuronal conocido como *Red Neuronal con Retraso de Tiempo*. La arquitectura característica de este tipo de red se muestra en la figura 2.1.

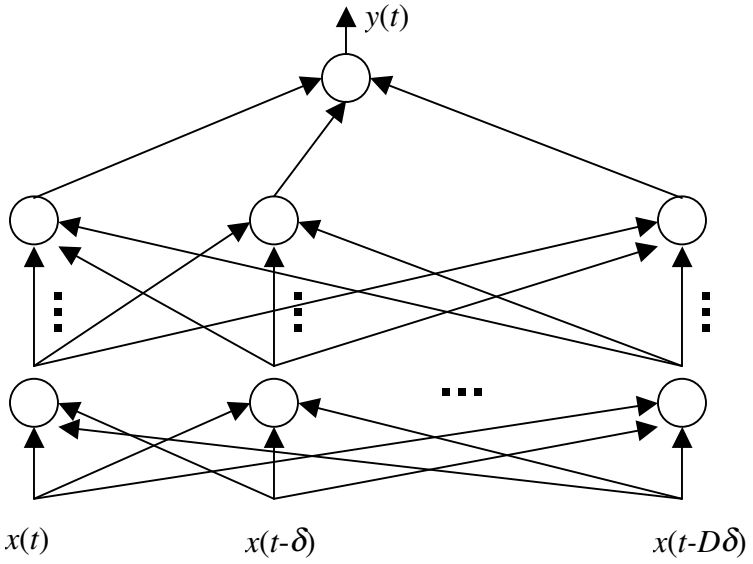


Figura 2.1. Arquitectura de una Red Neuronal con Retraso de Tiempo

Una Red Neuronal con Retraso en Tiempo presenta un comportamiento equivalente al de un filtro con Respuesta Impulso Finita (Finite Impulse Response), por lo que también es conocida como red *FIR* o *FIRnet*.

Una red FIR representa un modelo de la forma

$$y(t) = f\left(x(t), x(t-\delta), \mathrm{K}\ , x(t-D\delta)\right)$$

De acuerdo al teorema de Takens [Takens, 1981], bajo ciertas condiciones existe una relación funcional de la forma

$$x(t+h) = g\left[x(t), x(t-\delta), \mathrm{K}\ , x(t-D\delta)\right]$$

Si se establece $y(t)=x(t+h)$, entonces una Red Neuronal FIR puede utilizarse para modelar o predecir una serie de tiempo.

2.2 Extracción de reglas

Varias de las técnicas anteriormente mencionadas basan su funcionamiento en el estudio de los datos de una sola serie de tiempo para su modelado y predicción. Ya sea que estén basadas en un análisis estadístico, en la reconstrucción en el espacio de fase, o en cualquier otro método, algunas técnicas no toman en cuenta la posible relación de la serie de tiempo bajo estudio con otras series de tiempo disponibles.

En contraste, existen otras técnicas [Avila & Figueroa, 2002] cuyo funcionamiento se basa en el descubrimiento de reglas a partir de un conjunto de series de tiempo. Este tipo de técnica permite descubrir las relaciones existentes entre valores o variaciones de valores en las series de tiempo. El estudio realizado por expertos en áreas específicas sigue comúnmente este esquema, es decir, el experto observa el comportamiento de otras variables a las que se tiene acceso, y en base a esta información intenta analizar o predecir el comportamiento de la variable de interés. Por ejemplo, en economía es común intentar predecir el comportamiento del valor de las acciones de cierta compañía mediante la observación de variables macroeconómicas y las condiciones políticas y sociales del país en el que se encuentra. En genética el caso es aún más claro, ya que uno de los principales problemas a los que se enfrenta esta área es el descubrimiento de patrones de activación e inhibición entre distintos genes y/o promotores.

Más allá del modelado o predicción de una serie de tiempo, la obtención de reglas permite comprender mejor el fenómeno bajo estudio, aportando información que pudiera estar oculta a simple vista. Además, la información obtenida por herramientas de este tipo permite cuantificar las relaciones encontradas, eliminando problemas de subjetividad introducidos por factores humanos.

En el ambiente informático, estas ventajas han llevado a un uso creciente de distintas herramientas de minería de datos [Komorowski & Zytkow, 1997], que intentan encontrar de forma automática relaciones entre variables a partir de la información contenida en una Base de Datos. Las reglas obtenidas sirven como base sólida para la definición de estrategias y toma de decisiones. Sin embargo, muchas de las técnicas utilizadas para minería de datos son relativamente simples, debido en parte a que técnicas más sofisticadas

requieren gran cantidad de procesamiento que, al aplicarse a grandes bases de datos, requerirían tiempos de cómputo demasiado extensos.

2.2.1 Técnicas para la extracción de reglas

Existen diversas técnicas para la extracción de reglas. Entiéndase por regla la abstracción de una estructura, generalmente oculta, dentro de una serie de tiempo o un conjunto de series de tiempo.

Dentro de los algoritmos más modernos y eficientes se encuentra MSDD (Multi-Stream Dependency Detection) [Oates et. al., 1996], que efectúa una búsqueda sistemática de tipo general a específico, con una heurística de primero el mejor, sobre datos categóricos. El tipo de reglas generadas por este algoritmo se conoce como *dependencias*, las cuales son de tipo predecesor-sucesor y se espera que sean altamente predictivas. MSDD arroja las k dependencias más fuertes, en donde la fortaleza de una regla está dada por una estadística conocida como *estadística G*.

También se han desarrollado algoritmos que permiten la extracción de reglas comprensibles a humanos a partir de Redes Neuronales entrenadas sobre una serie de tiempo. Este tipo de algoritmo produce distintos tipos de reglas, tales como reglas IF THEN [Gaweda et. al., 2000] o árboles de decisión [Craven & Shavlik, 1997], que representan la información contenida en las Redes Neuronales.

2.2.2 Distintos tipos de reglas

Existen diversos tipos de reglas que expresan información distinta. Por ejemplo, las dependencias arrojadas por MSDD son ocurrencias de eventos en el tiempo que se presentan de manera especial, es decir, demasiado frecuente o infrecuentemente.

Otro tipo de regla se expresa mediante árboles de decisión. Un árbol de decisión es una estructura que toma como entrada un objeto o situación descrita por un conjunto de propiedades y produce como salida una decisión de tipo verdadero/falso. Así, los árboles de decisión representan funciones Booleanas. También es posible construir árboles de decisión con un intervalo más amplio de valores [Russell and Norvig, 1995].

La mayor parte de las técnicas de extracción de reglas se enfrentan a un problema de representación debido a que los tipos de datos analizados pueden no ser compatibles. Por ejemplo, la representación de datos ordinales y no ordinales mediante los mismos símbolos puede llevar a confusiones que resultan en un mal procesamiento o en una interpretación errónea de los resultados. Una de las formas más usuales para expresar reglas de relación entre variables heterogéneas es llevar dichas relaciones al espacio de probabilidad. En este sentido, el conjunto de variables X_1, X_2, ..., X_N, se convierte en un conjunto de variables aleatorias, de las cuales es posible extraer la función de densidad de probabilidad conjunta:

$$f(X_1, X_2, \mathrm{K}\ , X_N) : X_1 \times X_2 \times \mathrm{K} \times X_N \rightarrow [0..1]$$

De esta forma, no importa el tipo de las variables con las que se trabaje. Cada variable toma valores en su propio espacio, independiente del espacio de otras variables. Sin embargo, esta representación involucra una restricción de tipo computacional debido a que es difícil representar funciones continuas cuyo comportamiento no es previamente conocido. Debido a esto, es común discretizar las variables continuas a fin de que la función de densidad de probabilidad conjunta se convierta en una función discreta, en donde cada una de las variables toma r_i valores x_{i1}, x_{i2}, ..., $x_{i(ri)}$, i=1...N:

$$\begin{aligned}
&\Pr(X_1 = x_{11}, X_2 = x_{21}, \mathrm{K}\,, X_N = x_{N1}) = p_1 \\
&\Pr(X_1 = x_{12}, X_2 = x_{22}, \mathrm{K}\,, X_N = x_{N2}) = p_2 \\
&\Pr(X_1 = x_{13}, X_2 = x_{23}, \mathrm{K}\,, X_N = x_{N3}) = p_3 \\
&\quad\vdots \\
&\Pr(X_1 = x_{1r_1}, X_2 = x_{2r_2}, \mathrm{K}\,, X_N = x_{Nr_N}) = p_M
\end{aligned}$$

El tamaño del dominio de cada variable es independiente del resto de las variables. El número de valores de probabilidad necesarios para especificar la densidad conjunta de probabilidad es:

$$M = \prod_N^1 r_i$$

Como se puede observar, si la cardinalidad del espacio de probabilidad (r_i) es la misma para todas las variables, el número de valores de probabilidad crece exponencialmente con el número de variables.

La información contenida en la densidad conjunta de probabilidad puede representarse también en términos de probabilidad condicional. Sea el conjunto $I = \{X_{I_1}, X_{I_2}, \mathrm{K}\,, X_{I_K}\}$, cada entrada de la densidad conjunta de probabilidad se puede rescribir como:

$$\begin{aligned}
&\Pr(X_1, X_2, \mathrm{K}\,, X_{I_1}, X_{I_2}, \mathrm{K}\,, X_{I_k}, \mathrm{K}\,, X_N) = \\
&\qquad \Pr(X_1, X_2, \mathrm{K}\,, X_N \,|\, X_{I_1}, X_{I_2}, \mathrm{K}\,, X_{I_K}) \Pr(X_{I_1}, X_{I_2}, \mathrm{K}\,, X_{I_K})
\end{aligned}$$

Si las variables contenidas en el conjunto I son independientes del resto de las variables, la probabilidad conjunta se puede separar en dos factores:

$$\Pr(X_1, X_2, \mathrm{K}\,, X_{I_1}, X_{I_2}, \mathrm{K}\,, X_{I_K}, \mathrm{K}\,, X_N) = \Pr(X_1, X_2, \mathrm{K}\,, X_N) \Pr(X_{I_1}, X_{I_2}, \mathrm{K}\,, X_{I_K})$$

De este modo, el número de valores de probabilidad necesarios para especificar la distribución de probabilidad conjunta se reduce a:

$$M = \prod_{X_i \in I} r_i + \prod_{X_i \notin I} r_i$$

Aplicando este razonamiento sobre las variables contenidas en I y en su complemento I^c, se pueden formar conjuntos cada vez más pequeños hasta que todas las variables contenidas en los conjuntos sean dependientes entre sí.

3. MODELOS DE DEPENDENCIA CON GRAFOS

Una manera de modelar un conjunto de series de tiempo (ver capítulo 2) es expresar cada una como una variable y obtener su distribución conjunta de probabilidad (ver sección 2.2.2). Sin embargo, en la distribución conjunta de probabilidad existe gran redundancia debido a que no se toma en cuenta la independencia entre variables. Esto ha motivado el desarrollo de modelos gráficos que expresan relaciones de dependencia e independencia entre las variables involucradas. En particular, en algunos trabajos se ha estudiado el descubrimiento de relaciones de dependencia entre series de tiempo y su expresión en modelos basados en grafos. Este tipo de modelo captura relaciones de causalidad y dependencia entre las series de tiempo.

3.1 Teoría de la información

En la actualidad no es posible hablar de una Teoría de la Información única. Por tratarse de un concepto tan amplio, la información ha sido estudiada desde varias perspectivas, construyéndose así diversas teorías cuyas aportaciones resultan útiles para campos de estudio específicos.

3.1.1 Diversas teorías de la información

Definitivamente una de las teorías de la información más importantes es la de Shannon [Shannon, 1948], cuyas definiciones y resultados han servido como base para un sinnúmero de adelantos en diversas áreas, especialmente en comunicaciones y computación. A pesar de ser pionera en su campo, la Teoría de la Información de Shannon no ha sido desplazada por ninguna otra; por el contrario, se ha enriquecido hasta el punto de servir como fundamento a una gran cantidad de investigaciones en diversas disciplinas.

Otra teoría importante es la Teoría Algorítmica de la Información [Chaitin, 1977], cuyo fundamento se encuentra principalmente en los trabajos de Kolmogorov, Solomonoff y Martin-Löf. Dada una secuencia s de valores 0, 1 o comas, esta teoría basa su estudio en tres parámetros: la probabilidad $P(s)$ de que una máquina de Turing, cuya entrada es una secuencia aleatoria, produzca s como salida, la entropía $H(s)$ definida como el negativo del logaritmo base 2 de $P(s)$, y la complejidad $I(s)$, definida como la longitud del programa más pequeño (mínimo) cuya salida es s.

Existe un trabajo en el cual se realizan aportaciones importantes para establecer una teoría general de la información [Flückiger, 1995], proponiendo a las “construcciones del cerebro” como su unidad básica. En dicho trabajo se explica que el concepto de información se ha estudiado principalmente desde dos perspectivas: la estructural-atributiva y la funcional-cibernética. El primer enfoque concibe a la información como estructura, orden, variedad, etc., mientras que el segundo la entiende como funcionalidad, significado funcional o como atributos de un sistema organizado. Uno de los trabajos más representativos de la perspectiva funcional-cibernética sería la Teoría de la Información de

Shannon. No se menciona la clasificación en la que se encontraría la Teoría Algorítmica de la Información.

3.1.2 Teoría de la Información de Shannon

En esta tesis se utiliza la Teoría de la Información de Shannon, principalmente debido a su uso en la mayor parte de las investigaciones relacionadas con el modelado basado en grafos. Así, en gran parte de la literatura para el estudio de modelos de grafos se encuentra el concepto de independencia entre variables. La independencia es un concepto estricto que va más allá de la ortogonalidad (no correlación) entre los datos [Hyvärinen et. al., 2001], y que requiere el uso de técnicas poderosas para ser identificada. Específicamente, la información mutua e información mutua condicional son dos medidas cuyo manejo es indispensable a lo largo de este trabajo. Para comprender las medidas de información mutua, es necesario recordar el concepto de entropía.

La entropía H es una medida de la cantidad de incertidumbre acerca del próximo valor de un evento, dado que se conoce el valor de los eventos anteriores. Si se tiene un evento que puede tomar r distintos valores $x_1, x_2, ...,x_r$, cada uno con probabilidad $\Pr(x_i)=p_i$, la entropía debe cumplir las siguientes propiedades:

1. H debe ser continua en los p_i.
2. Si todos los p_i son iguales, $p_i = \frac{1}{r}$, H debe ser una función monótona creciente en r.
3. Si una posibilidad se divide en dos valores, el valor original de H debe ser una suma ponderada de los valores de las nuevas H.

Se demuestra que la única función que cumple las 3 propiedades anteriores es:

$$H = -K \sum_{i=1}^{r} p_i \log p_i$$

en donde K es una constante positiva, generalmente con valor 1.

Supongamos que existen dos eventos X y Y, con r_X posibles valores para el primero y r_Y para el segundo. Sea $\Pr(x, y)$ la probabilidad de ocurrencia conjunta del valor x para X y de y para Y. La entropía del evento conjunto es:

$$H(X,Y) = -\sum_{x,y} \Pr(x, y) \log \Pr(x, y)$$

Asimismo, se define la entropía condicional de un evento y, $H(Y|X)$, como la entropía promedio de Y para cada valor de X, pesada con la probabilidad de obtener ese valor en X:

$$H(Y|X) = -\sum_{x,y} \Pr(x, y) \log \Pr(y|x)$$

La entropía condicional expresa el promedio de incertidumbre sobre el valor de *Y* dado algún valor de *X*. Se ha demostrado que

$$H(Y|X) = H(X,Y) - H(X)$$

La información mutua *I(X, Y)* es la cantidad de información ganada acerca de *X* cuando se conoce *Y* y viceversa:

$$I(X,Y) = H(Y) - H(Y \mid X) = H(X) + H(Y) - H(X,Y)$$

que es equivalente a

$$I(X,Y) = \sum_{x,y} \Pr(x, y) \log \frac{\Pr(x, y)}{\Pr(x) \Pr(y)}$$

Igualmente, la información mutua condicional en una variable *Z* es igual a

$$I(X,Y|Z) = H(X|Z) - H(X|Y,Z)$$

y esto se reduce a

$$I(X,Y|Z) = \sum_{x,y,z} p(x, y, z) \log \frac{p(x, y, z)}{p(x|z)\, p(y|z)}$$

Cuando dos variables *X* y *Y* son independientes, el conocimiento del valor que toma *X* no influye en el conocimiento que se tiene sobre el valor de *Y*. En otras palabras, dado que *X* y *Y* son independientes, *I(X,Y)*=0.

3.2 Conceptos de modelado con grafos

Existen grafos que representan relaciones de dependencia (D-Map) e independencia (I-Map) [Pearl, 1988], en donde cada nodo en el grafo representa una variable. En el caso de los mapas de dependencia se asegura que, si dos nodos están conectados, existe una relación de dependencia entre las variables a las que representan, de modo que un grafo vacío es el caso trivial de un D-Map. Los mapas de independencia aseguran que si dos nodos no están conectados, entonces las variables a las que representan son independientes, de modo que un grafo completo es un caso trivial de un I-Map. Cuando un grafo es un D-Map y un I-Map se dice que es un *Mapa Perfecto*.

Existen modelos de grafos que utilizan arcos dirigidos, no dirigidos y ambos. En modelos representados por grafos mixtos (con arcos dirigidos y no dirigidos), los arcos dirigidos representan una dependencia causal y los no dirigidos representan una dependencia

contemporánea [Eichler, 2001]. Los grafos dirigidos representan comúnmente relaciones de causalidad, por lo que en ocasiones son llamados grafos causales.

En los modelos de grafos que utilizan arcos dirigidos se hacen necesarias algunas definiciones:

- Un vértice *A* es un *Padre* de un vértice *B* y *B* es un *Hijo* de *A* si y solo si existe un arco dirigido de *A* a *B*.
- Un *Ancestro* de un vértice *V* es cualquier vértice *W* tal que existe un camino dirigido de *W* a *V*.
- Un *Descendiente* de un vértice *V* es cualquier vértice *W* tal que existe un camino dirigido de *V* a *W*.
- *Padres(V)* es el conjunto de padres del vértice *V*.
- *Hijos(V)* es el conjunto de hijos del vértice *V*.
- *Ancestros(V)* es el conjunto de ancestros del vértice *V*.
- *Descendientes(V)* es el conjunto de descendientes del vértice *V*.

Otras definiciones importantes son aquellas que se refieren a los nodos de aristas convergentes. Para grafos dirigidos se tienen las siguientes definiciones [Sprites et. al., 2000]:

- Un vértice *V* en un grafo *G* es un *nodo de aristas convergentes* (*collider*) en un camino no dirigido π si y solo si existen dos distintos arcos en π incidentes a *V*.
- Un vértice *V* en un grafo *G* es un *nodo de aristas convergentes sin blindaje* en π si *V* es un nodo de aristas convergentes en π, *V* es adyacente a dos vértices V_1 y V_2, $V_1 \neq V_2$, y V_1 y V_2 no son adyacentes en *G*.

En el grafo mostrado en la figura 3.1, *B* y *E* son nodos de aristas convergentes.

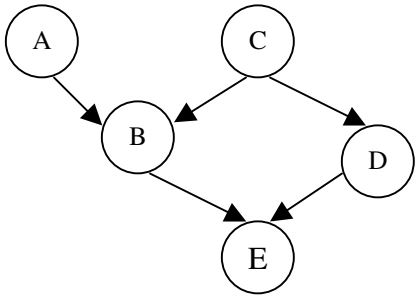


Figura 3.1 Grafo dirigido

En el caso de grafos no dirigidos, un nodo de aristas convergentes se puede definir de la siguiente manera [Eichler, 2001]:

Sea $G=<V, E>$ un grafo mixto, y $\pi=<e_1, e_2, ..., e_n>$ un camino con arcos $e_i=v_{i-1}-v_i$. Un vértice intermedio v_i, $1 \leq i \leq n-1$ es un nodo de aristas convergentes en π si se encuentran, ya sea dos arcos incidentes, o un arco incidente y un arco no dirigido en v_i.

Note que esta definición está referida a un camino. En el grafo mixto de la figura 3.2, *B* es un nodo de aristas convergentes en el camino *A*-*B*-*C* (ó *C*-*B*-*A*), al igual que *E* en el camino *B*-*E*-*D* (ó *D*-*E*-*B*). También *D* es un nodo de aristas convergentes en el camino *C*-*D*-*E* (ó *E*-*D*-*C*).

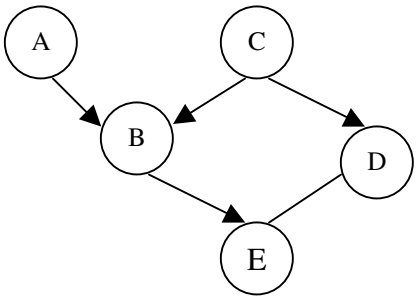


Figura 3.2 Grafo mixto

Existen condiciones que permiten caracterizar a un modelo de grafo. Una de las condiciones básicas para estos modelos es la condición de Markov. Para grafos dirigidos, la condición de Markov se puede expresar como sigue [Sprites et. al, 2000]:

Condición de Markov. Un grafo acíclico dirigido *G* sobre *V* con una distribución de probabilidad *P(V)* satisface la condición de Markov si y solo si, para todo $W \in V$, *W* es independiente de $V \setminus (Descendientes(W) \cup Padres(W))$ dado Padres(*W*).

En base a la Condición de Markov es posible definir la Condición de Minimalidad. Esta condición identifica a aquellos grafos que tienen el mínimo número de arcos para cumplir la condición de Markov, eliminando así los casos triviales:

Condición de Minimalidad. Sea *G* un grafo acíclico dirigido sobre *V*, y *P* una distribución de probabilidad sobre *V*, <*G*, *P*> satisface la Condición de Minimalidad si y solo sí para cada subgrafo propio *H* de *G* con vértices *V*, <*H*, *P*> no satisface la Condición de Markov.

Además de las relaciones de independencia exigidas por la condición de Markov, existen otras que se pueden expresar en un grafo. La *d-separación* es una de las relaciones de independencia más importantes en la representación de distribuciones de probabilidad con grafos. Básicamente, la d-separación obedece a la independencia de dos variables dada una tercera en la distribución de probabilidad representada por el grafo.

d-separación. Sea un grafo *G*. Si *X* y *Y* son vértices en *G*, $X \neq Y$, y sea *W* un conjunto de vértices en *G* que no contiene a *X* ni a *Y*, entonces *X* y *Y* están d-separados dado *W* si y solo si no existe un camino no dirigido π entre *X* y *Y*, tal que:

- Todo nodo de aristas convergentes en π tiene un descendiente en *W*
- Ningún otro vértice en π está en *W*

En el grafo de la figura 3.2, *A* y *C* están d-separados dado {*D*} o dado {}, pero están d-conectados dado {*B*}, dado {*E*} y también dado {*B*, *E*}. Así mismo, están d-separados dado {*E*, *D*}.

Una definición alternativa de d-separación se basa en el concepto de *camino activo* (también conocido como *camino abierto*):

Camino activo. Sea π un camino entre dos vértices A y B, π es un camino activo respecto a un conjunto de vértices W si y solo si todo vértice en π está activo respecto a W.

Vértice activo. Un vértice V está activo respecto a un conjunto de vértices W si y solo si se cumple alguna de las siguientes dos condiciones:

- V es un nodo de aristas convergentes y $V \in W$ o
- V no es un nodo de aristas convergentes y $V \notin W$

Así, la d-separación se puede definir de la siguiente manera:

d-separación. Dos vértices X y Y están d-separados dado un conjunto de vértices W si y solo si no existe un camino no dirigido entre X y Y que esté activo respecto a W.

La condición que permite saber si una distribución es representada por un grafo es la condición de fidelidad:

Condición de Fidelidad. Sea G un grafo causal y P una distribución de probabilidad generada por G. $<G, P>$ satisface la Condición de Fidelidad si y solo si toda relación de independencia condicional presente en P es implicada por la Condición de Markov aplicada a G.

Una distribución P es fiel a G si y solo si satisface tanto la Condición de Markov como la Condición de Fidelidad. Una condición de fidelidad más estricta es la condición de fidelidad normal para grafos acíclicos dirigidos [Cheng et. al., 1997]:

Fidelidad normal para grafos acíclicos dirigidos. En un modelo de probabilidad fiel a un grafo acíclico dirigido, para cualesquiera dos nodos que están conectados por al menos dos caminos de adyacencia, bajo una situación arbitraria en la que algunos caminos entre los dos nodos no están activos respecto a un conjunto W, si solo es posible aumentar la información mutua activando cualquiera de los caminos previamente no activos entre los dos vértices, sin desactivar cualquier camino previamente activo, se dice que este modelo de probabilidad es normalmente fiel a un grafo acíclico dirigido.

Esta última condición es comúnmente utilizada por técnicas de extracción de Redes Bayesianas para especificar el tipo de modelo con el cual se puede trabajar.

Cuando se estudian los modelos con grafos y se desean expresar relaciones de causalidad, se utiliza la *causalidad de Granger* [Granger, 1969]. La causalidad de Granger establece que una serie de tiempo es causal de otra si la predicción de esta última se ve afectada al tomar en cuenta la información contenida en la primera serie. Formalmente se define como sigue:

Causalidad de Granger. Considérese una distribución condicional respecto a dos conjuntos de información disponibles en el tiempo t, digamos I_t e $I_t^*=\{I_t, x_t, x_{t-1}, ...\}$, en donde x_t denota a una variable posiblemente causal. La variable x_t es definida como causal de la variable y_t en el sentido de Granger si existe un $h\in\{1, 2, ...\}$ tal que:

$$E\left(y_{t+h}\middle|I_t\right)\neq E\left(y_{t+h}\middle|I_t^*\right)$$

En la gran parte de la literatura sobre modelos de grafos [Dahlhaus & Eichler, 2000], el símbolo $\perp$ representa ortogonalidad (no correlación) entre variables. Asimismo, $A \perp B \mid Z$ significa que A y B son ortogonales una vez que se eliminan los efectos lineales de Z.

Si V representa un conjunto de variables y $A\subseteq V$, entonces $X_A=\{X_A(t)\}$ representa un proceso multivariado dado por los las variables en A, y $\overline{X_A}(t)=\{X_A(s), s<t\}$ denota el pasado del subproceso X_A.

Pese a que se ha definido el concepto de causalidad de Granger, es necesario definir conceptos de no causalidad más específicos:

No causalidad. X_a es no causal para X_b, relativo al proceso X_V, y se denota como, $X_a \not\rightarrow X_b$ $[X_V]$ si:

$$X_b(t)\perp \overline{X}_a(t)\middle|\overline{X}_{V\setminus\{a\}}(t)$$

No correlación parcial contemporánea. X_a y X_b están parcialmente no correlacionadas de forma contemporánea en relación a un proceso X_V, denotado $X_a \not\sim X_b$ si:

$$X_a(t)\perp X_b(t)\middle|\overline{X}(t), X_{V\setminus\{a,b\}}(t)$$

3.3 Redes Bayesianas

Uno de los modelos de grafo dirigido más utilizados son las Redes Bayesianas. Una Red Bayesiana es un grafo acíclico dirigido (DAG) en donde cada nodo representa una variable aleatoria, y cada arco representa una relación de dependencia entre las variables involucradas. Una variable X_i tiene un arco incidente de una variable X_j si y solo si X_i depende de X_j. Los modelos de Red Bayesiana tienen la ventaja de ser ampliamente intuitivos. Tanto la estructura como las distribuciones son fácilmente comprensibles para casi cualquier observador. Los cálculos hechos sobre la red son transparentes y es posible verificarlos de manera relativamente sencilla.

La figura 3.3 muestra un ejemplo de una Red Bayesiana. En esta red está representado el hecho de que el mes del año tiene influencia tanto en el clima como en periodos vacacionales. A su vez, el clima tiene influencia sobre el tipo (o número) de enfermedades

que se presentan, y tanto las enfermedades como los periodos vacacionales tienen un efecto sobre la productividad de las empresas.

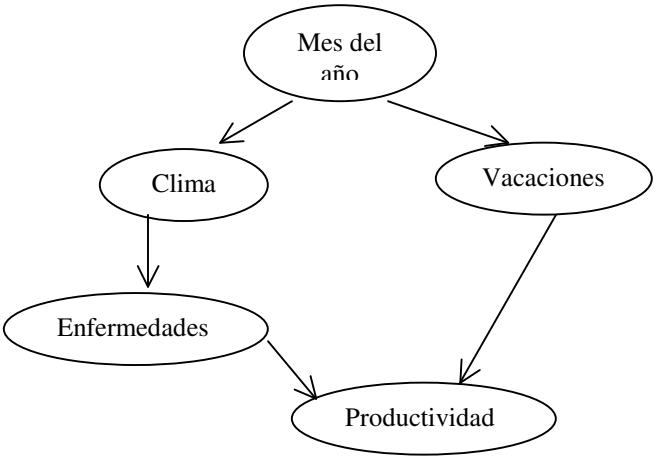


Figura 3.3 Ejemplo de una Red Bayesiana

Cada nodo en la Red Bayesiana tiene asociada una distribución de probabilidad que depende solamente de sus padres. Es posible calcular la distribución conjunta a partir de la Red Bayesiana utilizando algunas leyes de probabilidad, de modo que cualquier cálculo que utilice la distribución conjunta de probabilidad puede realizarse a partir de la red. Además, una Red Bayesiana permite realizar procesos razonamiento tales como predicción y abducción utilizando reglas de probabilidad [Pearl & Rusell, 2000].

Por ejemplo, sea *V=Vacaciones*, *M=Mes_del_año*, *E=Enfermedades*, *C=Clima* y *D=*P*roductividad*. Suponga que se desea calcular la probabilidad de cada posible valor para la variable *Vacaciones*. La densidad de probabilidad condicional de esta variable tiene la forma:

$$\Pr(V=Si \mid M=Enero) = p_{1,1}$$
$$\Pr(V=Si \mid M=Febrero) = p_{1,2}$$
$$\vdots$$
$$\Pr(V=No \mid M=Diciembre) = p_{2,12}$$

Por la regla de la cadena se sabe que

$$\Pr(V=u, M=v) = \Pr(V=u \mid M=v)\ \Pr(M=v)$$

además,

$$\Pr(V = u) = \sum_{v} \Pr(V = u, M = v)$$

de donde se puede obtener la distribución de probabilidad marginal para la variable *Vacaciones*:

$$\Pr(V=\text{Si})=p_1$$
$$\Pr(V=No)=p_2$$

Ahora, suponga que se conoce que la productividad de la empresa fue alta. La densidad de probabilidad para la variable *Vacaciones* se puede obtener utilizando la Regla de Bayes:

$$\Pr(V=u|D=v)=\frac{\Pr(D=v|V=u)\Pr(V=u)}{\Pr(D=v)}$$

En donde el valor de $\Pr(D=v|V=u)$ se encuentra en la tabla de densidad conjunta de probabilidad para la variable *D*. $\Pr(V=u)$ se calculó anteriormente, y $\Pr(D=v)$ se puede calcular siguiendo un procedimiento similar al utilizado para calcular $\Pr(V=u)$.

En caso de que existan nodos intermedios entre la variable conocida y la variable que se desea calcular, se pueden utilizar reglas de probabilidad para realizar el cálculo de manera recursiva. Por ejemplo, suponga que se desea calcular la probabilidad de que el mes del año haya sido *Enero* dado que se obtuvo una productividad *v*. Por reglas de probabilidad se sabe que:

$$\begin{aligned}
&\Pr(M=Enero|D=v)\\
&=\sum_{V,C,E}\Pr(M=Enero,V,C,E|D=v)\\
&=\sum_{V,C,E}\Pr(M=Enero|C,V)\Pr(C,V,E|D=v)\\
&=\sum_{V,C,E}\Pr(M=Enero|C,V)\Pr(V|D=v)P(C,E|D=v)\\
&=\sum_{V,C,E}\Pr(M=Enero|C,V)\Pr(V|D=v)\Pr(C|E)\Pr(E|D=v)
\end{aligned}$$

Existe un gran número de variantes de las Redes Bayesianas simples. Una de tales variantes consiste en considerar al tiempo como un parámetro importante para el cálculo en la red. La consideración del tiempo puede ser atacada de diversas formas. Por ejemplo, se han propuesto ideas tales como considerar al tiempo como una variable aleatoria [Berzuini, 1990] o hacer a las funciones de densidad de probabilidad dependientes del tiempo [Tawfik & Neufeld, 1994]. Sin embargo, es necesario observar que este tipo de inclusión del tiempo se realiza respecto al momento en que se calcula alguna probabilidad en la red, y no respecto a la relación temporal (si existe) entre las variables involucradas, como es el caso de algunos modelos de grafos. La expresión de relaciones temporales entre las variables probablemente tendría que realizarse durante la construcción de la Red Bayesiana, aunque ésta no proporciona algún medio para representar dichas relaciones.

Cuando se habla de una Red Bayesiana, se conoce como estructura al grafo dirigido que representa la relación de dependencia entre los nodos, y se conoce como parámetros a las densidades de probabilidad existentes en cada nodo. Es importante notar que las densidades de probabilidad que se obtienen para cada nodo son condicionales respecto a los valores de sus nodos "Padre", es decir, sea Padres*(X)*={Padre$_1$*(X)*, Padre$_2$*(X)*, ..., Padre$_n$*(X)*} el conjunto de nodos padre de *X*, la densidad de probabilidad de *X* es dependiente de Padres*(X)*:

$$X \sim f\left(X \middle| Padre_1(X), Padre_2(X), \text{K}, Padre_k(X)\right)$$

Por lo tanto, la extracción de los parámetros requiere necesariamente una estructura definida para la red.

Se han desarrollado varias técnicas para la extracción de la estructura de una Red Bayesiana a partir de datos. En general estos métodos se dividen en dos grandes clases: aquellos que se basan en la maximización de una función de adaptación para cada estructura y aquellos que se basan en el descubrimiento de relaciones de dependencia. Debido a que el espacio de búsqueda en todos los posibles grafos es demasiado grande, los métodos del primer grupo generalmente utilizan métodos de búsqueda heurísticos para encontrar la mejor estructura, como es el caso del algoritmo glotón utilizado por K2 [Cooper & Herskovits, 1992] o algoritmos genéticos [Etxeberria et. al., 1997].

Los métodos del segundo grupo [Eichler, 2001] comúnmente utilizan conceptos de la teoría de la información de Shannon, tales como entropía e información mutua. Una de las ventajas de este tipo de algoritmo es que, en general, son más rápidos debido a que no requieren el cálculo de los parámetros para cada configuración ni la evaluación de la probabilidad de un número considerable de casos en la Red Bayesiana. Además, no requieren aleatoriedad, lo que facilita la reproducción de resultados.

4. ESTADO DEL ARTE

Tanto la extracción de Redes Bayesianas a partir de Bases de Datos como la construcción de modelos de grafos para series de tiempo son áreas que han sido atacadas por varios investigadores alrededor del mundo. Asimismo, la parte de discretización de series de tiempo referente al tamaño de los intervalos de cuantificación ha seguido desarrollándose en los últimos años.

4.1 Discretización

La modulación por código de pulso (PCM, Pulse-Code Modulation) es un tipo de modulación desarrollada en el área de Comunicaciones Digitales, que sirve para cuantificar una señal analógica muestreada en cierto número de niveles discretos. Este tipo de modulación utiliza un código para designar cada nivel en cada tiempo de muestra [Stremler, 1993].

Si se desea cuantificar una señal utilizando M niveles discretos, en cada tiempo de muestra debe decidirse cuál de los ocho M niveles es la mejor aproximación a la señal. Eligiendo el valor más cercano, éste se mantiene hasta el siguiente tiempo de muestra. Este proceso de cuantificación introduce diferencias respecto al valor real de la señal, las cuales pueden considerarse como ruido. El aumento del número de niveles de cuantificación M tiende a reducir este ruido. La cuantificación puede ser *lineal* (con niveles igualmente espaciados) o *no lineal* (con niveles espaciados de manera no uniforme). A la diferencia que existe entre niveles contiguos se le llama *tamaño del intervalo de cuantificación*, y comúnmente se denota por la letra q.

Después de cuantificar la señal se asigna un dígito a cada nivel, de manera que exista correspondencia uno a uno entre los niveles y el conjunto de dígitos. Esto se llama digitalización de la señal y reduce la señal a un conjunto de dígitos en tiempos de muestra sucesivos, originando un sistema de modulación completamente digital.

En la transmisión de mensajes que tienen valores de muestra repetidos, la transmisión repetida representa un desperdicio de capacidad de comunicación, porque tales valores tienen un bajo contenido de información. Una forma de evitar esta situación es enviar sólo las diferencias entre valores de muestra sucesivos, codificadas en forma digital. Esto se conoce como Modulación de Código de Pulsos Diferencial (DPCM, Differential Pulse-Code Modulation).

Entre las desventajas de DPCM está el hecho de que, si se comete un error, se mantiene una polaridad incorrecta hasta que éste se corrija. Además, estos sistemas corren el riesgo de una sobrecarga por tasa de elevación debido a las operaciones de diferenciación y truncamiento. Por ejemplo, si dos muestras adyacentes difieren en más de $\pm M$ niveles, el sistema podría enviar solamente $\pm M$, y la sobrecarga resultante podría causar un error en la reconstrucción.

Muchos algoritmos de Machine Learning requieren un espacio discreto de valores de entrada. La digitalización de señales es similar a la discretización de series de tiempo. En este sentido, podrían utilizarse las técnicas PCM y DPCM para discretizar series de tiempo que sirvan como entrada a algún algoritmo que trabaje con valores discretos. Sin embargo, los esfuerzos referentes a técnicas de discretización para Machine Learning no suponen que se trabaje con series de tiempo, de modo que cuando existe un valor continuo de entrada, éste no se encuentra directamente relacionado con el anterior y por lo tanto no es posible aplicar un algoritmo de tipo diferencial, tal como DPCM. Debido a esto, las técnicas de discretización en este campo se concentran en la definición de tamaños de intervalo óptimos.

4.1.1 Definición de tamaño de intervalos

Los métodos de discretización en Machine Learning se pueden clasificar en *globales* y *locales*, *supervisados* y *no supervisados*, y en *estáticos* y *dinámicos* [Dougherty et. al., 1995].

Los métodos locales producen particiones que son aplicadas a regiones localizadas del espacio de instancias. Los métodos globales producen una malla sobre el espacio continuo n-dimensional completo de instancias, y cada variable es discretizada en regiones independientes de las demás. La malla contiene $\prod_{i=1}^{n} k_i$ regiones, en donde cada k_i es el número de particiones en la variable *i*-ésima.

Los métodos no supervisados son aquellos que no hacen uso de etiquetas de instancia en el proceso de discretización, es decir, no existe un conocimiento *a-priori* acerca de valores representativos para las clases a formar. Los métodos supervisados son aquellos que hacen uso de tales etiquetas.

Los métodos estáticos ejecutan una etapa de discretización de los datos para cada variable, y determinan el número de intervalos k a producir de manera independiente para cada una de ellas. En contraste, los métodos dinámicos llevan a cabo una búsqueda sobre todo el espacio de posibles valores de k para todas las variables en forma simultánea, capturando interdependencias durante la discretización.

El método de discretización más simple se conoce en Machine Learning como *Intervalo de Ancho Fijo*, y es equivalente a PCM con cuantificación lineal. Este método simplemente divide el intervalo de valores observados en k cajones de igual tamaño, en donde k es un parámetro definido por el usuario.

El método de *Intervalos de Igual Frecuencia* divide una variable continua en k cajones, de forma que si se tienen m valores, cada cajón contendrá $\frac{m}{k}$ valores adyacentes (posiblemente repetidos). Una variación de este método, *Máxima Entropía Marginal* [Chmielewski & Grzymala-Busse, 1994], ajusta las fronteras de modo que se disminuya la entropía dentro de cada intervalo.

El algoritmo *1R* [Holte, 1993] intenta dividir el dominio de cada variable continua en cajones, cada uno conteniendo elementos que, en su mayoría, pertenecen a una clase particular, con la condición de que cada cajón debe incluir al menos un número previamente especificado de instancias.

El sistema ChiMerge [Kerber, 1992] provee un método heurístico estadísticamente justificado para discretización supervisada. Este algoritmo comienza colocando cada valor continuo en su propio intervalo, y procede usando la prueba χ^2 para determinar cuando dos intervalos adyacentes deben ser combinados. Este método prueba la hipótesis de que dos intervalos adyacentes son estadísticamente independientes, realizando una medición empírica de la frecuencia esperada de las clases representadas en cada uno de los intervalos. El proceso de combinación se controla utilizando un umbral de χ^2, que indica el máximo valor de χ^2 que garantiza la combinación de dos intervalos.

Antes de continuar con el método StatDisc es necesario explicar la Φ-medida, que se utiliza para cuantificar fluctuaciones. Sea $X=\{X_1, X_2, ..., X_K\}$, en donde cada $X_k = \{x_{k_1}, x_{k_2}, \mathrm{K}, x_{k_{N_k}}\}$, k=1...K. Sea $z_{k_i} \equiv x_{k_i} - \bar{x}_k$, en donde $\bar{x}_k$ es el promedio de los valores para el conjunto X_k. Se define la variable Z_k como $Z_k \equiv \sum_{i=1}^{N_k} (x_{k_i} - \bar{x}_k)$. Por construcción, el promedio de Z_k sobre todos los valores de X_k, denotado por $\langle Z \rangle$, es igual a cero. $\langle N \rangle$ denota el promedio del número de valores N_k sobre todos los conjuntos X_k. Se define la Φ-medida [Mrowczynski, 2000] como:

$$\Phi \equiv \sqrt{\frac{\langle Z^2 \rangle}{\langle N \rangle}} - \sqrt{\overline{z^2}}$$

El método StatDisc [Richeldi & Rossotto, 1995] crea una jerarquía de intervalos de discretización utilizando la Φ-medida como criterio para combinar intervalos. Este método puede combinar a la vez un número de intervalos previamente especificado, a diferencia de ChiMerge que solo puede combinar dos a la vez. La combinación de intervalos continúa hasta que se alcanza un umbral Φ. Entonces, se puede explorar la jerarquía final de discretizaciones, y se puede seleccionar automáticamente una discretización adecuada.

Existen varios métodos basados en la entropía. Uno de ellos [Chiu et. al., 1990] propone un método de discretización basado en maximizar la entropía de Shannon sobre el espacio discretizado, utilizando un algoritmo de escalada de la colina para encontrar una partición inicial, y posteriormente aplicando de nuevo el método a intervalos particulares para obtener intervalos más finos. El algoritmo D-2 utiliza discretización basada en entropía en dominios de árboles de decisión. Otro algoritmo [Fayyad & Irani, 1993] utiliza una heurística de minimización recursiva de la entropía, y la acopla con un criterio de simplificación de resultados llamado *Minimum Description Length* para controlar el número de intervalos producidos sobre el espacio continuo. Por último, existe un algoritmo [Pfahringer, 1995] que utiliza entropía para seleccionar un gran número de puntos

candidatos para partición, y emplea una búsqueda de "primero el mejor" con una heurística Minimum Description Length para encontrar una discretización óptima.

El método *Adaptive Quantizers* [Chan et. al., 1991] combina discretización supervisada y no supervisada. Se comienza con una partición binaria de anchos de intervalos iguales. Entonces, se induce un conjunto de reglas de clasificación en los datos discretizados y se prueba su desempeño al predecir salidas discretizadas. El intervalo que produjo la peor discretización se divide entonces en dos particiones del mismo ancho, y se repite el proceso de inducción y evaluación hasta que se alcanza un criterio de desempeño.

Un par de métodos que utilizan discretización supervisada y no supervisada se conoce de manera genérica como *Monothetic Contrast Criterions* (MCC) [Van de Merckt, 1993]. El primer método hace uso de un algoritmo de agrupamiento no supervisado que busca las fronteras de la partición que produzcan el mayor contraste, de acuerdo a una función de contraste. El segundo método simplemente redefine las funciones objetivo a ser maximizadas dividiendo la función de contraste anterior por la entropía de una partición propuesta.

El algoritmo de *Maximización del Valor Predictivo* [Weiss et. al., 1990] hace uso de un método de discretización supervisado para encontrar las fronteras de una partición con valores predictivos localmente máximos, es decir, aquellos con mayor probabilidad de realizar una clasificación correcta. La búsqueda de tales fronteras comienza de manera burda, y se refina para encontrar las fronteras de partición localmente óptimas.

La Cuantificación por Vectores [Kohonen, 1989] consiste en particionar un espacio continuo N-dimensional en un *Mosaico de Voronoi* (*Voronoi Tessellation*), y representar cada punto por la región en la que cae. Como éste método crea regiones locales, se considera un método de discretización local.

Dado un conjunto de *N* puntos en un plano, un Mosaico de Voronoi divide el dominio en un conjunto de regiones poligonales, las fronteras de las cuales son bisectrices perpendiculares de las líneas que unen los puntos.

4.1.2 Algoritmos de agrupamiento

El objetivo de un algoritmo de agrupamiento es, dado un conjunto de *n* objetos descritos a través de *m* rasgos, crear particiones de éste conjunto. Las agrupaciones formadas deben cumplir que la semejanza de los objetos dentro de una agrupación sea máxima, mientras que la semejanza de los objetos pertenecientes a agrupaciones diferentes sea mínima [Gil & Badía, 2002]. Existen varias clasificaciones para los algoritmos de agrupamiento, entre las que se encuentran:

Jerárquicos / Por particiones. Un algoritmo de agrupamiento jerárquico forma una secuencia de particiones, en la que cada partición está anidada en la siguiente. Los algoritmos de agrupamiento por particiones generan una sola partición.

Aglomerativos / Divisionales. Los algoritmos aglomerativos comienzan colocando a cada objeto en su propio agrupamiento, y gradualmente fusionan los agrupamientos hasta cumplir alguna condición. Los algoritmos divisionales comienzan colocando todos los objetos en un solo agrupamiento, y gradualmente dividen éste agrupamiento. Note que en ambos casos el agrupamiento es jerárquico.

"Monothetic" / "Polythetic". Es común que los objetos a agrupar estén descritos por varias características. Un algoritmo de agrupamiento *monothetic* utiliza las características una a una, mientras que un algoritmo *polythetic* utiliza todas las características al mismo tiempo.

Incrementales / Simultáneos. Los algoritmos incrementales actualizan los agrupamientos a medida que cambia el conjunto de datos, sin necesidad de repetir proceso completo. Los algoritmos simultáneos trabajan con todo el conjunto de datos al mismo tiempo.

Estáticos / Dinámicos. Los algoritmos estáticos producen un número de agrupaciones previamente definido. Los algoritmos dinámicos crean las agrupaciones en base a los datos, por lo que el número de agrupaciones es variable.

Una familia importante de algoritmos de agrupamiento son aquellos basados en centroide. Entre ellos, el más utilizado es *K-Medias* [McQueen, 1967], el cual es un algoritmo estático no jerárquico. Este algoritmo inicialmente toma K objetos y crea K agrupaciones conteniendo un objeto cada una. A continuación, toma cada uno de los objetos restantes y los asigna a aquella agrupación cuyo centroide geométrico se encuentre a la menor distancia. La posición del centroide se calcula cada vez que un componente es añadido a la agrupación, continuando así hasta que todos los objetos han sido asignados a las agrupaciones finales.

El algoritmo K-Medias depende enormemente de los primeros K objetos que se toman al inicio. Una posibilidad consiste en utilizar aquellos K objetos cuya distancia sea máxima, en cuyo caso el algoritmo es simultáneo. Otra posibilidad, la más usual, consiste en tomar los K objetos de manera aleatoria, en cuyo caso el algoritmo es incremental y aleatorio.

Una variación de los algoritmos basados en centroides son los algoritmos basados en medoides, en los cuales cada agrupamiento está representados por un objeto dentro del mismo. El algoritmo más representativo de ésta clase es PAM.

PAM (Partitioning Around Medoids) fue desarrollado para encontrar los K objetos más representativos (medoides) que reprenten K agrupamientos, de tal modo que los objetos no seleccionados se encuentren agrupados con el medoide más similar a ellos. La distancia total entre objetos no medoide y los medoides puede ser reducida intercambiando un medoide por otro objeto de manera iterativa. Este proceso puede ser largo, aún para conjuntos de objetos de tamaño moderado y pocos medoides.

CLARA (Clustering Large Applications) fue desarrollado para superar los problemas de complejidad computacional de PAM. En lugar de encontrar objetos representativos para el conjunto de datos completo, CLARA reduce la complejidad tomando múltiples muestras de

los objetos y aplicando PAM en cada muestra. Los medoides finales son obtenidos del mejor resultado de esas múltiples muestras.

CLARANS (Clustering Large Applications based on RANge Search) [Ng & Han, 2002] está formalizado como búsqueda a través de un grafo, en donde cada objeto está representado por un nodo. CLARANS comienza eligiendo *K* nodos al azar, midiendo la distancia total existente entre los objetos y su respectivo medoide. Posteriormente fija K-1 medoides y elige un nuevo medoide al azar para el agrupamiento restante. Si la distancia total entre los objetos y sus medoides disminuye, almacena el nodo actual como un mínimo local. El proceso se reinicia con otro nodo elegido al azar, y repite la búsqueda de un nuevo mínimo local hasta cumplir algún criterio.

CLASA (Clustering Large Applications based on Simulated Annealing) genera medoides aplicando el método de recocido simulado. La colección de *K* medoides se toma como un estado.

MCMRS (Multi-Centroid, Multi-Run Sampling Scheme) es un algoritmo que aprovecha el hecho de que es mucho más rápido calcular los centroides que los medoides. Busca los centroides utilizando algún algoritmo conocido para esto, y encuentra aquellos nodos cercanos a los centroides. Posteriormente realiza una búsqueda sobre los nodos elegidos para encontrar los medoides.

IMCMRS (Incremental Multi-Centroid, Multi-Run Sampling Scheme) es una variación de MCMRS. Genera múltiples agrupamientos basados en centroides, y para cada agrupamiento, para cada centroide, elige como medoide al nodo más cercano al centroide. Este proceso itera un número determinado de veces.

Otra familia de algoritmos son aquellos basados en densidad. Este tipo de algoritmos tiene la capacidad de detectar agrupaciones con formas arbitrarias, como las que se muestran en la figura 4.1. Entre los algoritmos más representativos de esta familia se encuentra *DBScan* [Ester et. al., 1996].

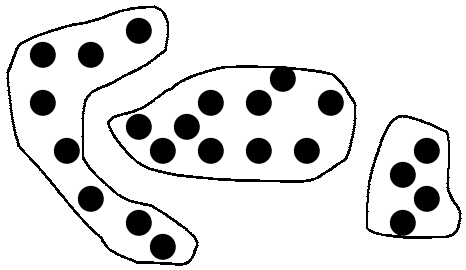


Figura 4.1 Agrupaciones con formas arbitrarias

DBScan define a un objeto *p* como directamente alcanzable desde un objeto *q* si la distancia entre ellos es menor a un umbral, y el objeto *p* contiene más de un número específico de objetos cercanos a él. Posteriormente, define a un objeto *p* como alcanzable por densidad si existe una secuencia de objetos p_1, p_2, ..., p_n tales que $p=p_1$, $q=p_n$, y p_{i+1} es

directamente alcanzable desde p_i. Por último define a un objeto p como conectado por densidad con un objeto q si existe un objeto o tal que p y q son alcanzables por densidad desde o. Las agrupaciones se forman por aquellos objetos que están conectados por densidad.

Uno de los algoritmos de agrupamiento que ha producido mejores resultados es *Chameleon* [Karypis et. al, 1999]. Este es un algoritmo dinámico, jerárquico, aglomerativo y simultáneo que modela a los objetos como vértices de un grafo. De este modo, es capaz de tomar en cuenta tanto interconectividad como cercanía entre agrupaciones. El proceso de agrupamiento consta de 3 etapas. Inicialmente Chameleon crea la matriz de distancias entre los objetos, y genera un grafo en donde cada objeto es un nodo, y existe un arco entre dos objetos p y q si p está entre los k vecinos más cercanos a q. Posteriormente, los grafos se particionan de modo que se minimice el número de arcos de bipartición. Por último, combina repetidamente los pequeños subgrafos utilizando mediciones conocidas como interconectividad relativa y cercanía relativa, ambas definidas en el artículo en donde se presenta.

4.2 Extracción de grafos a partir de series de tiempo

La noción de que una serie de tiempo puede ser una versión retrasada de otra se ha desarrollado en diversas áreas. Una de ellas es astronomía, en donde se han desarrollado modelos de series de tiempo como funciones lineales de otra serie retrasada en tiempo [Scargle, 2001]:

$$X(t) = S(t) + B(X)$$
$$Y(t) = aS(t-\tau) + B(Y)$$

En donde S es una señal desconocida, X y Y son series de tiempo muestreadas a partir de S, B es una función que representa al ambiente, y τ representa el retraso existente entre estas dos series. Se utilizan métodos bayesianos para descubrir los parámetros a y τ. Específicamente, para descubrir τ se construye la matriz de correlación entre X y Y.

En química, otro trabajo [Arkin et. al., 1997] ha desarrollado la idea de que los retrasos entre las series de tiempo pueden indicar relaciones de causalidad. En dicho trabajo se intentaron deducir las interacciones en redes de reacciones químicas complejas a través de series de tiempo medidas experimentalmente, provenientes de especies que componen el sistema. Para ello, se aplicó el análisis de escalamiento multidimensional (MDS) y algoritmos heurísticos a funciones de correlación retrasadas en tiempo, provenientes de las series de tiempo, obteniéndose un diagrama a partir del cual se pueden deducir las interacciones entre las especies. Para crear dicho diagrama se consideró que, si la serie de tiempo para una especie dada tenía una correlación máxima cuando el retraso era negativo respecto a una serie de tiempo de referencia, entonces esa especie recibía las señales de entrada después de la especie de referencia. Similarmente, si las dos series estaban máximamente correlacionadas cuando el retraso era cero, pero la correlación tendía a retrasos negativos, la variación en la especie seguía a la variación en la especie de

referencia, y si los retrasos eran positivos entonces la variación en la especie precedía a la variación en la especie de referencia.

Esta idea ha sido retomada en el campo de modelado de relaciones entre series de tiempo mediante grafos. Se ha propuesto un concepto de causalidad basada en el hecho de que un efecto no puede preceder temporalmente a su causa [Dahlhaus & Eichler, 2000], permitiendo así descubrir la dirección de los arcos en un grafo dirigido. Es precisamente este concepto de causalidad el que da fundamento a la alineación de las secuencias discretas presentada en esta tesis.

Existen principalmente tres tipos de modelos de grafos. En el primero, una variable $X(t)$ en un tiempo específico t está representada por un vértice separado en el grafo. Dentro de esta clase se encuentran los grafos de cadena (*chain graphs*). En el segundo tipo, cada vértice representa a una variable, lo que lleva a un modelo más general de la estructura de dependencias entre las series de tiempo. Dentro de esta clase se encuentran los grafos de causalidad de Granger (*Granger Causality Graphs*). El tercer tipo de modelo consta de grafos no dirigidos, en los que los arcos reflejan dependencia entre variables. Los grafos de correlación parcial (*Partial Correlation Graphs*) son modelos de este tipo.

En las siguientes definiciones se utilizan los conceptos de *Causalidad de Granger*, *No causalidad* y *No correlación parcial contemporánea*, descritos en la sección 3.2.

Un grafo de cadena de series de tiempo (Time Series Chain Graph), denotado como grafo TSC, se define como:

Grafo de cadena. El grafo de cadena de un proceso estacionario X es un grafo $G_{TS}=(V_{TS}, E_{TS})$ con $V_{TS} = V \times Z$ tal que:

- $(a,t-u) \rightarrow (b,t) \notin E_{TS} \leftrightarrow u \geq 0 \vee X_a(t-u) \perp X_b(t) \left| \overline{X_V}(t) \setminus \{X_a(t-u)\} \right.$
- $(a,t-u) \text{—} (b,t) \notin E_{TS} \leftrightarrow u \neq 0 \vee X_a(t) \perp X_b(t) \left| \overline{X_V}(t) \cup \{X_{V \setminus \{a,b\}}(t)\} \right.$

En donde $\overline{X}(t) = \{X(s), s \leq t\}$. Dado que el proceso X es estacionario, se tiene que (a, t) —$(b, t) \notin T_{ST}$ si y solo sí (a, s) —$(b, s) \notin T_{ST}$.

Los grafos de causalidad de Granger son grafos mixtos cuyo conjunto de vértices es igual al conjunto de variables. Para los arcos dirigidos se utiliza la noción de causalidad de Granger, mientras que para los arcos no dirigidos se utiliza la misma noción que para los grafos de cadena.

Grafo de causalidad. Un grafo de causalidad de un proceso estacionario X es un grafo mixto $G_C=(V, E_C)$ tal que para todo $a, b \in V$ con $a \neq b$

- $a \rightarrow b \notin E_C \Leftrightarrow X_a \not\rightarrow X_b [X_V]$
- $a \text{—} b \notin E_C \Leftrightarrow X_a \not\text{—} X_b [X_V]$

Implícitamente se asume que cada componente depende de su propio pasado, lo cual quedaría expresado por un bucle en cada nodo. Dado que tales bucles no afectan las propiedades de separación del grafo, se omiten para obtener mayor simplicidad.

La relación entre un grafo de cadena $G_{TS}=(V_{TS}, E_{TS})$ y un grafo de causalidad $G_C=(V, E_C)$ se establece de la siguiente manera:

Agregación. Sean G_C y G_{TS} el grafo de causalidad y el grafo de cadena respectivamente, ambos pertenecientes a un proceso estacionario X. Entonces se tiene que:

- $a \to b \notin E_C \Leftrightarrow (a, t\text{-}u) \to (b, t) \notin E_{TS} \quad \forall u>0 \quad \forall t \in \mathbf{Z}$
- $a - b \notin E_C \Leftrightarrow (a, t) - (b, t) \notin E_{TS} \quad \forall t \in \mathbf{Z}$

Otro tipo de modelo son los grafos de correlación parcial (Partial Correlation Graph), comúnmente referidos como grafos PC. Este tipo de modelo tiene un concepto de separación simple, y permite conclusiones adicionales acerca de la estructura de dependencia de las series. Se define como:

Grafo de correlación parcial. El grafo de correlación parcial $G_{PC}=(V, E_{PC})$ de un proceso estacionario X está dado por

$$a - b \notin E_{PC} \Leftrightarrow X_a \perp X_b \,|\, X_{V\setminus\{a,b\}}$$

Es posible caracterizar el grafo de correlación parcial en términos de la inversa de la matriz espectral del proceso.

Sean X_a, X_b dos series de tiempo, la densidad de cross-covarianza con desplazamiento u se define como [Dahlhaus et. al., 1997]:

$$q_{ab} = \frac{\operatorname{cov}(dX_a(t), dX_b(t+u))}{dt\, du}$$

En donde $dX_a = X_a(t+dt)\text{-}X_a(t)$. el cross-espectro $f_{ab}(\lambda)$ se define como la Transformada de Fourier de la densidad de cross-covarianza:

$$f_{ab}(\lambda) = \frac{1}{2\pi} \int_{-\infty}^{\infty} q_{ab}(u) e^{-i\lambda u}\, du$$

De manera similar, la densidad de autocovarianza se define como:

$$q_{aa} = \frac{\operatorname{cov}(dX_a(t), dX_a(t+u))}{dt\, du}$$

Y el autoespectro se define como la Transformada de Fourier de la densidad de autocovarianza:

$$f_{aa}(\lambda) = \frac{p_a}{2\pi} + \frac{1}{2\pi}\int_{-\infty}^{\infty} q_{aa}(u)e^{-i\lambda u}\,du$$

En donde $p_a = \dfrac{\Pr(dX_a(t)=1)}{dt}$ es la intensidad media de X_a.

Así, la matriz espectral de un conjunto de series de tiempo X_1, X_2, .. X_k se define como:

$$f(\lambda) = \begin{pmatrix} f_{11}(\lambda) & f_{12}(\lambda) & \Lambda & f_{1k}(\lambda) \\ f_{21}(\lambda) & f_{22}(\lambda) & \Lambda & f_{2k}(\lambda) \\ \mathrm{M} & \mathrm{M} & \mathrm{O} & \mathrm{M} \\ f_{k1}(\lambda) & f_{k2}(\lambda) & \Lambda & f_{kk}(\lambda) \end{pmatrix}$$

Sea $g(\lambda) = f(\lambda)^{-1}$ la inversa de la matriz espectral del proceso estacionario X, entonces se cumple que:

$$a \text{—} b \notin E_{PC} \Leftrightarrow g_{ab}(\lambda) = 0 \quad \forall\lambda \in [-\pi,\pi]$$

Por último, la asociación entre dos series de tiempo X_a y X_b se puede medir mediante la coherencia espectral $|R_{ab}(\lambda)|^2$, en donde:

$$R_{ab}(\lambda) = \frac{f_{ab}(\lambda)}{\sqrt{f_{aa}(\lambda)f_{bb}(\lambda)}}$$

La coherencia espectral toma valores entre cero y uno. Una coherencia espectral de cero para todas las frecuencias indica independencia lineal, mientras que un valor de uno indica una relación lineal perfecta.

Los modelos extraídos en esta tesis son, en cierta medida, similares a los grafos de causalidad. Entre las principales diferencias se encuentran que estos últimos pueden presentar ciclos dirigidos y que no cuentan con información de ningún tipo acerca de las características de las relaciones que se presentan.

4.3 Extracción de Redes Bayesianas a partir de Bases de Datos

El área de extracción de Redes Bayesianas se divide comúnmente en cuatro casos, dependiendo de la integridad de los datos y de la parte de la red (estructura o parámetros) que se desea aprender, como se muestra en la tabla 4.1.

	Estructura	***Parámetros***
Datos completos	Generación y evaluación de varias estructuras, análisis de dependencias	MLE, MAP
Datos incompletos	Combinaciones de métodos	EM, Muestreo de Gibbs

Tabla 4.1. Casos de extracción de Redes Bayesianas y algunas opciones

La estructura de una Red Bayesiana se refiere al grafo acíclico dirigido que representa las relaciones de dependencia/independencia entre las variables. Los parámetros se refieren a las distribuciones de probabilidad condicional que se encuentran en los nodos de la Red Bayesiana. Es importante recordar que, dado que la distribución de probabilidad condicional para cada nodo depende de los padres del mismo, la estimación de los parámetros de una Red Bayesiana requiere una estructura definida.

Cuando se trabaja con Bases de Datos, el problema de datos espurios o faltantes se convierte en una prioridad, por lo que se han desarrollado diversos métodos para estimarlos. Sin embargo, cuando se trabaja con series de tiempo, se pueden utilizar técnicas mucho más convenientes (por ejemplo, interpolación, modelado o predicción) para atacar este problema, por lo cual en este trabajo no se aborda el tema de datos incompletos.

4.3.1 Extracción de los parámetros

El caso más simple en la tabla 4.1 es la estimación de los parámetros θ de una Red Bayesiana cuando se cuenta con datos completos. En este caso, el enfoque más sencillo es utilizar el método *Maximum Likelihood Estimation* (MLE) [Hyvärinen et. al., 2001].

MLE asume que los parámetros desconocidos θ son constantes o que no existe información previa sobre ellos. Este estimador presenta propiedades asintóticas que lo hacen teóricamente una buena opción, especialmente cuando el número de muestras es grande.

La estimación $\hat{\theta}_{MLE}$ de los parámetros θ se elige de modo que maximice la función de probabilidad:

$$p(x_n|\theta) = p(x(1), x(2), \mathrm{K}\ , x(n)|\theta)$$

de las mediciones $X_n = \{x(1), x(2),..., x(n)\}$. El valor obtenido para $\hat{\theta}_{MLE}$ es aquel que maximiza la probabilidad de los valores $x(1), x(2),..., x(n)$.

La construcción de la función de estimación para MLE puede ser muy complicada si las mediciones son dependientes. Por lo tanto, es usual asumir que las mediciones $x(j)$ son independientes entre sí, con lo que se obtiene una expresión del tipo

$$p(X_n|\theta) = \prod_{j=1}^{n} p\left(x(j)|\theta\right)$$

A partir de la estadística suficiente se puede obtener una estimación MLE. Por ejemplo, si se tiene una variable aleatoria independientes de cualquier otra variable, con un espacio muestreal {*V*, *F*} y un conjunto de mediciones *x*(1), *x*(2),..., *x*(*n*), entonces la estadística suficiente sería el número de muestras *x*(*j*) con valor *V*, denotado por n_V, y el número de muestras con valor *F*, denotado por n_F. La estimación de la probabilidad de obtener *V* sería:

$$\hat{\theta}_{MLE} = \frac{n_V}{n_V + n_F}$$

La densidad de probabilidad *a priori* de los parámetros θ, denotada $p_\theta(\theta)$, representa una densidad de probabilidad conocida o asumida antes de aplicar el método. Es decir, $p_\theta(\theta)$ representa conocimiento previo sobre la distribución θ. Similarmente, $p_{\theta|X}\left(\theta|X_n\right)$ denota la *densidad posterior* de los parámetros θ dadas las mediciones X_n, es decir, la densidad de probabilidad calculada por el estimador.

Es posible aplicar el mismo principio de MLE a estimación bayesiana, lo que produce el estimador *Maximum A Posteriori* (MAP), denotado por $\hat{\theta}_{MAP}$. El estimador MAP se define como el valor del parámetro θ que maximiza la densidad posterior $p_{\theta|X}\left(\theta|X_n\right)$ de θ dadas las mediciones X_n. El estimador MAP puede ser interpretado como el valor más probable de los parámetros θ dados los datos X_n.

Dado un valor para θ, la densidad posterior puede ser calculada utilizando la Regla de Bayes:

$$p_{\theta|X}\left(\theta|X_n\right) = \frac{p_{X|\theta}\left(X_n|\theta\right)p_\theta(\theta)}{p_X\left(X_n\right)}$$

El denominador $p_X\left(X_n\right)$ de la ecuación anterior es la densidad de probabilidad de los datos X_n, la cual no depende de los parámetros θ y solamente sirve para normalizar la densidad posterior, de modo que para encontrar el estimador MAP basta con encontrar el valor de θ que maximiza el numerador. Dicho valor de θ es la densidad conjunta:

$$p_{\theta,X}\left(\theta|X_n\right) = p_{X|\theta}\left(X_n|\theta\right)p_\theta(\theta)$$

Note que si la distribución $p_\theta(\theta)$ es uniforme, entonces MLE y MAP se convierten en el mismo estimador.

4.3.2 Extracción de la estructura

Para la extracción de la estructura de Redes Bayesianas se han desarrollado dos líneas de investigación. Una de ellas consiste en generar diversas estructuras, extraer los parámetros de cada una y evaluar el likelihood de los datos a partir de la red. En este enfoque, la elección de las estructuras puede realizarse utilizando alguna heurística. Entre los principales métodos dentro de esta línea se encuentran Maximum Likelihood Estimation, el algoritmo K2 [Cooper & Herskovits, 1992], y la aplicación de algoritmo genético [Etxeberria et. al., 1997]. Note que, para evaluar la calidad de una determinada estructura, es necesario extraer los parámetros de la Red Bayesiana.

La segunda línea de investigación consta de aquellos métodos que estudian las relaciones de dependencia para construir la estructura que mejor se adapte a la distribución conjunta subyacente a los datos. Entre los principales métodos en esta línea se encuentran los árboles de dependencia [Chow & Liu, 1968], los poly-trees [Rebane & Pearl, 1987], el algoritmo SGS [Sprites et. al., 2000], su sucesor PC, y un algoritmo de tres etapas [Cheng et. al., 1997].

El método más intuitivo y sencillo para extraer la estructura es Maximum Likelihood Estimation, en donde se toma en cuenta el likelihood de la Red Bayesiana respecto a los datos. Dicho de otro modo, sean c_1, c_2, ..., c_m m casos en una base de datos, X_1, X_2, ..., X_n las n variables presentes en la Red Bayesiana, y x_{ik} el valor de la variable X_i en el caso c_k. Para extraer la estructura con el método MLE, se generan varias estructuras permitidas S, a cada S se le extraen los parámetros θ_S, y se calcula el likelihood de la red [Heckerman, 1996]:

$$L(S,\theta_S : D) = \prod_k p(c_k | S,\theta_S) = \prod_k \prod_i p\left(x_{ik} \middle| Pa^S(X_i) \right)$$

En donde $Pa^S(x_{ki})$ denota a los padres de x_{ki} en la estructura S. Por último, se selecciona aquella estructura que presente el mayor likelihood. Debido a que la adición de información genera modelos más precisos, este método tiende a generar estructuras cuyos grafos subyacentes son completos.

Uno de los algoritmos más utilizados para la extracción de la estructura de Redes Bayesianas es el algoritmo K2, en el cual se asume que la densidad de probabilidad sobre las posibles estructuras es uniforme, y que se cuenta con un ordenamiento de las variables. Este algoritmo utiliza una heurística de algoritmo glotón para buscar el conjunto de padres de cada nodo. Sea r_i el número de valores que toma la variable X_i, w_{ij} la j-ésima única instancia de $Padres(X_i)$, q_i el número de instancias w_{ij}, N_{ijk} el número de casos en los que la variable X_i toma el valor x_{ik} y las variables en $Padres(X_i)$ son instanciadas como w_{ij}, y $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$. El algoritmo trata de maximizar la función:

$$g(X_i, Pa(X_i)) = \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}!$$

El algoritmo inicia suponiendo que ningún nodo tiene padres. En cada paso, agrega aquel padre que maximice la función *g*, y se detiene cuando la agregación de cualquier nodo padre haría que *g* disminuyera. Al utilizarse un algoritmo glotón, este método encuentra generalmente mínimos locales.

Una de las alternativas al uso de un algoritmo glotón es la aplicación de algoritmo genético para extraer la estructura de la red. En este sentido, los individuos son las estructuras de las Redes Bayesianas y la función objetivo es la misma *g* definida anteriormente. La Red Bayesiana se representa por una cadena $h_{11}h_{21}...h_{n1}h_{12}h_{22}...h_{n2}...h_{1n}h_{2n}...h_{nn}$, en donde *n* es el número de variables de la estructura, h_{ij}=1 si la *j*-ésima variable es un nodo padre de la *i*-ésima variable y h_{ij}=0 en otro caso.

Comenzando con una población individual, el algoritmo genético selecciona de manera iterativa dos individuos (los padres), y en base a estos se crean otros individuos (los padres tienen hijos). Posteriormente se crea una población combinando la anterior con los individuos recién creados (ha pasado una generación). Este proceso se repite hasta que se alcanza algún criterio de paro. El algoritmo regresa el mejor individuo de acuerdo a la función objetivo. Cuando se crea la población inicial, se debe definir su tamaño y la manera en que se crean los individuos.

Es común que los padres se seleccionen de acuerdo a una probabilidad proporcional al valor de la función objetivo para ellos. Usualmente se utilizan dos operadores en la creación de los hijos: la operación de hibridación y la operación de mutación. Una vez que los padres han sido seleccionados, son sujetos de una operación de hibridación con una probabilidad dada, y entonces los individuos creados a partir de tal operación son mutados con otra probabilidad. Comúnmente se consideran dos operaciones de hibridación: la operación de hibridación uniforme y la fusión de arcos. El operador de mutación consiste en la adición o eliminación aleatoria de un arco. Si las modificaciones crean ciclos en el grafo dirigido, los arcos son eliminados aleatoriamente hasta que el grafo dirigido se vuelve acíclico.

Como uno de los primeros métodos basados en el descubrimiento de relaciones de dependencia entre variables se encuentra el método de árboles de dependencia. Este método asume que solo existen dependencias entre pares de variables, y que el grafo que representa esas dependencias es un árbol. El algoritmo consiste básicamente en los siguientes pasos:

1. Calcular la información mutua entre cada par de variables
2. Encontrar el par o los pares de variables cuya información mutua sea máxima
3. Agregar un arco entre los dos nodos que representen a las variables, ordenando los arcos de manera descendente de acuerdo a la información mutua entre las variables.
4. Regresar al paso 2 hasta que se hayan conectado todos los nodos.

Una de las variaciones a los árboles de dependencia es el método de poliárboles, en los que las variables pueden tener múltiples causas. El algoritmo garantiza que, si el proceso que generó los datos está estructurado como un poliárbol, entonces la estructura del árbol puede ser recuperada de manera precisa, y que la dirección de los arcos puede ser determinada en la medida de lo posible. El algoritmo consiste en los siguientes pasos:

1. Iniciar con un grafo no dirigido, generado a partir de mediciones de dependencia entre variables.
2. Recorrer el grafo hasta encontrar una tripleta de nodos en los que dos arcos sean convergentes sobre uno de los nodos. A este nodo se le llama *nodo multipadre*.
3. A partir del nodo multipadre determinar las direcciones de los arcos, utilizando el hecho de que los padres son independientes uno del otro.
4. Ir al paso 2 hasta que no se puedan orientar más arcos.
5. Utilizar una semántica externa para orientar los arcos restantes.

A pesar de que las estructuras de árbol presentan ventajas tales como algoritmos de cálculo de probabilidades con baja complejidad, la restricción respecto a que la distribución de probabilidad haya sido generada por una estructura de este tipo es demasiado fuerte. Esto ha motivado la creación de algoritmos con restricciones más relajadas, entre los que se encuentran SGS. El algoritmo SGS tiene como objetivo crear un grafo acíclico dirigido G tal que la distribución conjunta de probabilidad P sea fiel a G. Este algoritmo consiste básicamente en los siguientes pasos:

1. Formar el grafo no dirigido completo H sobre el conjunto de vértices V
2. Para cada par de vértices A y B
   Si existe un subconjunto S de $V\backslash\{A, B\}$ tal que A y B están d-separados dado S
   Eliminar el arco entre A y B de H.
   End For
3. Sea K el grafo no dirigido resultante del paso 2.
   Para cada tres vértices A, B y C, tal que los nodos A, B y B, C son adyacentes en K, pero los nodos A y C no lo son
   Si no existe un subconjunto S de $\{B\}\cup V\backslash\{A, C\}$ que d-separa A y C.
   Orientar A-B-C como $A\rightarrow B\leftarrow C$
   End For
4. Repetir
   Si $A\rightarrow B$, B y C son adyacentes, A y C no son adyacentes, y no hay un arco incidente en B
   Orientar B-C como $B\rightarrow C$.
   Si hay un camino dirigido de A a B, y un arco entre A y B

```
            Orientar A–B como A→B.
    Hasta que no se puedan orientar más arcos.
```

En el peor caso, el algoritmo SGS requiere un número de pruebas de d-separación que crece exponencialmente con el número de vértices, como sucede con cualquier algoritmo basado en relaciones de independencia. El problema con el algoritmo SGS es que el peor caso es también el caso esperado.

Un algoritmo más eficiente, conocido como PC, comienza formando el grafo no dirigido completo y lo adelgaza eliminando arcos con relaciones de independencia condicional del orden de cero, adelgaza de nuevo con las relaciones de independencia de orden uno, y así sucesivamente. Sea *Adyacencias*(*C*, *A*) el conjunto de vértices adyacentes a *A* en el grafo acíclico dirigido *C*. El algoritmo PC es el siguiente:

```
1. Formar el grafo completo no dirigido C sobre el
   conjunto de vértices V.
2. n = 0
   Repetir
        Repetir
             Seleccionar un par ordenado de variables X y Y
             que sean adyacentes en C, tal que
             Adyacencias(C, X)\{Y} tenga cardinalidad mayor
             o igual a n, y un subconjunto
             S⊆Adyacencias(C, X)\{Y} de cardinalidad n
             Si X y Y están d-separados dado S
                  Eliminar el arco X–Y de C y almacenar S
                  en Sepset(X, Y) y Sepset(Y, X)
        Hasta que, para todos los pares ordenados de
        variables adyacentes X y Y, Adyacencias(C, X)\{Y}
        tenga cardinalidad mayor o igual a n, y se haya
        realizado la prueba de d-separación para todos los
        subconjuntos S⊆Adyacencias(C, X)\{Y} de
        cardinalidad n.
        n = n+1
   Hasta que, para cada par ordenado de vértices adyacentes
   X, Y, Adyacencias(C, X)\{Y} sea de cardinalidad menor
   que n.
3. Para cada tres vértices X, Y y Z tales que los pares X,
   Y y Y, Z sean adyacentes en C, pero el par X, Z no sea
   adyacente en C, orientar X–Y–Z como X→Y←Z si y solo si
   Y∉Sepset(X, Z)
4. Repetir
        Si A→B, B y C son adyacentes, A y C no son
        adyacentes, y no hay un arco incidente en B
             Orientar B–C como B→C.
```

```
        Si hay un camino dirigido de A a B, y un arco entre
        A y B
             Orientar A–B como A→B.
    Hasta que no se puedan orientar más arcos.
```

Entre otros algoritmos desarrollados se encuentra uno que extrae la estructura de la red mediante un proceso de tres etapas: bosquejo, engrosamiento y adelgazamiento. En la primera fase, este algoritmo calcula la información mutua de cada par de nodos como una medida de cercanía, y crea un bosquejo basado en esta información. El bosquejo es un grafo poco conectado. En el caso especial de que la Red Bayesiana sea un árbol o un bosque, esta fase puede construir la red de manera correcta, haciendo innecesarias la segunda y tercera fases. En la segunda fase, el algoritmo añade arcos cuando los pares de nodos no pueden ser d-separados. El resultado tiene la estructura de un I-Map del modelo de dependencia subyacente dado que el modelo subyacente es un grafo acíclico dirigido con fidelidad normal. En la tercera fase, cada arco del I-Map es examinado usando pruebas de independencia condicional, y es eliminado cuando los dos nodos pueden ser d-separados. El resultado de esta fase tiene la estructura de un mapa perfecto cuando el modelo subyacente presenta fidelidad normal. Por último, se realiza un procedimiento para orientar los arcos del grafo.

5. DISCRETIZACIÓN BASADA EN VECTORES

Se puede entender un método de discretización como una función que va de un conjunto $A \subseteq \Re$ a un conjunto finito de símbolos S. En la práctica, dada una serie de tiempo

$$\{x_t : t=1...n\}$$

se puede definir cada uno de los valores a ser discretizados $a_i \in A$ de acuerdo al valor de cada punto de la serie de tiempo, por ejemplo $a_i=x_i$, o como la variación de cada punto de la serie de tiempo respecto al anterior, por ejemplo $a_i=x_i-x_{i-1}$. En el primer caso diremos que el método de discretización toma en cuenta la *magnitud* de la serie de tiempo, mientras que en el segundo caso diremos que toma en cuenta su *pendiente*.

Cuando se discretiza tomando en cuenta la magnitud, cualquier análisis que se realice con los datos estará limitado al intervalo considerado para dicha discretización. Por ejemplo, suponga que se intenta predecir una serie de tiempo que crece monotónicamente, como se muestra en la figura 5.1. La predicción permanecerá constante, tomando el mayor valor al que haya sido asignado un símbolo.

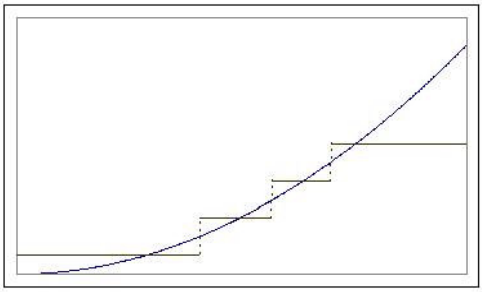


Figura 5.1. Falla de la discretización basada en la magnitud

Cuando se discretiza tomando en cuenta la pendiente, este problema desaparece debido a que el valor de algún punto en la serie de tiempo es relativo al valor del punto anterior. Sin embargo, cuando se discretiza de esta forma es imposible encontrar ciertas relaciones no lineales, como la mostrada en la figura 5.2.

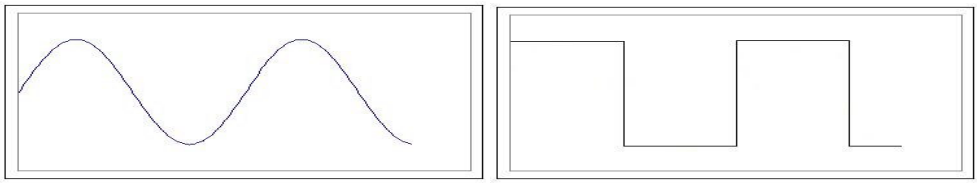


Figura 5.2. Series de tiempo cuya relación no puede ser encontrada si se discretizan tomando en cuenta la pendiente

El método de discretización presentado en este capítulo es capaz de tomar en cuenta tanto la magnitud como la pendiente de la serie de tiempo. Esta discretización se presenta junto a un algoritmo de agrupamiento determinista, relativamente rápido, y que permite especificar el número de agrupaciones a conformar.

5.1 Descripción

En general, un método de discretización es aquel que asocia a un conjunto de valores continuos con un símbolo determinado. Hasta ahora, los métodos de discretización han definido dicho conjunto de valores continuos como intervalos de valores que pueden estar referidos, ya sea al valor de la serie de tiempo o a la variación de la misma respecto al punto anterior.

Si se desea tomar en cuenta tanto el valor de la serie de tiempo como su variación respecto al punto anterior, entonces cada punto debería estar expresado como una dupla de la forma

(*valor*, *variación*)

Se observa que referirse a la variación de la serie de tiempo es equivalente a referirse a su pendiente *m*. A su vez, la pendiente se relaciona con el ángulo θ que tendría una recta con pendiente *m* respecto a la horizontal. De este modo, si llamamos *r* al valor de la serie de tiempo, cada punto podría ser expresado por una dupla de la forma

(r, θ)

Claramente, esta es la forma general de un vector en $\Re^2$ expresado en coordenadas polares. Siguiendo este razonamiento, los vectores de este tipo pueden ser convertidos a coordenadas cartesianas, de modo que sea posible calcular la Distancia Euclidiana entre pares de ellos. Posteriormente, bastaría con utilizar un algoritmo de agrupamiento para crear clases de vectores cercanos que puedan ser representados por un mismo símbolo, como se muestra en la figura 5.3.

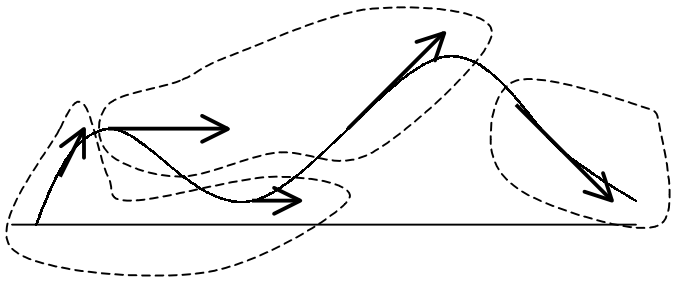


Figura 5.3 Conversión de los puntos de la serie de tiempo a vectores y su agrupamiento

Parecería que el proceso descrito es suficiente para obtener la discretización deseada. Sin embargo, debido al aumento en la magnitud de los vectores al crecer el valor de la serie de tiempo, se observa que la comparación entre vectores con mayor magnitud (aquellos que se encuentran en valores altos de la serie) será más sensible a cambios en la pendiente. Es decir, vectores con magnitud grande tenderán a agruparse en clases separadas debido a que la distancia entre ellos será grande, aun cuando los cambios en su pendiente no tendrían el mismo efecto en vectores de menor magnitud, como se muestra en la figura 5.4.

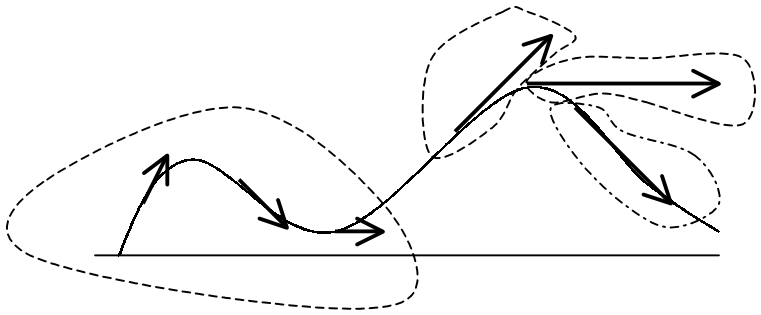


Figura 5.4 Vectores con mayor magnitud se agrupan en clases separadas

A primera vista, se podría pensar que la solución a este problema es normalizar los vectores que se comparan. Sin embargo, esto haría imposible reconocer diferencias en la magnitud, obteniéndose un método de discretización por variaciones. De aquí se infiere que la solución debe conservar las diferencias en la magnitud de los vectores.

Sean dos vectores $v_1=(r_1, \theta_1)$ y $v_2=(r_2, \theta_2)$, la diferencia entre sus magnitudes debe ser siempre r_2-r_1. Si se establece la magnitud de v_1 como un valor arbitrario σ, para cumplir la condición anterior la magnitud de v_2 debe ser igual a $\sigma+(r_2-r_1)$, como se muestra en la figura 5.5.

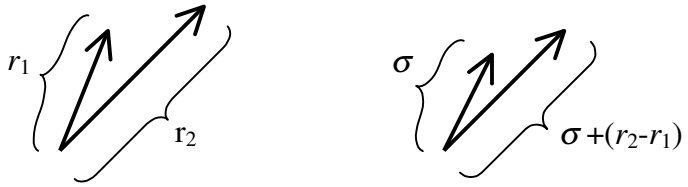


Figura 5.5 Conversión de las magnitudes de los vectores

Al realizar esta conversión, el ángulo entre los vectores no se modifica, por lo que la diferencia entre ellos depende únicamente de la diferencia entre sus magnitudes y sus ángulos originales. De ésta forma se logra que vectores con diversas magnitudes sean comparados de manera homogénea.

Aunque se ha utilizado el ángulo de los vectores para fines de claridad en la explicación, en realidad su uso requeriría la aplicación de funciones trigonométricas costosas en tiempo y precisión. En la práctica se pueden obtener las coordenadas cartesianas del vector a partir de la pendiente, que se aproxima como la diferencia del punto especificado respecto al anterior.

Una vez que los valores de la serie de tiempo han sido convertidos a vectores, es necesario agrupar dichos vectores de acuerdo a la distancia entre ellos. Para ello, se puede utilizar cualquier algoritmo de agrupamiento que cumpla las siguientes propiedades:

1. ***Permita especificar el número de agrupaciones.*** Generalmente, durante la discretización se especifica el número de símbolos deseados. Como cada agrupación se convierte en un símbolo, es indispensable poder especificar el número de agrupaciones.
2. ***Sea determinista.*** Un algoritmo aleatorio generalmente arrojará resultados diferentes en cada corrida, lo cual impide especificar parámetros (tales como σ) que produzcan el menor error de discretización.
3. ***Sea eficiente.*** Los algoritmos de discretización que requieren construir la matriz de distancias utilizan demasiados recursos y tiempo de procesamiento cuando el número de elementos crece. Por lo tanto, sería preferible un algoritmo incremental.

El algoritmo de agrupamiento que se presenta cumple con estas tres características. Se trata de un algoritmo incremental basado en representantes, que permite especificar el número de agrupaciones.

El algoritmo inicia formando *K* agrupaciones, cada una con un representante inicial diferente. Estos representantes podrían ser los *K* primeros vectores que se introduzcan para ser agrupados. Cada agrupación tiene asociado un número real positivo λ conocido como *tolerancia*, que inicialmente es cero.

Cada vez que se añade un nuevo vector *v*, se busca el representante cuya distancia sea menor. Si la distancia entre *v* y el representante elegido es menor o igual que la tolerancia para esa agrupación, el vector *v* es agregado a la agrupación, terminando el proceso para este vector, como se muestra en la figura 5.6.

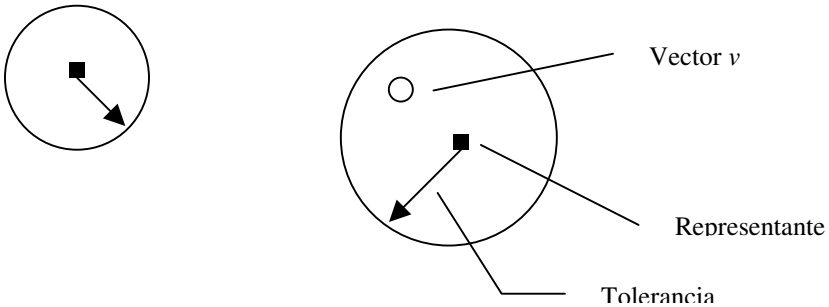


Figura 5.6 El nuevo vector *v* es agregado a la agrupación

En caso de que la distancia entre *v* y el representante sea mayor que la tolerancia, se elige un nuevo representante tal que la distancia entre éste y *v* sea menor que la tolerancia, pero que además sea tan cercano como sea posible al vector más alejado de *v* dentro de la agrupación, como se muestra en la figura 5.7.

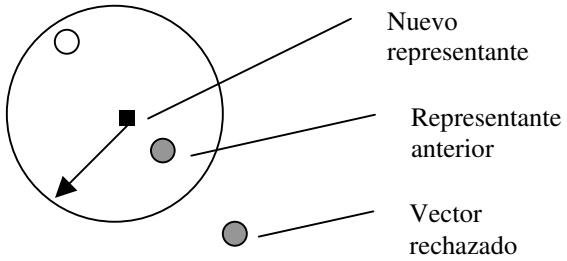


Figura 5.7 Elección de un nuevo representante para incluir a *v*

Al elegirse un nuevo representante, es probable que la distancia de éste con algunos vectores sea mayor a lo permitido por la tolerancia. En este caso, se dice que cada uno de los vectores que quedaron fuera de la tolerancia fueron *rechazados*, y deben buscar al representante más cercano. Si dicho representante pertenece a una agrupación distinta a la que pertenecía el vector rechazado, éste se une a la agrupación siguiendo el procedimiento descrito hasta ahora. En caso de que el representante elegido sea el de la agrupación a la cual pertenecía el vector rechazado, se elige un nuevo representante que se encuentre entre el vector rechazado y el vector más alejado de éste, y se aumenta la tolerancia de la agrupación para abarcar a todos los vectores, como se muestra en la figura 5.8. Esto se repite para agregar el vector *v* a la agrupación.

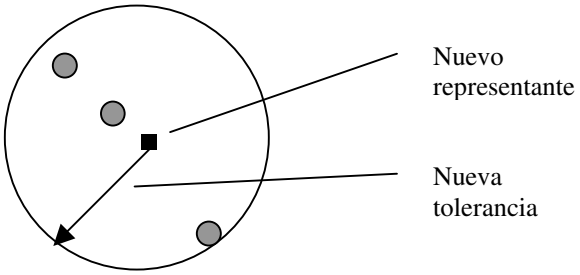


Figura 5.8 Elección de un nuevo representante y aumento de la tolerancia

En las siguientes dos secciones se expresan más formalmente las ideas que se acaban de presentar.

5.2 Conversión de los valores continuos

Sea $\{x_t : t=1...n\}$ una serie de tiempo. Cada punto x_i es expresado como una dupla de la forma

$$v_i = (x_i, m_i) \qquad i=2..n$$

en donde

$$m_i = x_i - x_{i-1}$$

Sea $V=\{v_2, v_3, ... v_n\}$ el conjunto de todas las duplas extraídas a partir de la serie de tiempo. Definimos la función d: $V^2 \rightarrow \Re$ como:

$$d(v_i, v_j) = \sqrt{(a_i - a_j)^2 + (b_i - b_j)^2} \qquad \textbf{(5.1)}$$

en donde

$$a_i = \frac{\sigma}{\sqrt{1 + m_i^2}} \qquad b_i = m_i a_i$$

$$a_j = \frac{x_j - (x_i - \sigma)}{\sqrt{1 + m_j^2}} \qquad b_j = m_j a_j$$

σ es un parámetro llamado *peso de la pendiente*. Este parámetro define la proporción en la que se tomará en cuenta la pendiente en la serie de tiempo durante la evaluación de la función d. Si $\sigma = 0$, la pendiente no se toma en cuenta, y la función d depende únicamente de x_i y x_j. Si $\sigma \rightarrow \infty$ el valor de la función d depende únicamente de m_i y m_j.

5.3 Agrupamiento

Una vez que los puntos han sido convertidos, se agrupan de acuerdo a la función d. El algoritmo de agrupamiento propuesto recibe como parámetro de entrada el número de agrupaciones que serán creadas.

Sea K el número de agrupaciones a ser creadas, y sean C_1, C_2, ..., C_K, $K<n$, conjuntos de duplas $v_i \in V$. C_i es la i-ésima agrupación.

Sean las duplas r_1, r_2, ..., $r_K \in V$, $r_i \neq r_j \ \forall i \neq j$. Cada una de estas duplas está contenida en una y solo una agrupación, es decir, $r_i \in C_i \ \forall i$. La dupla r_i es llamada el *representante* de la agrupación C_i.

Sean $\lambda_1, \lambda_2, ..., \lambda_K \in \Re$, $\lambda_i \geq 0 \ \forall i$. λ_i es llamado *tolerancia* de la agrupación C_i. Para todo λ_i siempre se debe cumplir que $d(v_p, v_q) \leq \lambda_i \ \forall v_p, v_q \in C_i$.

El algoritmo de agrupamiento inicia asignando un representante inicial a cada agrupación. Todos los representantes deben ser diferentes, y bien podrían ser las K primeras duplas que se introduzcan para ser agrupadas. Posteriormente, el algoritmo recibe cada nueva dupla y encuentra aquella agrupación cuyo representante minimice la función de distancia d:

```
Sea Ci = {ri} y ti=0 ∀i
Para cada nuevo punto vn
   Encontrar el conjunto Cs tal que d(vn, rs) ≤ d(vn, ri) ∀i
   AgregarAAgrupación (vn, Cs)
End For
```

Si la nueva dupla cae dentro de la tolerancia de la agrupación elegida, se agrega a dicha agrupación. En caso contrario, debe elegirse un representante para la agrupación de modo que la nueva dupla caiga dentro de la tolerancia. Aquellas duplas cuya distancia al representante sea mayor que la tolerancia deberán buscar la agrupación cuyo representante sea el más cercano. En caso de que una dupla regrese a su agrupación original, ésta es agregada provocando la elección de un nuevo representante y aumentando la tolerancia.

```
AgregarAAgrupación (vn, Cs):
Fs = ∅
Si d(vn, rs) > λs
   Encontrar vo∈Cs ∋ d(vn, vo)≥d(vn, vi)  ∀vi∈Cs
   Seleccionar un nuevo rs∈Cs tal que d(vn, rs)≤λs ∧
     [d(vn, rs)]2+[d(rs, vo)]2 ≤ [d(vn, vi)]2+[d(vi, vo)]2 ∀vi∈Cs
   Sea el conjunto Fs={vi∈Cs | d(rs, vi)>λs}
   Cs = Cs \ Fs
Para cada vf∈Fs
   Seleccionar el conjunto Ce tal que d(re, vf) ≤ d(rj, vf) ∀j
   Si Ce=Cs
      IncluirEnAgrupación (vf, Cs)
   Else
      AgregarAAgrupación (vf, Ce)
End For
IncluirEnAgrupación (vn, Cs)
```

La elección del nuevo representante y el aumento a la tolerancia se realizan buscando la dupla más lejana a la que se desea incluir, y seleccionando aquella dupla que se encuentre en medio de las dos.

```
IncluirEnAgrupación (vn, Cs):
Si d(vn, rs) > λs
   Encontrar vo'∈Cs ∋ d(vn, vo') ≥ d(vn, vi) ∀vi∈Cs
   Seleccionar un nuevo rs tal que
      [d(vn, rs)]2+[d(rs, vo')]2 ≤ [d(vn, vi)]2+[d(vi, vo')]2
                                                    ∀vi∈Cs
   λs = max[d(vn, rs), d(vo', rs)]
Cs = Cs∪{vn}
```

La idea principal detrás de este algoritmo es intentar balancear la tolerancia entre las agrupaciones. Es decir, el algoritmo intenta impedir que la tolerancia de una agrupación

crezca antes de que se compruebe con las demás agrupaciones que este crecimiento es indispensable.

Después de esta etapa de discretización, cada punto puede ser recuperado utilizando el representante de la agrupación a la cual pertenece.

5.4 Recuperación del los valores continuos

Para recuperar la serie de tiempo a partir de la discretización, se toma como base el representante de cada agrupación. El valor continuo resultante se calcula mediante un promedio pesado entre la amplitud y la pendiente de los representantes. El peso de la pendiente en dicho promedio está dado por el parámetro σ:

$$\bar{x}_i = \frac{x_i^r e^{-\sigma/I} + (x_{i-1} + m_i^r) e^{-I/\sigma}}{e^{-\sigma/I} + e^{-I/\sigma}} \qquad \textbf{(5.2)}$$

en donde $\bar{x}_i$ es el valor continuo recuperado en la posición i, x^r_i es el componente de amplitud y m^r_i es el componente de pendiente del representante de la agrupación a la cual pertenece la dupla v_i, I es el intervalo dinámico de la serie de tiempo, y x_{i-1} es el valor continuo anterior de la serie.

El intervalo dinámico de la serie de tiempo es la diferencia entre sus valores máximo y mínimo. Como se puede observar, el valor del parámetro σ está referido a dicho intervalo.

El valor continuo anterior x_{i-1} puede ser el valor previo recuperado o un valor especificado. Si se desea recuperar una serie de tiempo a partir de una discretización, es necesario especificar el primer valor de la serie, de modo que se pueda utilizar como parámetro x_{i-1}.

5.5 Pruebas y resultados

El método de Discretización Basada en Vectores entrega símbolos discretos cuya relación con la magnitud y la pendiente de la serie de tiempo está dada por el valor del parámetro σ. Si el valor de este parámetro es similar al intervalo dinámico que toma la serie de tiempo, se espera una relación equilibrada.

Como ejemplo tome la serie de tiempo original mostrada en la figura 5.9. Esta se obtuvo a partir de la ecuación $t^3+8t^2-44t+15$, en donde t toma valores entre –15 y 10. En este caso, el intervalo de valores es de 2275, que resulta de restar al máximo valor (1375) el mínimo (-900).

Durante la discretización se utilizaron 4 símbolos y un valor de σ =3254.5, elegido de modo que se minimizara el ruido al recuperar la serie de tiempo. Los símbolos obtenidos se

muestran como cuatro valores diferentes: 0, 500, 1000 y 1500. Estos valores fueron tomados para fines de visualización y son completamente irrelevantes.

La discretización de la figura 5.9 muestra claramente el comportamiento del método de discretización aquí descrito. Note, por ejemplo, que si se hubiera utilizado un método que solamente tomara en cuenta la amplitud, entonces los puntos t=-11 y t=-2 estarían representados por el mismo símbolo. Asimismo, si se hubiera tomado solamente la pendiente, entonces el punto t=-14 estaría representado por el mismo símbolo que el punto t=9 o t=10.

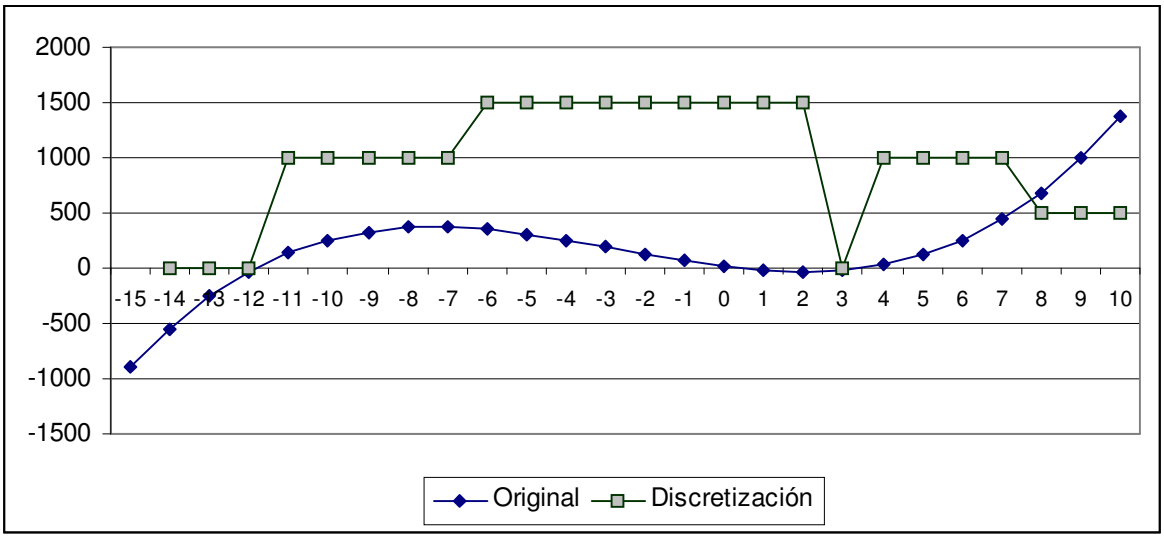


Figura 5.9. Discretización de una serie de tiempo

Observe que, en términos generales, los cuatro símbolos obtenidos se pueden describir como:

1. Valores bajos con pendiente creciente (símbolo 0)
2. Valores medios con pendiente creciente, cercana a cero (símbolo 1000)
3. Valores medios con pendiente decreciente (símbolo 1500)
4. Valores altos con pendiente creciente (símbolo 500)

5.5.1 Comparación con otros métodos

Al revisar el método de Discretización Basada en Vectores, resulta evidente que su eficacia para recuperar la serie de tiempo depende del número de símbolos o agrupaciones a formar y del parámetro σ. En la tabla 5.1 se muestra la Relación Señal a Ruido (SNR) obtenida al discretizar y recuperar la serie de tiempo *Seno* mostrada en la figura 5.10, utilizando 32 símbolos y variando el valor del parámetro σ desde 0 hasta 200 a intervalos de 20.

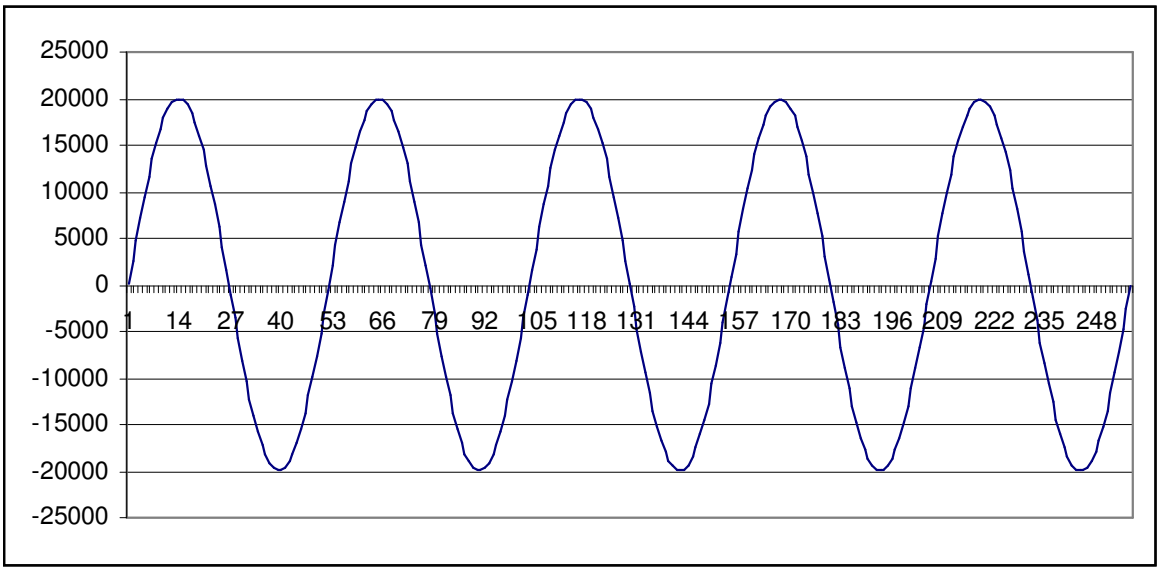


Figura 5.10. Serie de tiempo *Seno*

σ	SNR (dB)
0.0	56.88846761
20.0	56.45378732
40.0	56.48914757
60.0	56.49270806
80.0	56.5162774
100.0	57.06144038
120.0	58.19614256
140.0	58.24030508
160.0	56.11908117
180.0	55.8742502
200.0	57.50134759

Tabla 5.1. Relación Señal a Ruido obtenida con diferentes valores para el parámetro σ

Realizando una búsqueda local alrededor de σ=140 se encuentra que cuando σ=141.05 se obtiene una Relación Señal a Ruido (SNR) de 58.5993dB.

La tabla 5.2 muestra la Relación Señal a Ruido obtenida al discretizar y recuperar la misma serie de tiempo utilizando PCM (ver sección 4.1). Con 32 símbolos, la SNR alcanzada al utilizar PCM es visiblemente menor que la obtenida por el método de Discretización Basada en Vectores.

Número de símbolos	SNR (dB)
32	27.9043
64	33.7991
128	39.1547
256	45.0631
512	51.3041

Tabla 5.2. SNR obtenido al discretizar la serie *Seno* usando PCM

La tabla 5.3 muestra la Relación Señal a Ruido obtenida cuando la serie de tiempo *Seno* se discretiza y recupera utilizando DPCM. Note que este tipo de discretización requiere especificar el tamaño del intervalo de cuantificación *q*, que indica el tamaño mínimo de la variación que tendrá la serie de tiempo recuperada. Como se observa, este parámetro debe disminuir conforme aumenta el número de símbolos a fin de producir variaciones más finas en la serie.

Se observa que la mejor Relación Señal a Ruido obtenida cuando se utilizan 32 símbolos es 39.5637dB, menor a los 58. 5993dB alcanzados por el método de Discretización Basada en Vectores.

		q				
		64	128	256	512	1024
Número de símbolos	32	2.6643	21.6356	39.5637	33.5208	26.8216
	64	21.4882	45.5415	39.5637	33.5208	26.8216
	128	51.9384	45.5415	39.5637	33.5208	26.8216
	256	51.9384	45.5415	39.5637	33.5208	26.8216
	512	51.9384	45.5415	39.5637	33.5208	26.8216

Tabla 5.3. SNR obtenido al discretizar la serie Seno usando DPCM

La figura 5.11 muestra los valores discretos obtenidos utilizando el método de Discretización Basada en Vectores con σ=141.05. Note que cada símbolo representa a una clase, por lo que no importa el valor del símbolo, sino los puntos que están contenidos en él. Por ejemplo, todos los puntos que se encuentran en el valor 30 pertenecen a una clase identificada por este número, aunque dicha clase podría representar cualquier valor de amplitud y pendiente en la serie de tiempo.

A partir de la discretización anterior se recupera la serie de tiempo que se muestra en la figura 5.12. La serie recuperada únicamente presenta pequeños errores en los máximos y mínimos de la serie original.

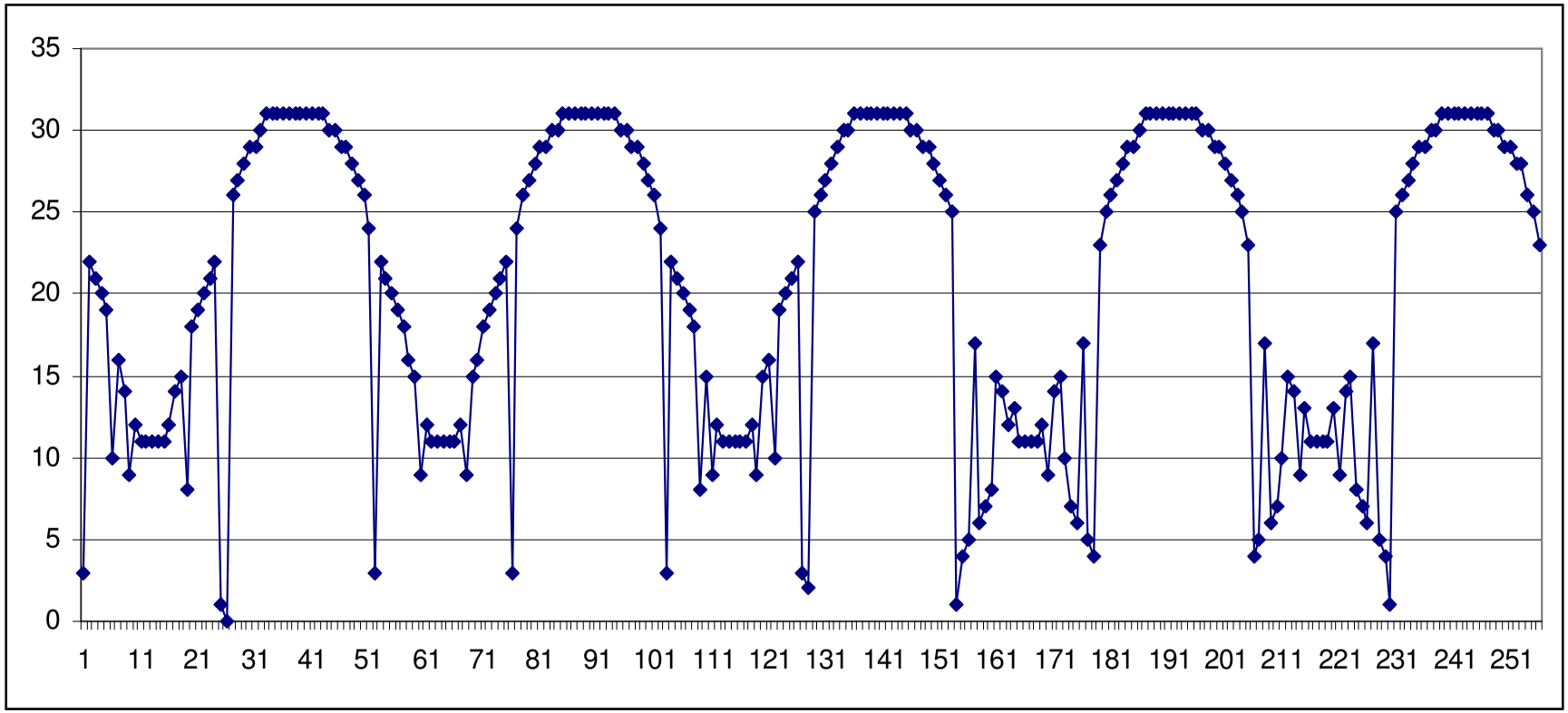


Figura 5.11. Valores discretos obtenidos para la serie *Seno* utilizando σ=141.05

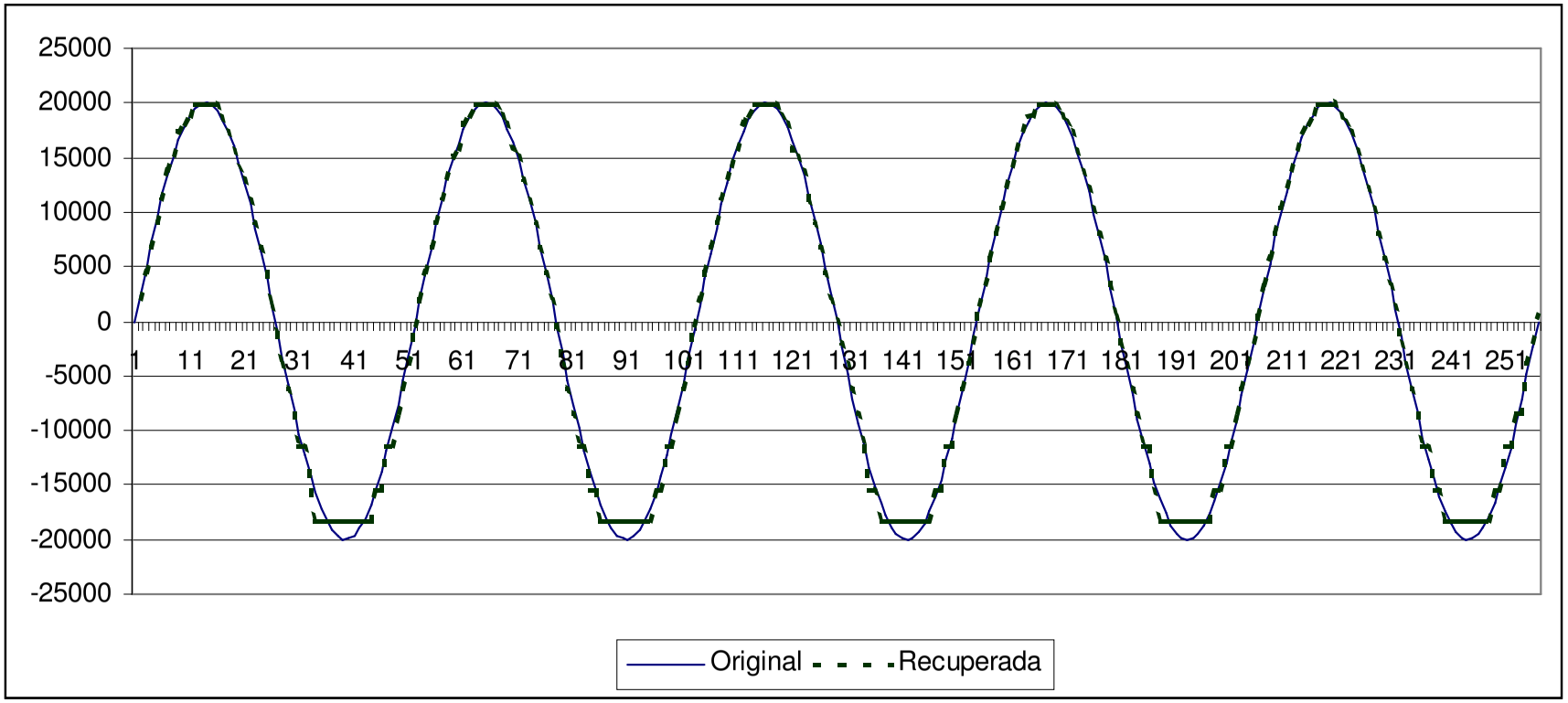


Figura 5.12. Recuperación de la serie *Seno*

La figura 5.13 muestra 5000 datos de una señal de voz típica en Comunicaciones Digitales. La tabla 5.4 muestra la Relación Señal a Ruido obtenida al discretizar y recuperar esta señal utilizando el método de Discretización Basada en Vectores con 32 símbolos y variando el valor del parámetro σ desde 0 hasta 900 en saltos de 100.

σ	SNR (dB)
0.0	54.8520785
100.0	36.5591896
200.0	33.2356502
300.0	38.9154879
400.0	31.3488222
500.0	29.4111089
600.0	31.1197248
700.0	20.654564
800.0	19.3402521
900.0	21.8302975

Tabla 5.4. SNR obtenido para diversos valores de σ en la señal de voz

Realizando una búsqueda local alrededor de σ=300 se encuentra que es este valor el que produce una mejor Relación Señal a Ruido. Asimismo, realizando una búsqueda local alrededor de σ=10 se encuentra que con σ=5.5 se logra una Relación Señal a Ruido de 55.4267dB.

La tabla 5.5 muestra los resultados obtenidos al discretizar y recuperar la misma señal utilizando PCM. Note que, cuando se utilizan 32 símbolos, PCM arroja una Relación Señal a Ruido de 21.2495dB, menor a los 55.4267dB obtenidos por el método de Discretización Basada en Vectores.

Número de símbolos	SNR (dB)
32	21.2495
64	27.6455
128	33.6430
256	39.7625
512	45.5476

Tabla 5.5. SNR obtenido al discretizar la señal de voz usando PCM

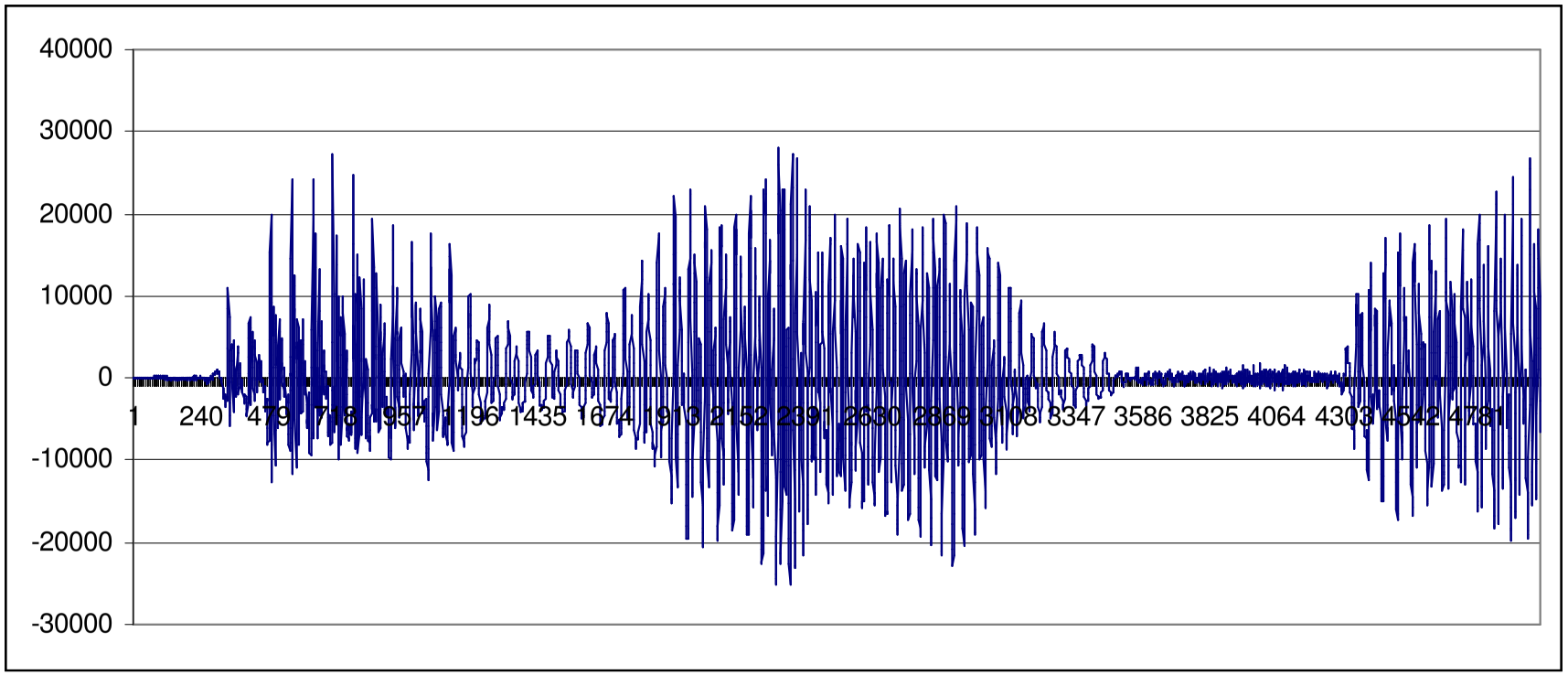


Figura 5.13. Señal de voz

La tabla 5.6 muestra los resultados al discretizar utilizando DPCM. La mejor Relación Señal a Ruido obtenida cuando se utilizan 32 símbolos es de 21.2695dB, muy similar al caso de PCM, y por tanto inferior a la obtenida con el método de Discretización Basada en Vectores.

		q				
		64	128	256	512	1024
Número de símbolos	32	0.7763	2.5757	8.1096	18.9619	21.2695
	64	2.5757	8.0818	19.2956	27.5858	21.2952
	128	8.0666	19.3178	33.2606	27.6429	21.2952
	256	19.3225	38.5854	33.5341	27.6429	21.2952
	512	42.1389	39.7171	33.5341	27.6429	21.2952

Tabla 5.6. SNR obtenido al discretizar la señal de voz usando DPCM

Algunos valores discretos obtenidos utilizando el método de Discretización Basada en Vectores, con σ=5.5, se muestran en la figura 5.14. Parte de la serie recuperada a partir de esta discretización se muestra en la figura 5.15. Aunque se ha discretizado y recuperado la señal entera, por razones de visualización se muestran únicamente 1000 valores (aquellos que se encuentran entre las posiciones 2000 y 3000 de la señal).

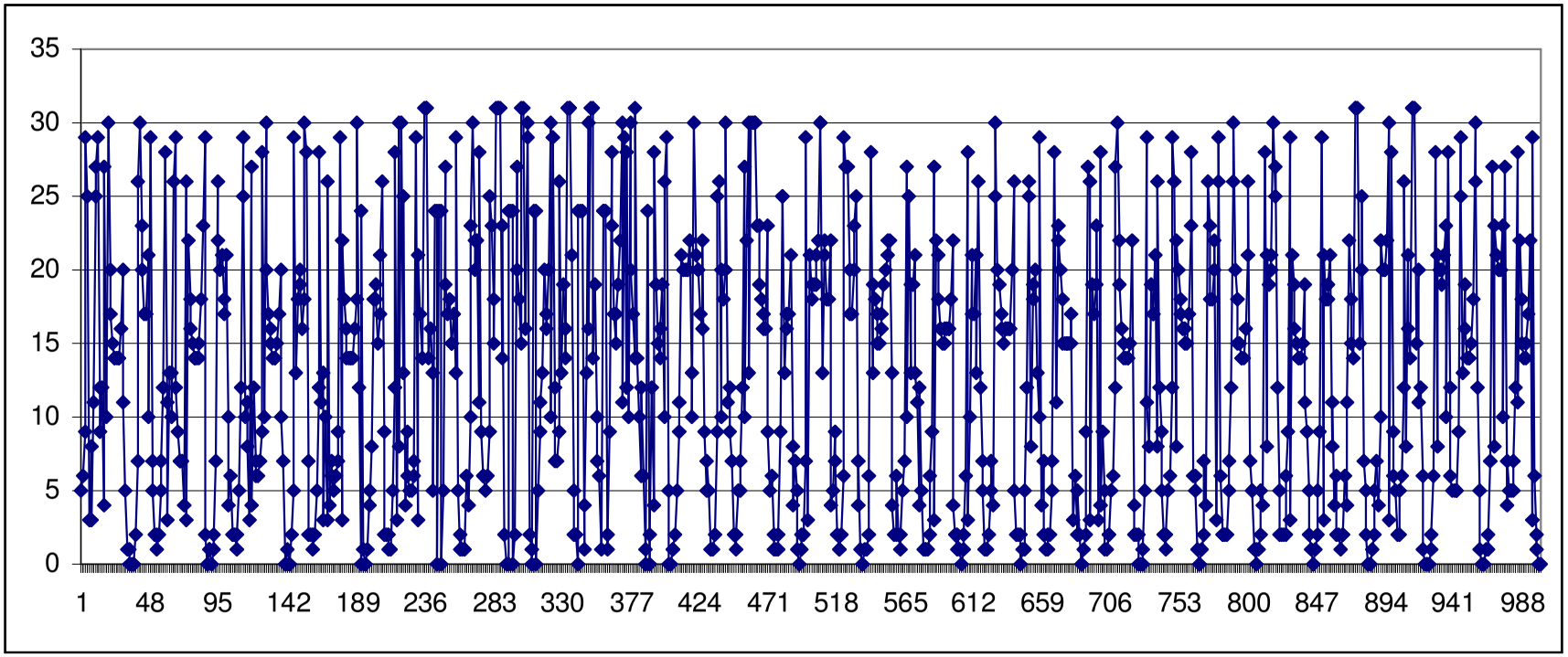


Figura 5.14. Algunos valores discretos obtenidos para la señal de voz utilizando σ=5.5

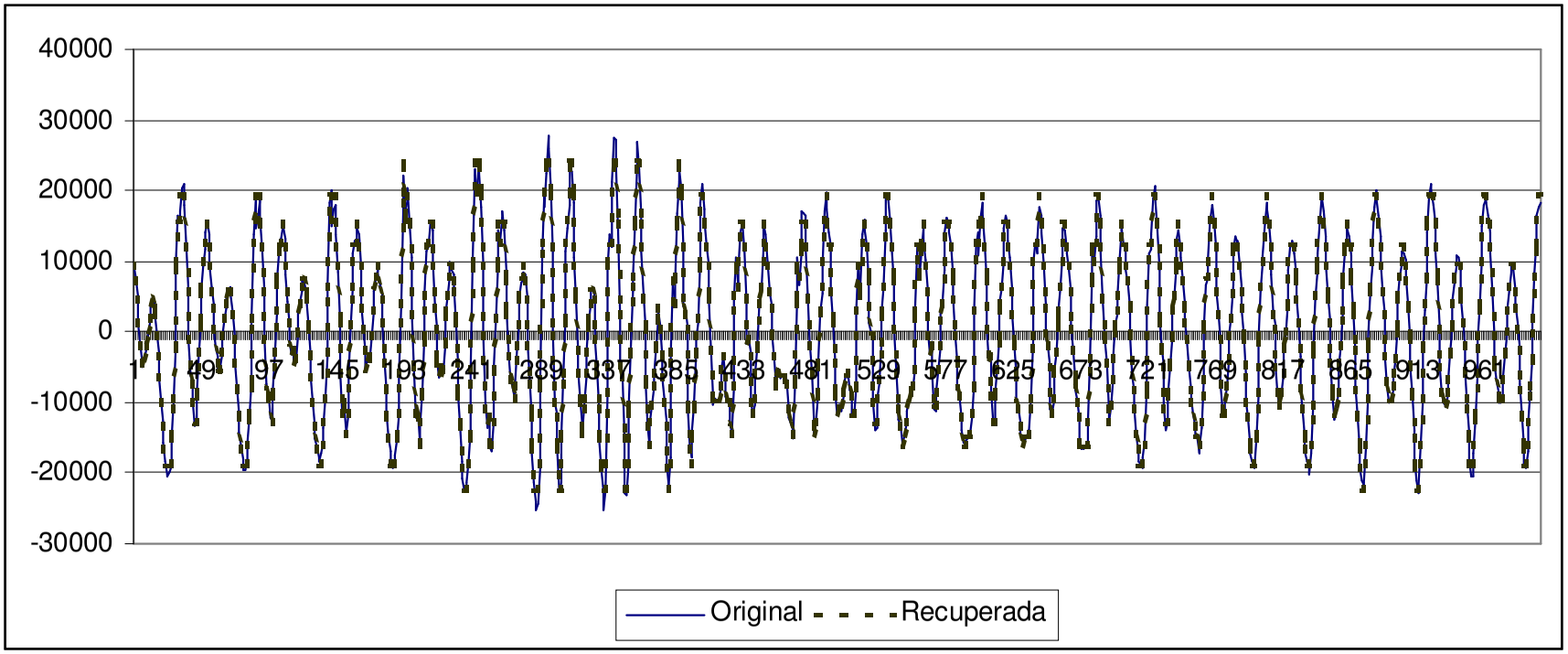


Figura 5.15. Recuperación de la señal de voz

6. DISCRETIZACIÓN Y ALINEACIÓN

La extracción automática de Redes Bayesianas se realiza comúnmente obteniendo valores de probabilidad a partir de un conjunto de registros en una base de datos, y combinándolos con conocimiento previo que proporcione una estimación (en ocasiones subjetiva) de los mismos. Es poco frecuente que se tomen en cuenta las relaciones de dependencia existentes entre los registros de la base de datos, es decir, generalmente el conjunto de registros es manejado sin tomar en cuenta algún orden en la base de datos. Estas relaciones de dependencia solo se consideran cuando se enfrenta el problema de datos incompletos [Ramoni & Sebastiani, 1997]. Así, la extracción de estructura a partir de una base de datos considera relaciones de la forma mostrada en la figura 6.1*a*. Sin embargo, el orden de los datos contiene una cantidad importante de información, y se le debe considerar al extraer la Red Bayesiana, como se muestra en la figura 6.1*b*.

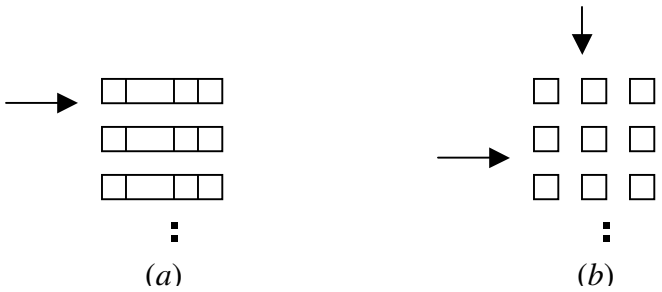


Figura 6.1 (*a*)La extracción a partir de una base de datos solo considera relaciones entre las variables en los casos. (*b*)La extracción a partir de series de tiempo debe considerar relaciones entre diferentes series y el orden en el que los valores se presentan en cada serie.

Si la resolución en el muestreo de las series de tiempo es suficientemente fina, después de la ocurrencia de un evento pasará algún tiempo antes de que su consecuencia sea visible [Dahlhaus & Eichler, 2000]. Esto revela algunas relaciones que añaden información a la Red Bayesiana y sirve para disminuir el espacio de búsqueda cuando se extrae la estructura.

6.1 Alineación de secuencias discretas

Como ya se ha explicado, cuando las series de tiempo han sido muestreadas con suficiente resolución, pasará algún tiempo antes de que un evento ocurrido en una serie de tiempo se vea reflejado en otra. Por lo tanto, antes de utilizar alguna técnica para extraer la estructura de la Red Bayesiana a partir de un conjunto de casos, es necesario descubrir este retraso y alinear las series.

El método utilizado para alinear las series consiste en definir una ventana de datos que se desplaza a través de cada serie de tiempo. El objetivo es encontrar la posición de cada ventana tal que se maximice la información mutua entre todos los pares de series de tiempo, como se muestra en la figura 6.2.

Figura 6.2 Búsqueda de la mejor alineación de las series de tiempo.

Si se cuenta con un conjunto de *m* series de tiempo, cada una de tamaño *n*, y si el máximo desplazamiento es *d* (esto es, el tamaño de la ventana es *n-d*), el espacio de búsqueda es (*n-d*)*m*. Como se observa, cuando crece el número y tamaño de las series de tiempo, el espacio de búsqueda puede ser grande, lo que hace conveniente el uso de algoritmos aleatorios.

6.2 Etapas de discretización y alineación

La elección de un valor correcto para el parámetro σ es importante, ya que de eso depende que se pueda encontrar una relación significativa entre las series de tiempo. La búsqueda de este valor puede realizarse intentando maximizar la información mutua entre las series de tiempo. Si se desea encontrar el parámetro σ que maximice la información mutua entre las series, será necesario unir las etapas de discretización y alineación debido a que el acoplamiento entre las secuencias discretas obtenidas depende de la combinación de estos dos parámetros.

Cuando el valor del parámetro σ ha sido previamente establecido, la discretización de las series de tiempo se puede separar de la etapa de alineación. En este caso, dado que las series de tiempo han sido sustituidas por secuencias discretas, se ha optado por utilizar un método basado en muestreo de Gibbs, empleado para la alineación de secuencias de ADN [Rouchka, 1997], de acuerdo al siguiente algoritmo:

MaxIter = Número máximo de iteraciones
$S = \{s_1, s_2, \ldots, s_m\}$ el conjunto de las *m* secuencias discretas
$\Theta = \{\theta_1, \theta_2, \ldots, \theta_m\}$ el conjunto de desplazamientos de las ventanas
Asignar a cada θ_i un valor aleatorio entre 0 y *d*
Mientras el número de iteraciones sea menor que *MaxIter*:
    Para cada s_i:
        $S = S \backslash \{s_i\}$
        Para cada posible valor *j* de θ_i
            $\theta_i = j$
            C_j = *CalificarConfiguraciónAlineación*()
        Para cada valor *j* de θi

$$p_j = \frac{C_j}{\sum_k C_k}$$

        Muestrear sobre las probabilidades p_j obtener el siguiente valor de θ_i

```
        S = S ∪{s_i}
        C_T = CalificarConfiguraciónAlineación()
Elegir la configuración de Θ que maximice C_T
```

Para calificar una configuración se forman los pares de aquellas secuencias entre las cuales la información mutua es máxima. Es decir, para cada secuencia *i* se busca aquella secuencia *j* con la cual se maximice la información mutua, y esta información mutua se añade al resultado:

```
CalificarConfiguraciónAlineación()
     Para cada secuencia s_i:
          Buscar s_j tal que I(s_i, s_j)≥I(s_i, s_k) ∀k
          C = C + I(s_i, s_j)
     End For
     Regresar C
```

En caso de que sea necesario buscar el valor del parámetro σ para cada serie, se utiliza un algoritmo basado en recocido simulado [Kirkpatrick et. al, 1983] que busca los valores σ y desplazamiento adecuados:

```
MaxIter = Número máximo de iteraciones
σ_δ1, σ_δ2,... σ_δm variación máxima de σ para cada serie
S = {s_1, s_2,..., s_m} el conjunto de las m series de tiempo
C_actual = 0
C_max = 0
Discretizar cada serie s_i utilizando σ_δi
Mientras el número de iteraciones sea menor que MaxIter:
     Elegir aleatoriamente una serie s_e a modificar
     Elegir un número aleatorio entre 0 y 1
     Si el número aleatorio es menor que 0.2, modificar el
      parámetro σ. En otro caso modificar el desplazamiento
     Elegir aleatoriamente la variación del parámetro
     Verificar que el valor resultante es válido
     Si el valor del parámetro no es válido
          Elegir el límite superior o inferior más cercano
     Aplicar la variación del parámetro a la serie de tiempo
     C_temp = CalificarConfiguraciónRecSim()
     Si AceptaCambio(C_temp, C_actual, (MaxIter-iteración)/MaxIter)
          C_actual = C_temp
          Si C_temp>C_max
               C_max = C_temp
     Else
          Reestablecer valor anterior del parámetro
Elegir la configuración que produjo C_max
```

```
AceptaCambio(C_temp, C_actual, temperatura)
     Si C_temp≥C_actual
          Regresar verdadero
                  (C_temp − C_actual) / temperatura
     P_aceptar = exp
     Obtener un número aleatorio x entre 0 y 1
     Si x<P_aceptar
          Regresar verdadero
     Regresar falso
```

En este caso, la calificación de la configuración debe tomar en cuenta, además de la información mutua entre las secuencias discretas, la calidad con que cada una de éstas reproduce la serie original.

```
CalificarConfiguraciónRecSim()
     CalidadRec = 0
     Para cada secuencia s_i:
          Pot_i = Potencia de la serie i
          Err_i = Error cuadrado de discretización de s_i
          CalidadRec = CalidadRec + log(Pot_i/Err_i)
     End For
     Regresar CalidadRec*CalificarConfiguraciónAlineación()
```

6.3 Pruebas y resultados

Para probar el procedimiento propuesto se ha utilizado un conjunto de series de tiempo extraídas principalmente de variables sociales y económicas. Los datos utilizados fueron obtenidos del Banco de Información Económica (BIE) generado por el INEGI, y representan mediciones mensuales tomadas de Enero de 1993 a Septiembre del 2002.

En todos los casos se obtuvieron los valores σ para las series utilizando el algoritmo basado en recocido simulado (ver sección 6.2), utilizando para todas las series una variación máxima de σ igual a 1000 unidades y un desplazamiento máximo de 18 meses.

Recuerde de la sección 1.2.3 que el proceso para la extracción de Redes Bayesianas predictivas consta de dos etapas. En la primera, discretización y alineación, se obtienen secuencias discretas a partir de las series de tiempo. En la segunda, extracción, se obtiene la Red Bayesiana a partir de las secuencias discretas.

En cada una de las pruebas que se presentan a continuación, se muestran las series de tiempo utilizadas, el valor de σ utilizado para discretizar cada una de ellas, las secuencias de valores obtenidos a partir de la discretización, y el desplazamiento de cada secuencia después de la etapa de alineación. En el capítulo 7 se muestran las estructuras de las Redes Bayesianas obtenidas a partir de las secuencias alineadas utilizando, cuando es posible, el algoritmo MLE y el algoritmo de tres etapas.

6.3.1 Prueba 1

Las figuras 6.3, 6.4, 6.5 y 6.6 muestran cuatro series de tiempo: el Indicador Global de la Actividad Económica, la Tasa de Desempleo Abierto, el último valor para cada mes del Indice de Precios y Cotizaciones de la Bolsa Mexicana de Valores y el Indice Nacional de Precios al Consumidor, respectivamente.

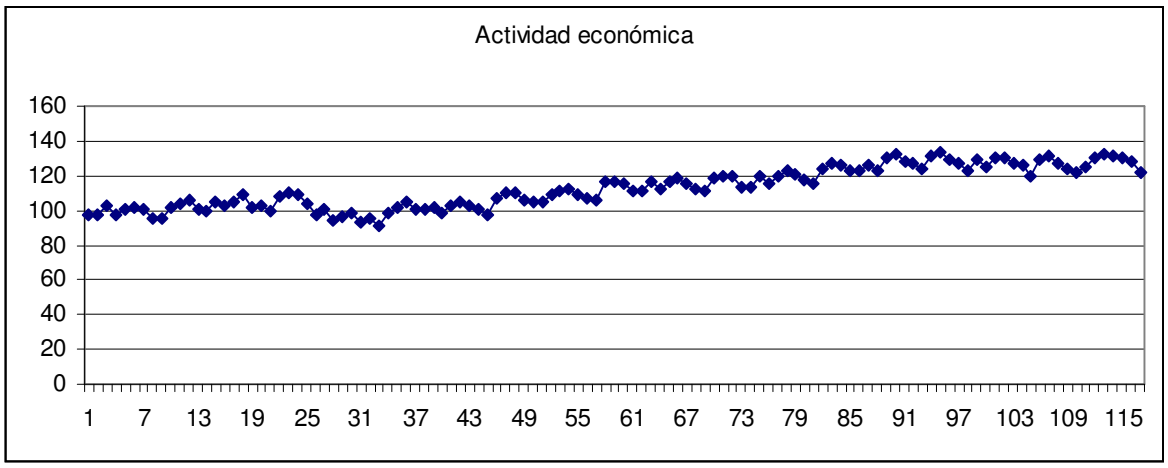


Figura 6.3. Serie de tiempo *Actividad económica*

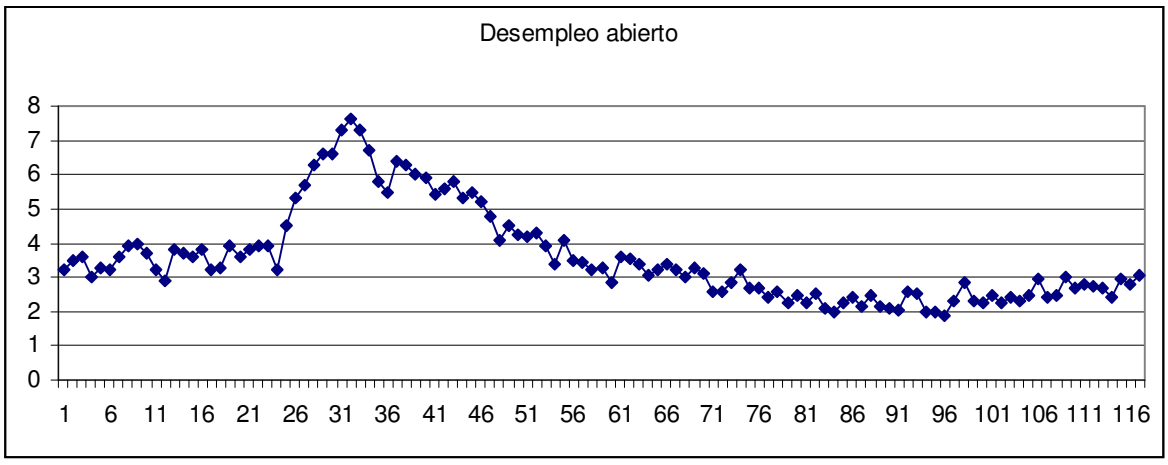


Figura 6.4. Serie de tiempo *Desempleo abierto*

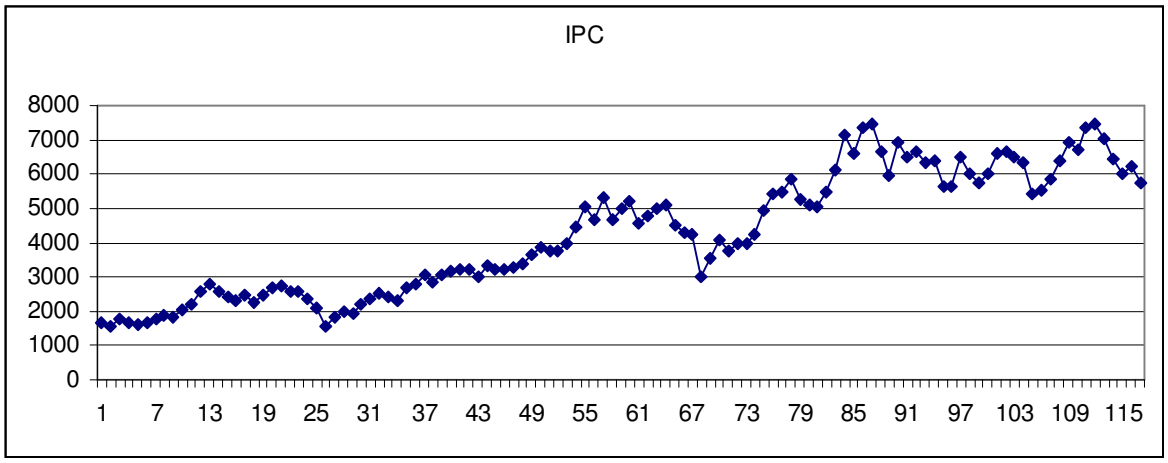


Figura 6.5. Serie de tiempo *IPC*

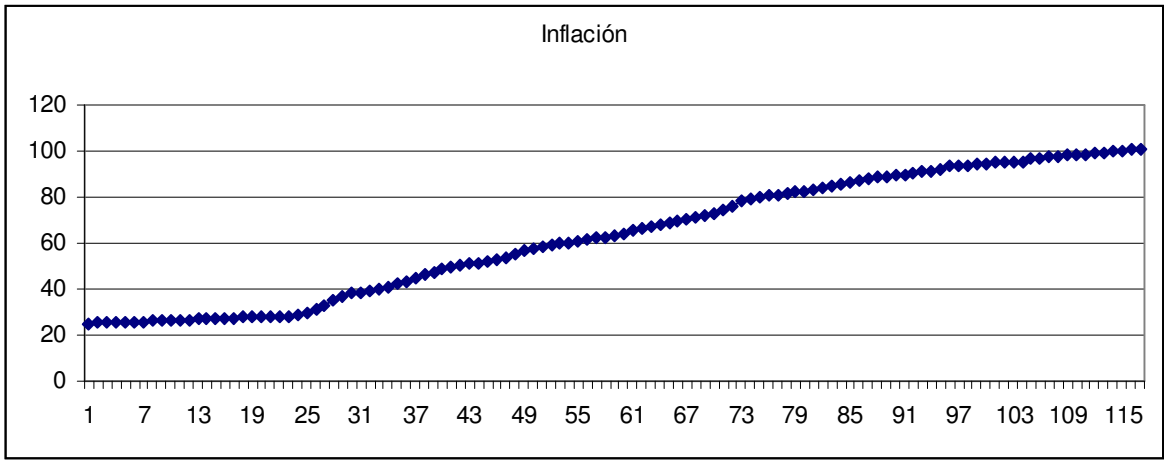


Figura 6.6. Serie de tiempo *Inflación*

Para discretizar las series de tiempo se utilizaron 4 símbolos. Dado que el parámetro σ representa el peso que tendrá la pendiente durante la discretización (ver capítulo 5), se buscan aquellos valores que maximicen la información mutua entre las secuencias discretas, intentando además que se minimice el ruido por discretización. Después de examinar varios valores se eligieron los mostrados en la tabla 6.1.

Serie de tiempo	σ
Actividad económica	0
Desempleo abierto	0
IPC	302.145
Inflación	4128.888

Tabla 6.1. Valores de σ utilizados para discretizar las series de tiempo

Las secuencias discretas obtenidas se muestran en las figuras 6.7, 6.8, 6.9 y 6.10.

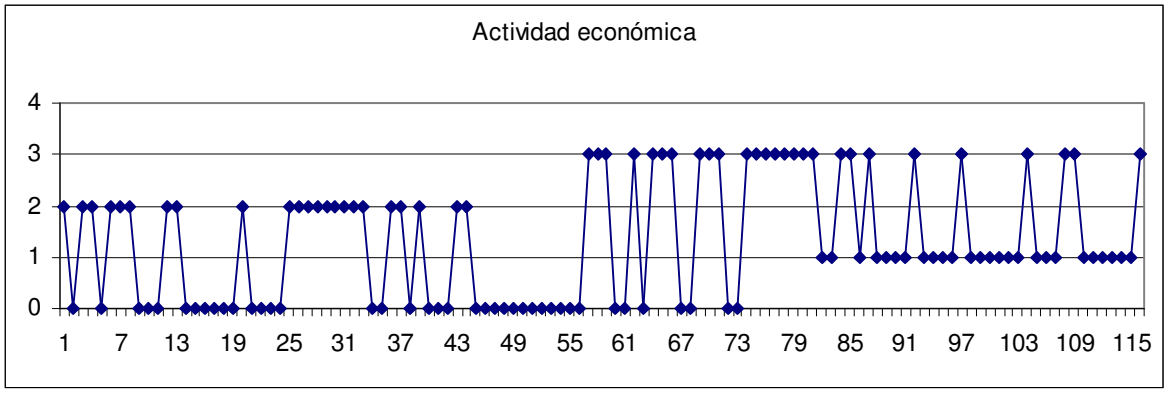


Figura 6.7. Discretización de *Actividad económica*

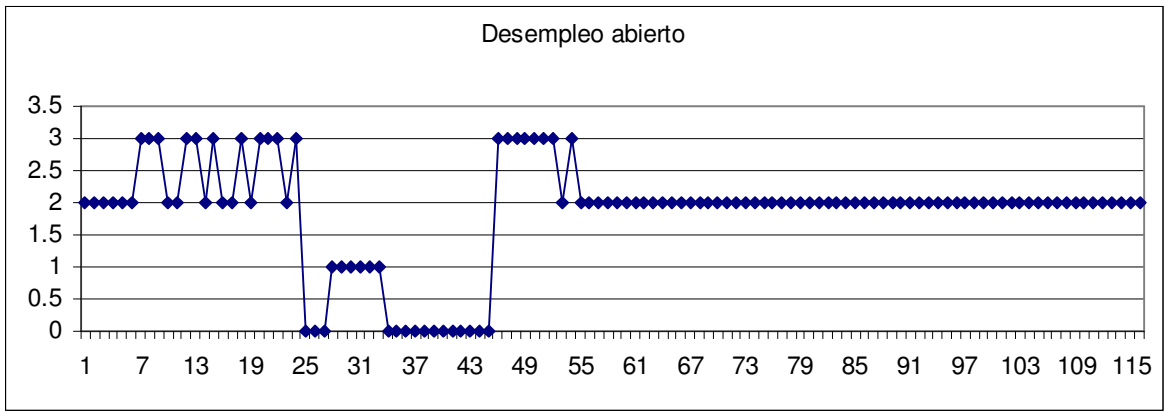


Figura 6.8. Discretización de *Desempleo abierto*

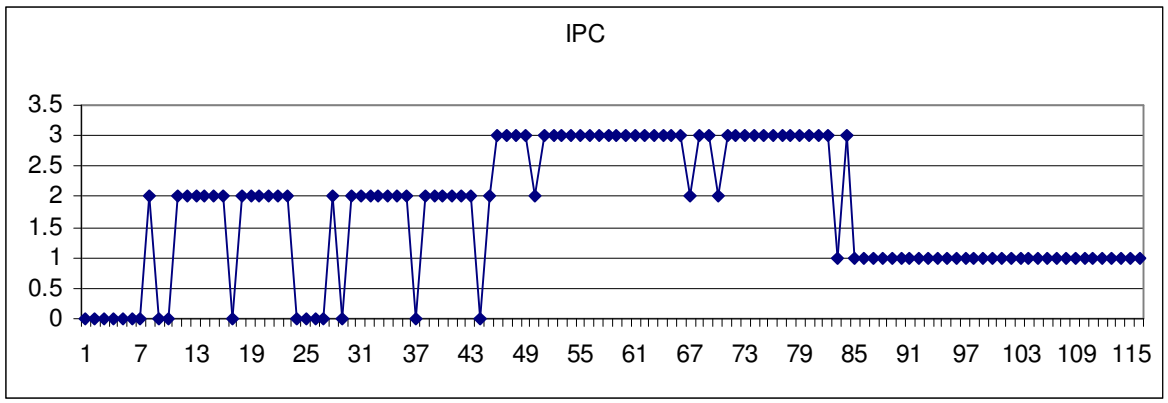


Figura 6.9. Discretización de *IPC*

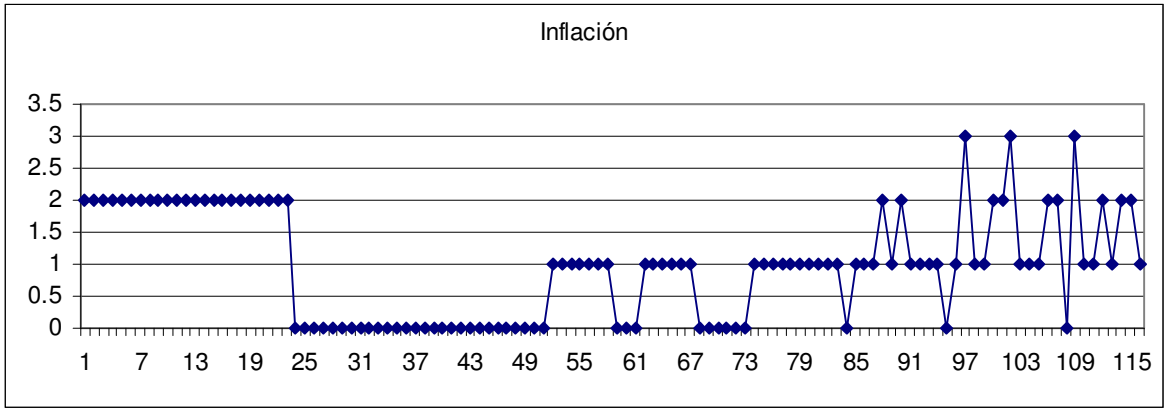


Figura 6.10. Discretización de *Inflación*

Es interesante observar la relación que guardan las secuencias discretas obtenidas, las series de tiempo originales y los hechos que éstas representan. Como ejemplo se puede observar la secuencia discreta obtenida a partir de la serie de tiempo *Inflación* (figura 6.10). Note que existe un cambio visible en el comportamiento de la secuencia al pasar del dato número 23 al número 24. Tomando en cuenta que el método de discretización utilizado descarta el primer valor de la serie original debido a que requiere la diferencia del punto actual respecto al anterior, el valor discreto número 24 representa el mes de Diciembre de 1994. Se entiende que la inflación se vio afectada por la crisis económica que tuvo su inicio durante ese mes, fenómeno que se refleja en las variaciones de la serie original, la cual pasó de 0.248 de Noviembre a Diciembre de 1994 (28.605 en Diciembre menos 28.357 en Noviembre) a 1.077 de Diciembre de 1994 a Enero de 1995 (29.682 en Enero menos 28.605 en Diciembre). Otros cambios importantes en el comportamiento de la secuencia discreta se observan entre los valores 51 y 52 (correspondientes a Abril y Mayo de 1997) y a partir del punto 96 (correspondiente a Enero de 2001) en adelante.

También se observa un cambio significativo entre los puntos 24 y 25 (correspondientes a Enero y Febrero de 1995 respectivamente) de la secuencia discreta correspondiente a la serie de tiempo *Desempleo abierto*. En este caso, el cambio es mucho más visible en la serie original, debido a que se presenta en su magnitud. Así, la serie original pasa de 3.2 en Enero de 1995 a 4.5 en Febrero y a 5.3 en Marzo del mismo año.

Las secuencias discretas quedan alineadas con los desplazamientos que se muestran en la tabla 6.2.

Secuencia discreta	Desplazamiento
Actividad económica	3
Desempleo abierto	10
IPC	0
Inflación	1

Tabla 6.2. Desplazamientos obtenidos al alinear las secuencias discretas

6.3.2 Prueba 2

Considérense las series de tiempo mostradas en las figuras 6.11, 6.12, 6.13 y 6.14, correspondientes a las Exportaciones en Millones de Dólares, el Indice Nacional de Precios al Productor tomando como base el año 1994 = 100, la Productividad de la Mano de Obra en la Industria Manufacturera tomando como base el año 1993 = 100, y el Costo Porcentual Promedio de Captación en Moneda Nacional (CPP) respectivamente.

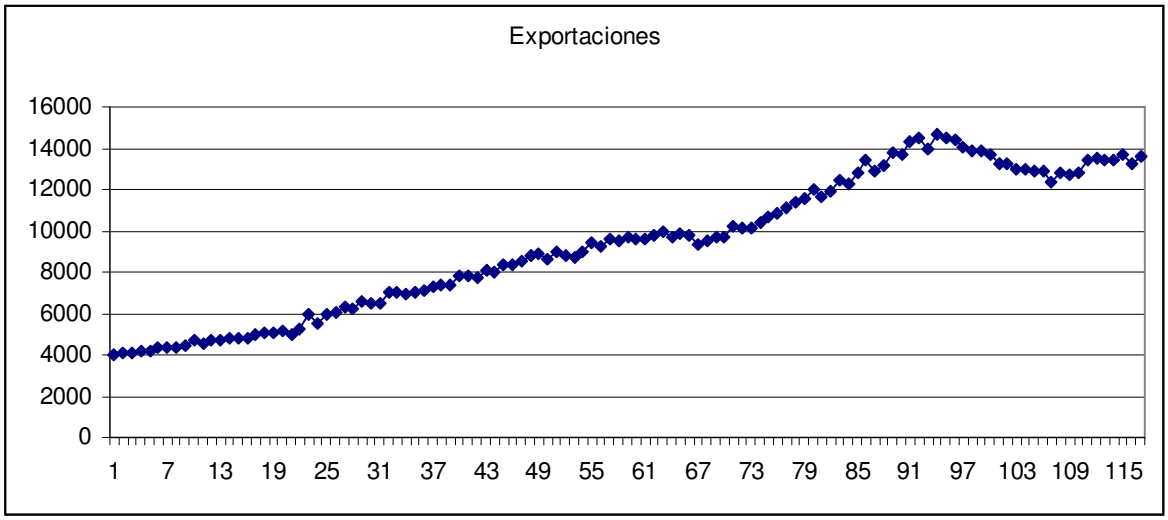


Figura 6.11. Serie de tiempo *Exportaciones*

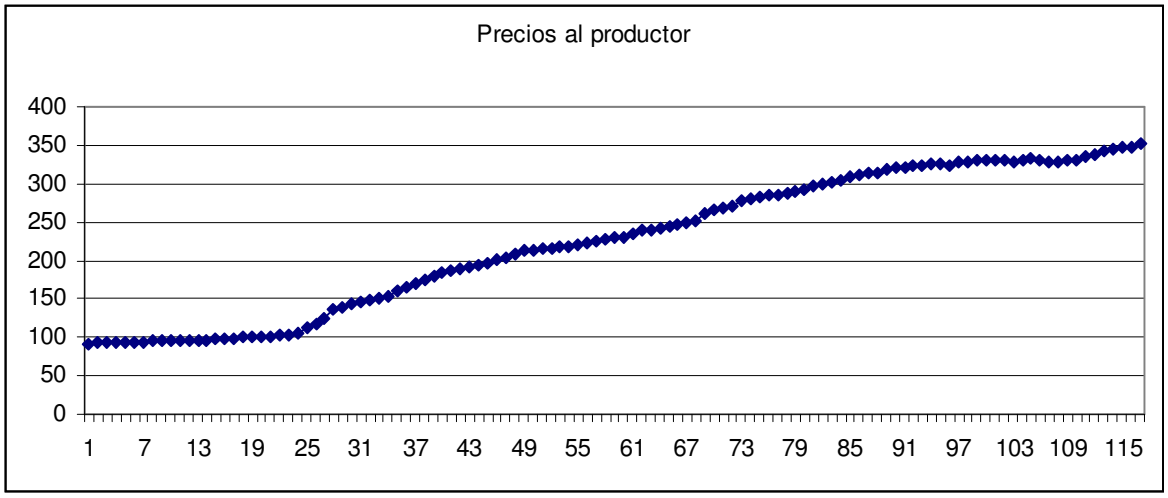


Figura 6.12. Serie de tiempo *Precios al productor*

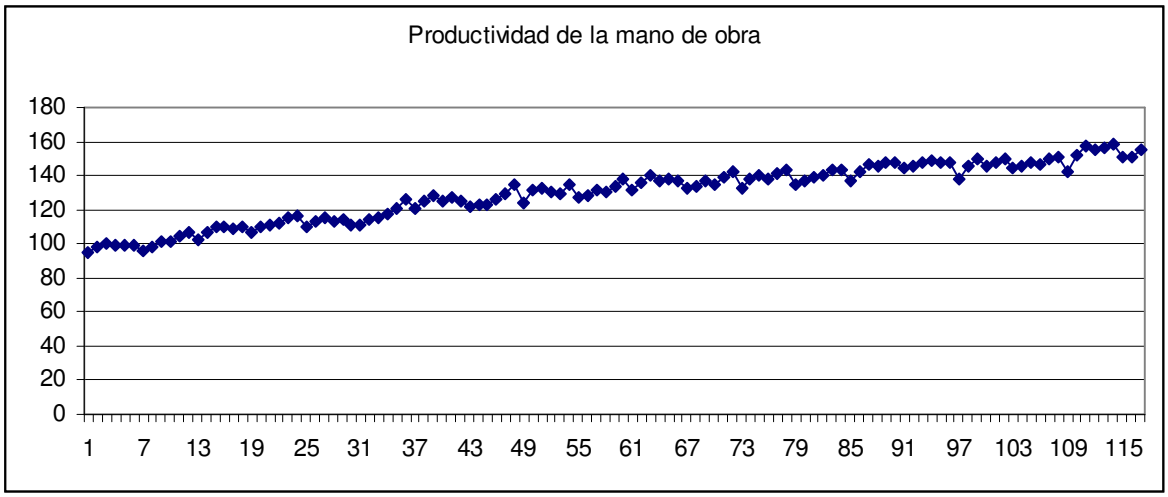


Figura 6.13. Serie de tiempo *Productividad de la mano de obra*

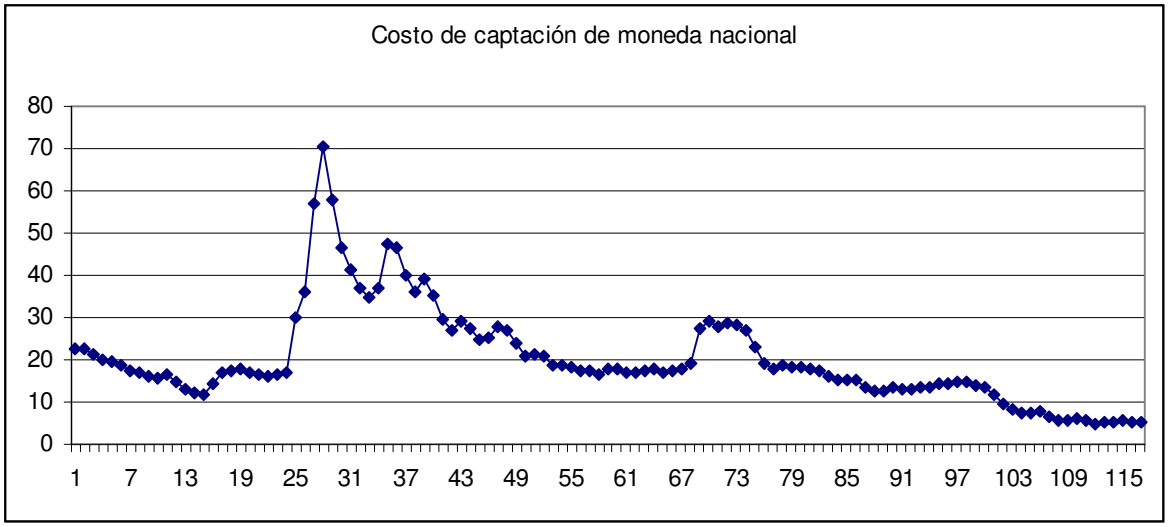


Figura 6.14. Serie de tiempo *Costo de captación de moneda nacional*

Las series de tiempo anteriores fueron discretizadas utilizando 4 símbolos y los valores de σ que se muestran en la tabla 6.3.

Serie de tiempo	σ
Exportaciones	21572.473
Precios al productor	1337.463
Productividad de la mano de obra	0
Costo de captación de moneda nacional	0

Tabla 6.3. Valores de σ utilizados para la discretización de las series de tiempo

Las secuencias discretas obtenidas a partir de las series de tiempo se muestran en las figuras 6.15, 6.16, 6.17 y 6.18.

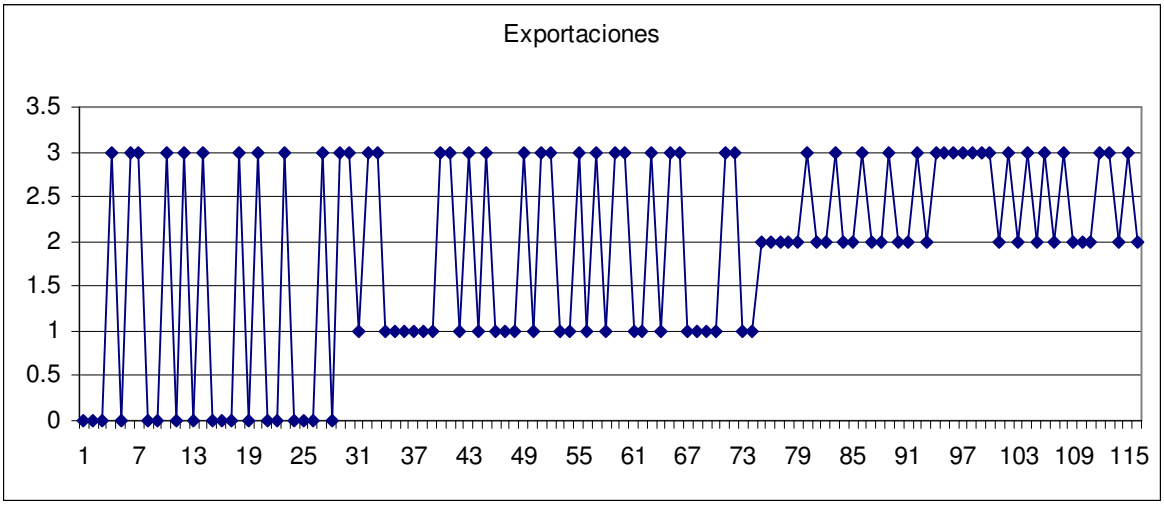


Figura 6.15. Discretización de *Exportaciones*

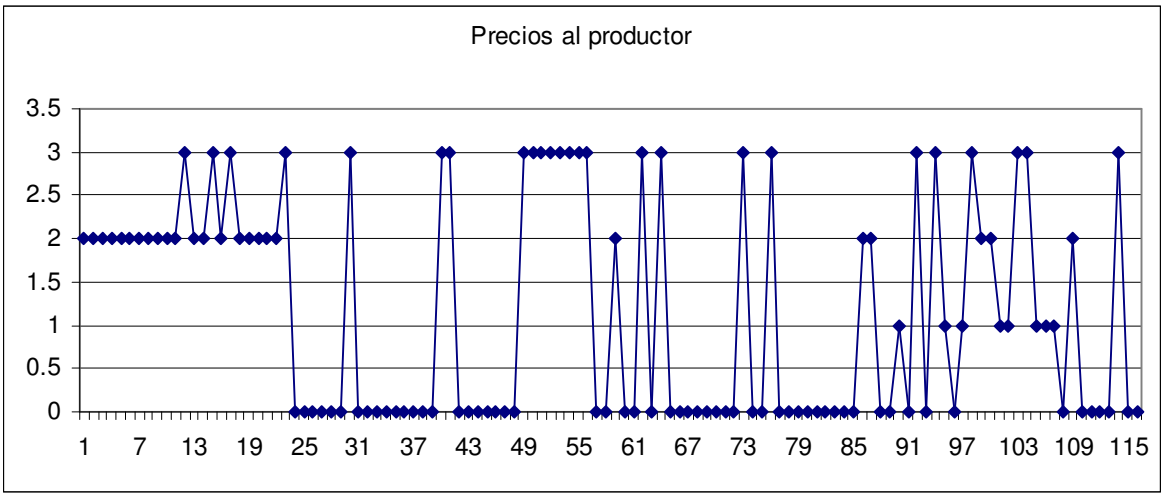


Figura 6.16. Discretización de *Precios al productor*

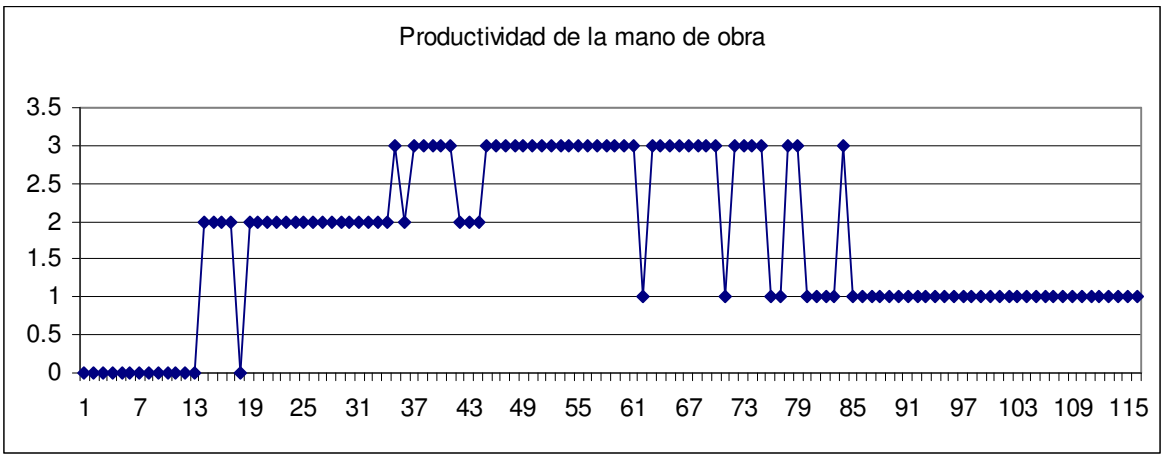


Figura 6.17. Discretización de *Productividad de la mano de obra*

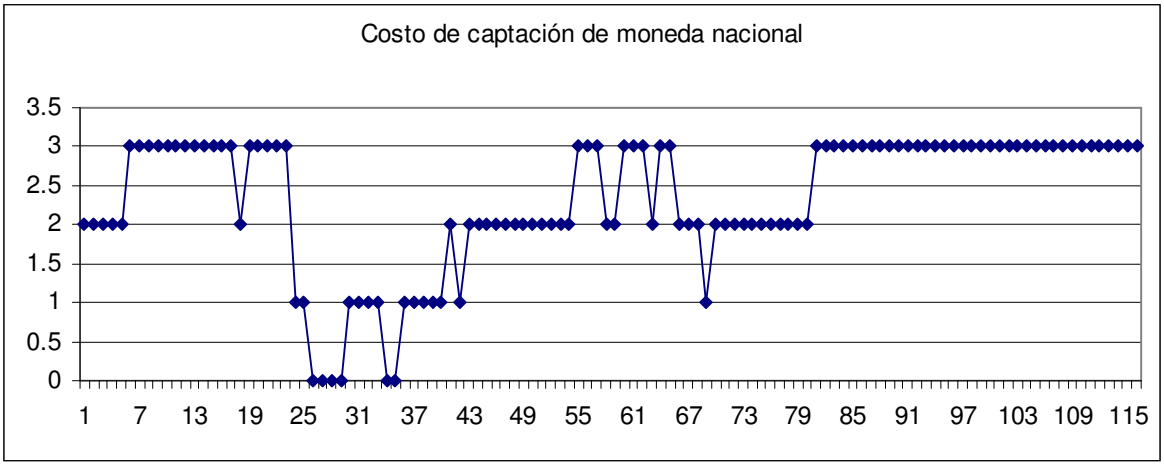


Figura 6.18. Discretización de *Costo de captación de moneda nacional*

Retomando la crisis de Diciembre de 1994, se puede ver claramente como las secuencias discretas obtenidas de las series *Precios al productor* y *Costo de captación de moneda nacional* muestran sus efectos de manera casi inmediata. Ambas secuencias presentan un cambio de comportamiento entre los puntos 23 y 24, correspondientes a Diciembre de 1994 y Enero de 1995 respectivamente.

Al alinear éstas secuencias se obtienen los desplazamientos que se muestran en la tabla 6.4.

Secuencia discreta	Desplazamiento
Exportaciones	1
Precios al productor	1
Productividad de la mano de obra	0
Costo de captación de moneda nacional	10

Tabla 6.4. Desplazamiento de las secuencias discretas

6.3.3 Prueba 3

En esta prueba se utilizaron diez series de tiempo obtenidas a partir de variables de tipo económico. Entre las variables utilizadas se encuentran *Actividad económica* (figura 6.3), *Desempleo abierto* (figura 6.4), *IPC* (figura 6.5), *Inflación* (figura 6.6), *Exportaciones* (figura 6.11) y *Costo de captación de Moneda Nacional* (figura 6.14).

Además de las series mencionadas, se utilizó la Colocación de Deuda Interna del Sector Público a Través de Valores (figura 6.19), las Importaciones (figura 6.20), la Producción de Productos Petrolíferos en Miles de Barriles por Día (figura 6.21) y los Salarios en la Industria Manufacturera en Dólares por Hora Hombre (figura 6.22).

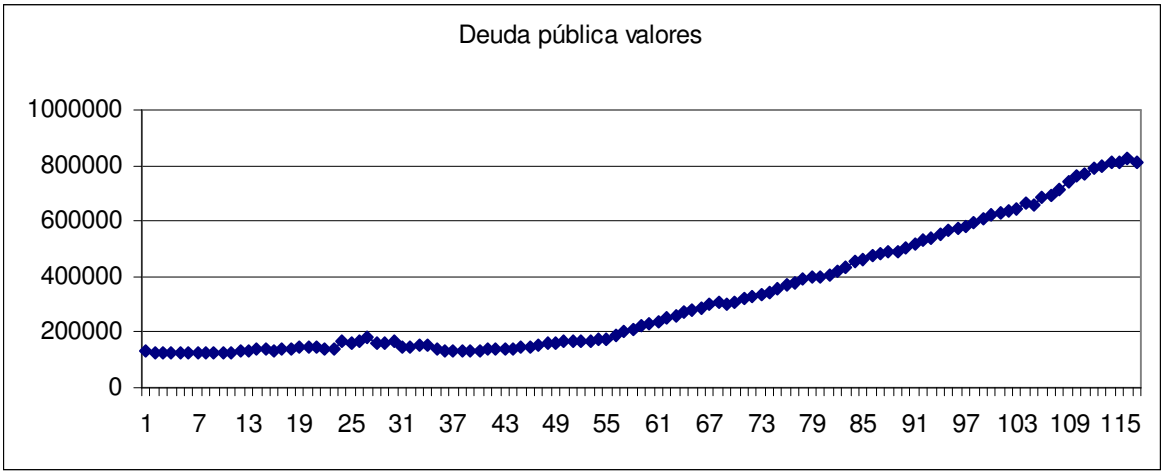


Figura 6.19. Serie de tiempo *Deuda pública valores*

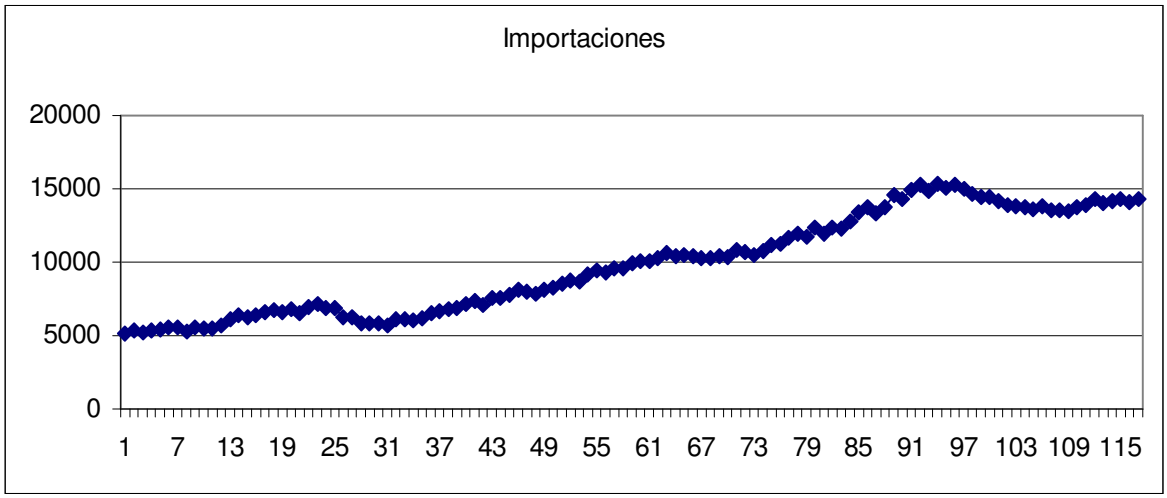


Figura 6.20. Serie de tiempo *Importaciones*

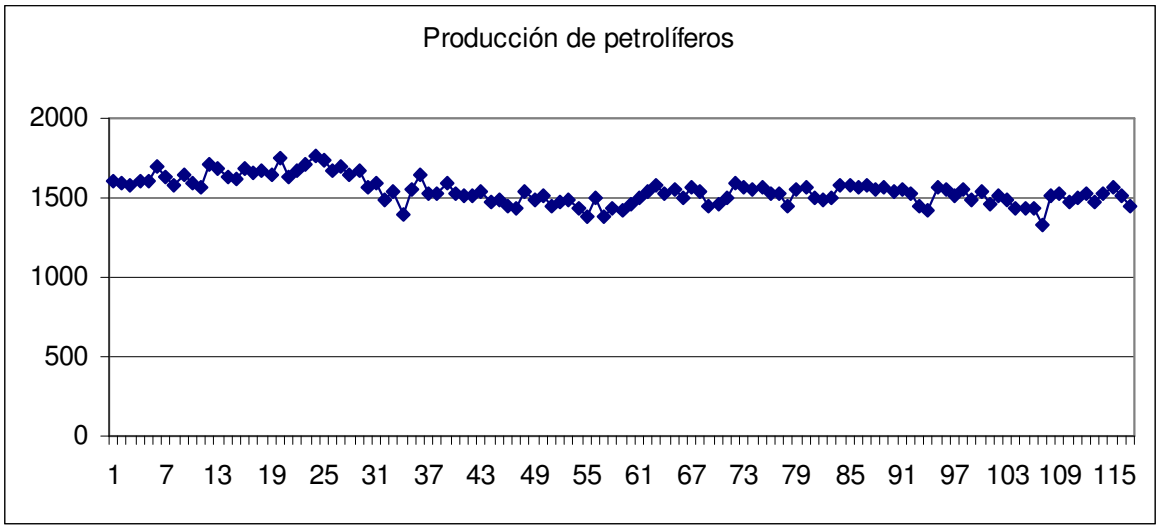


Figura 6.21. Serie de tiempo *Producción de petrolíferos*

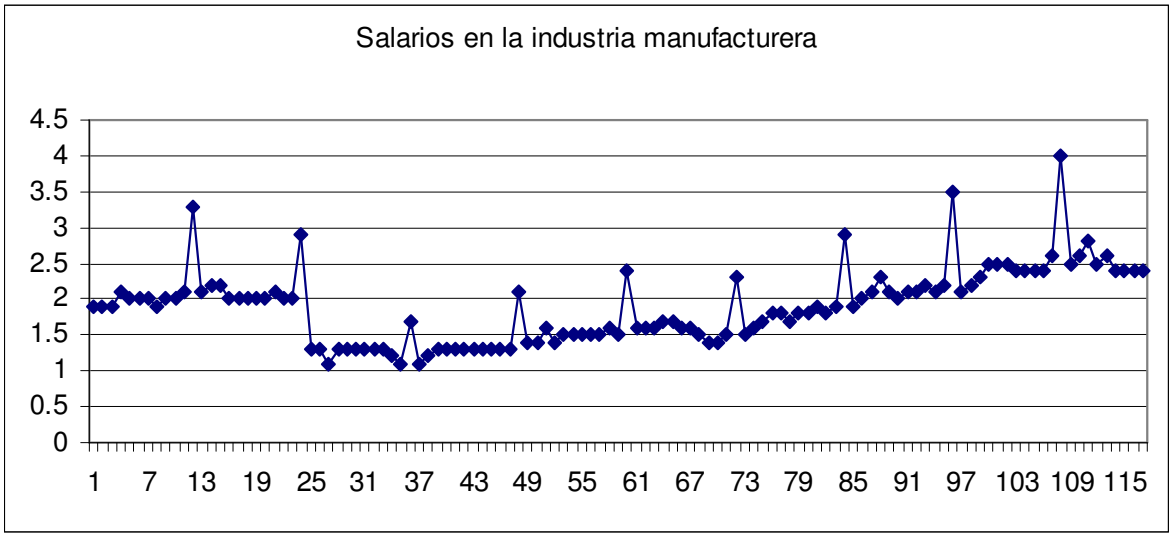


Figura 6.22. Serie de tiempo *Salarios industria Manufacturera*

Las series de tiempo fueron discretizadas utilizando el valor de σ que se indica en la tabla 6.5.

Serie de tiempo	σ
Desempleo abierto	0
IPC BMV	9.023
Inflación	11000.041
Actividad económica	0
Exportaciones	26377.109
Deuda pública valores	17500.781
Costo de captación de Moneda Nacional	0
Importaciones	40.823
Producción de petrolíferos	1.742
Salarios en la industria manufacturera	0

Tabla 6.5. Valores de σ utilizados para la discretización de las series de tiempo

Las secuencias discretas obtenidas a partir de éstas series de tiempo se muestran en las figuras 6.23, 6.24, 6.25, 6.26, 6.27, 6.28, 6.29, 6.30, 6.31 y 6.32.

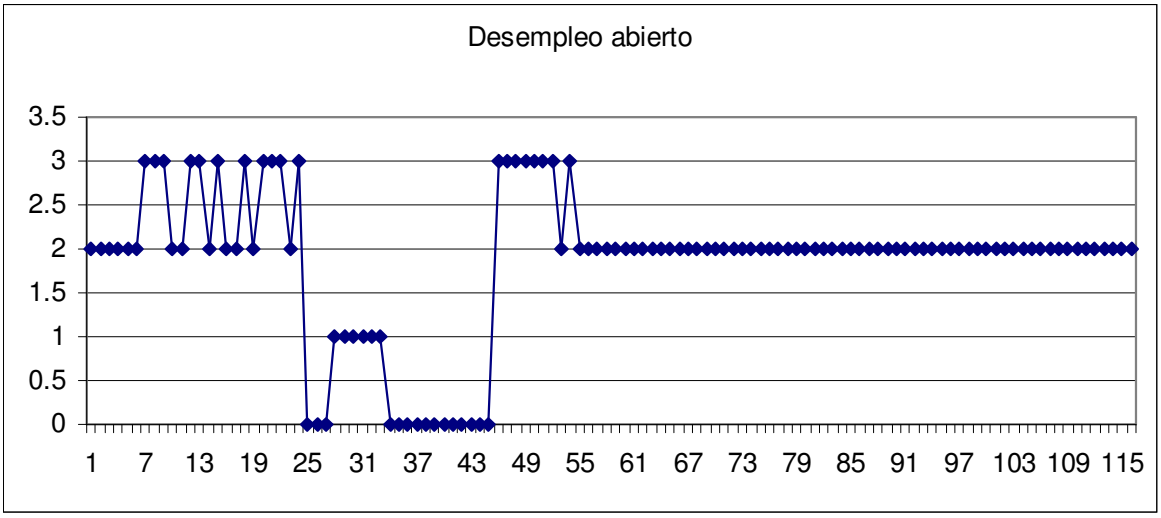


Figura 6.23. Discretización de *Desempleo abierto*

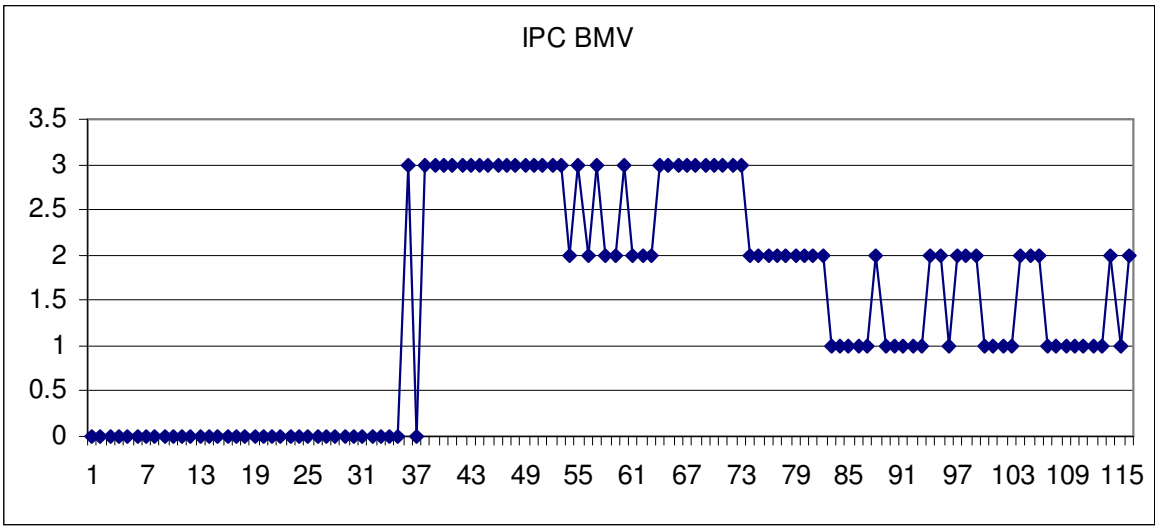


Figura 6.24. Discretización de *IPC BMV*

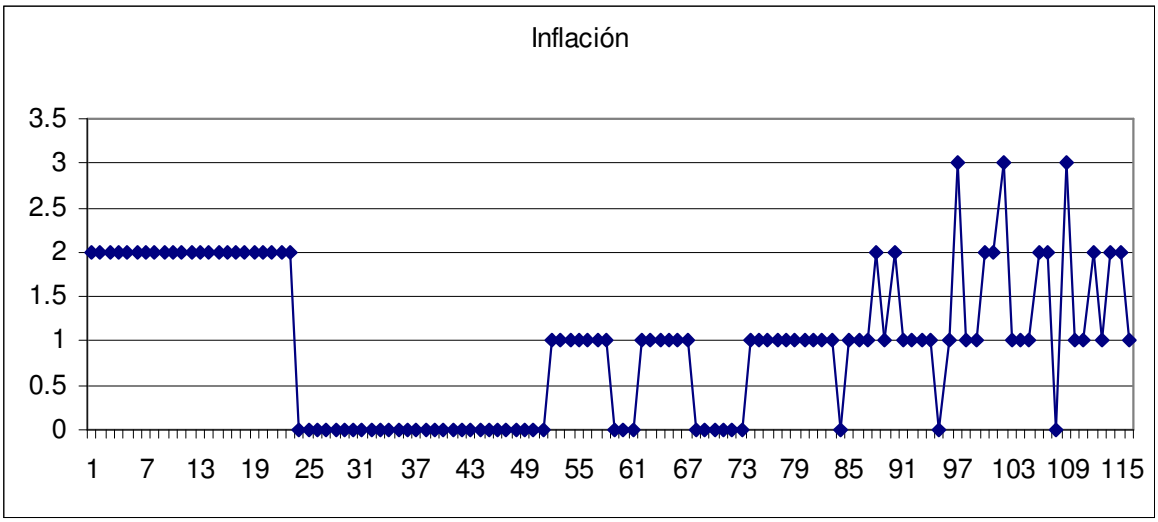


Figura 6.25. Discretización de *Inflación*

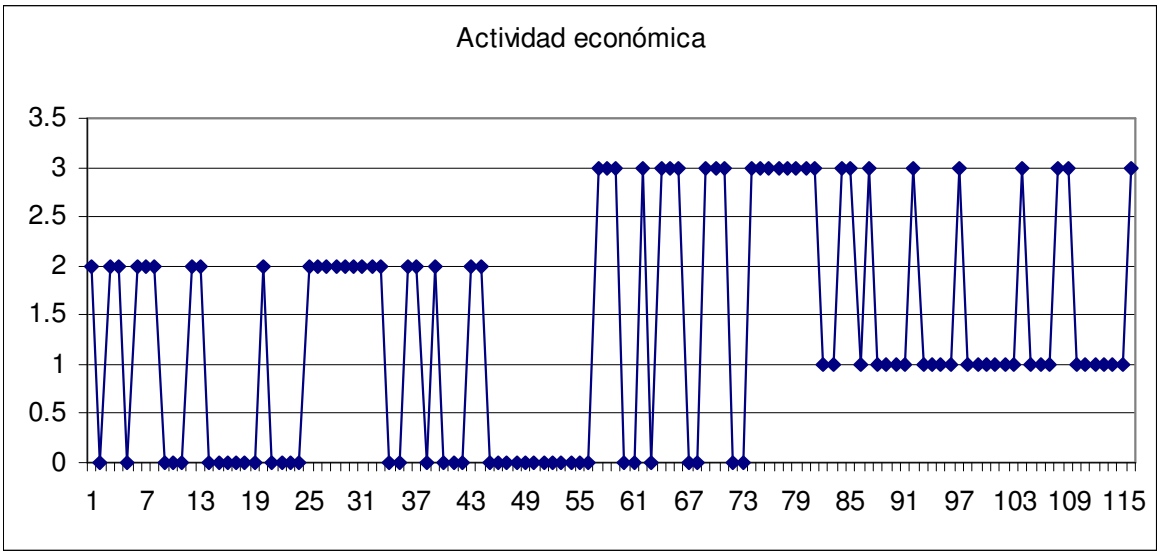


Figura 6.26. Discretización de *Actividad económica*

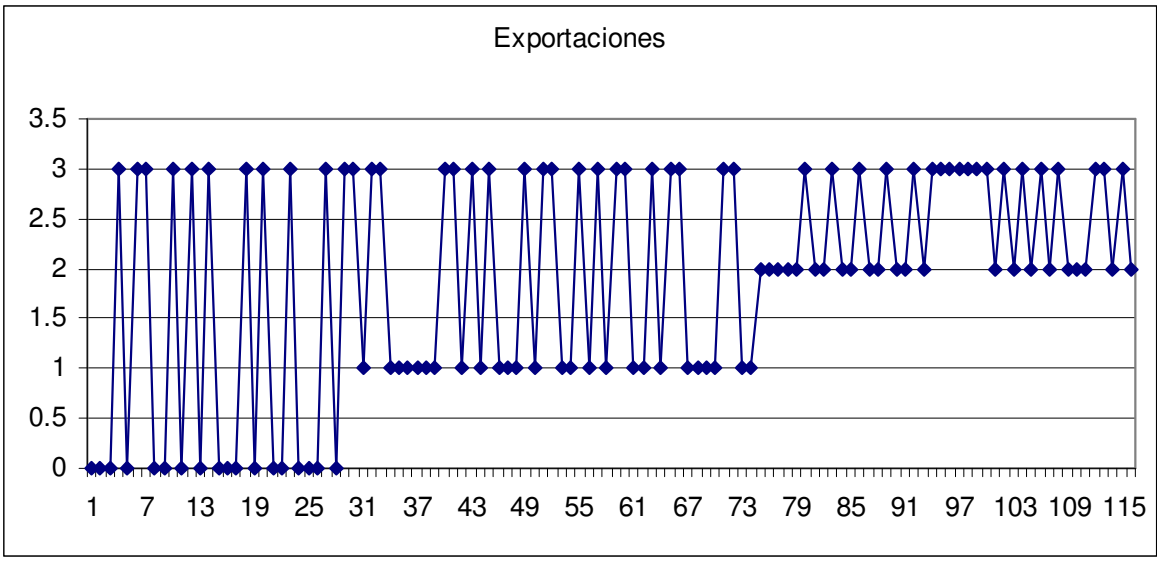


Figura 6.27. Discretización de *Exportaciones*

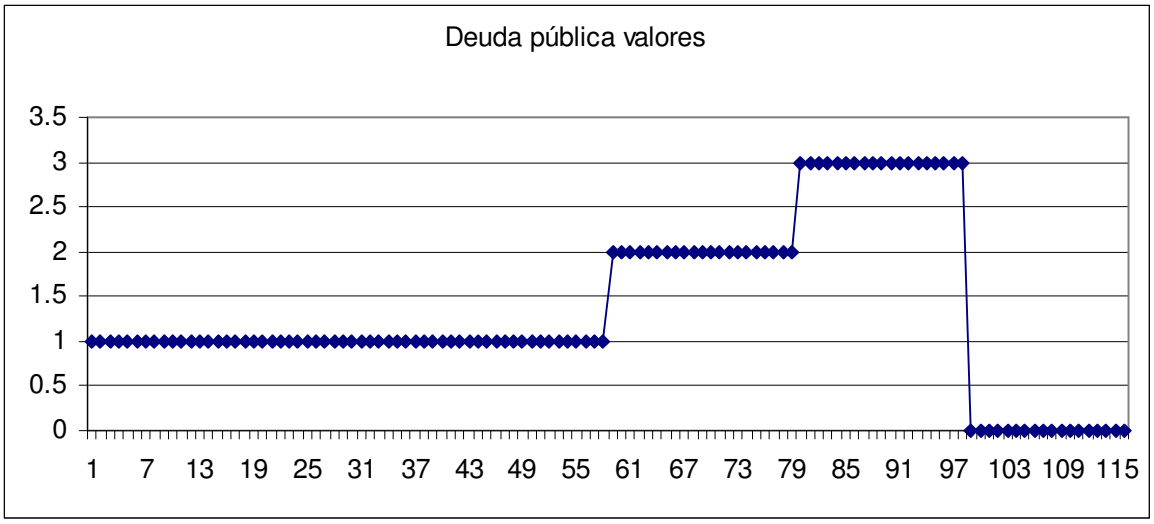


Figura 6.28. Discretización de *Deuda pública valores*

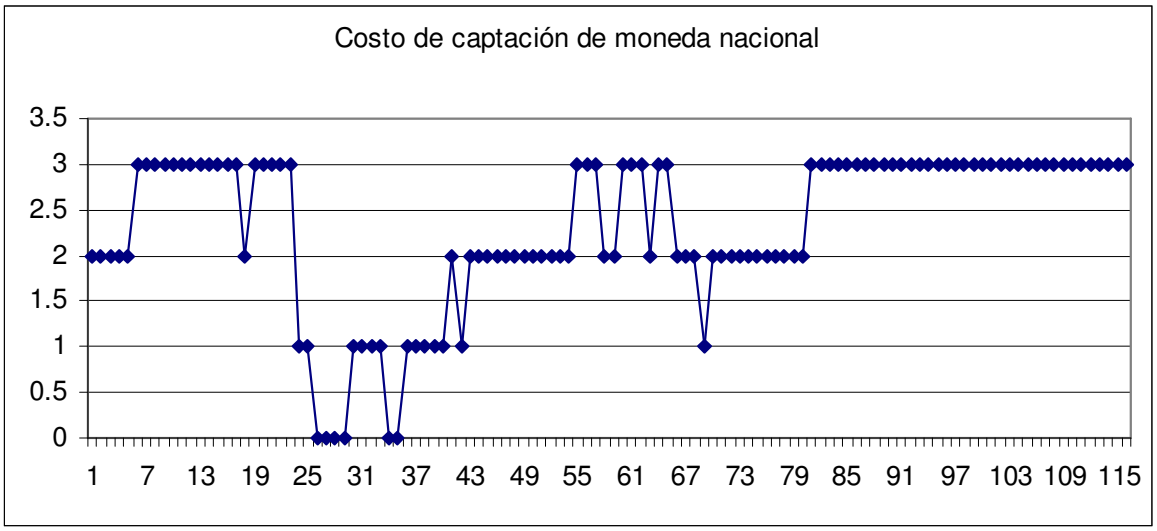


Figura 6.29. Discretización de *Costo de captación de Moneda Nacional*

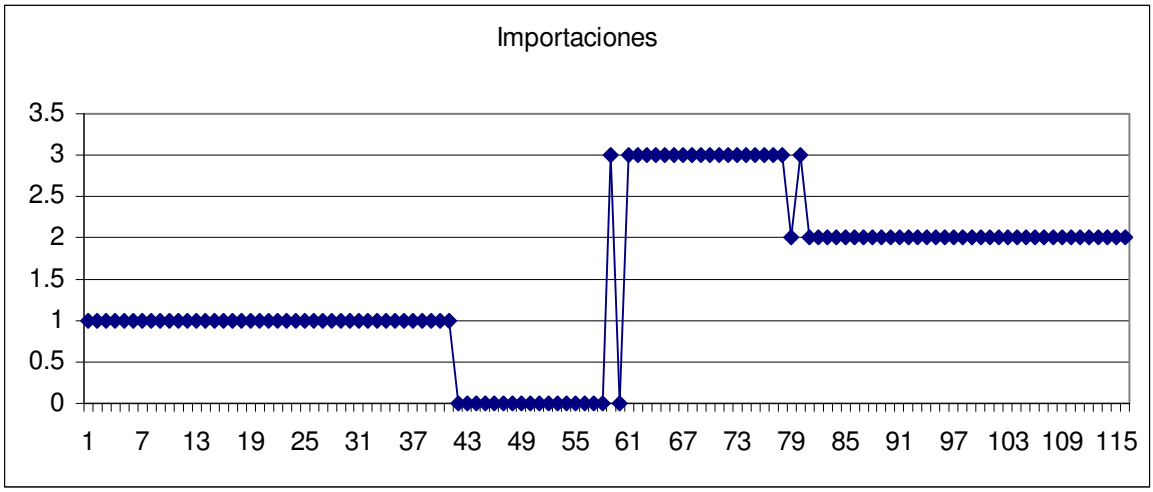


Figura 6.30. Discretización de *Importaciones*

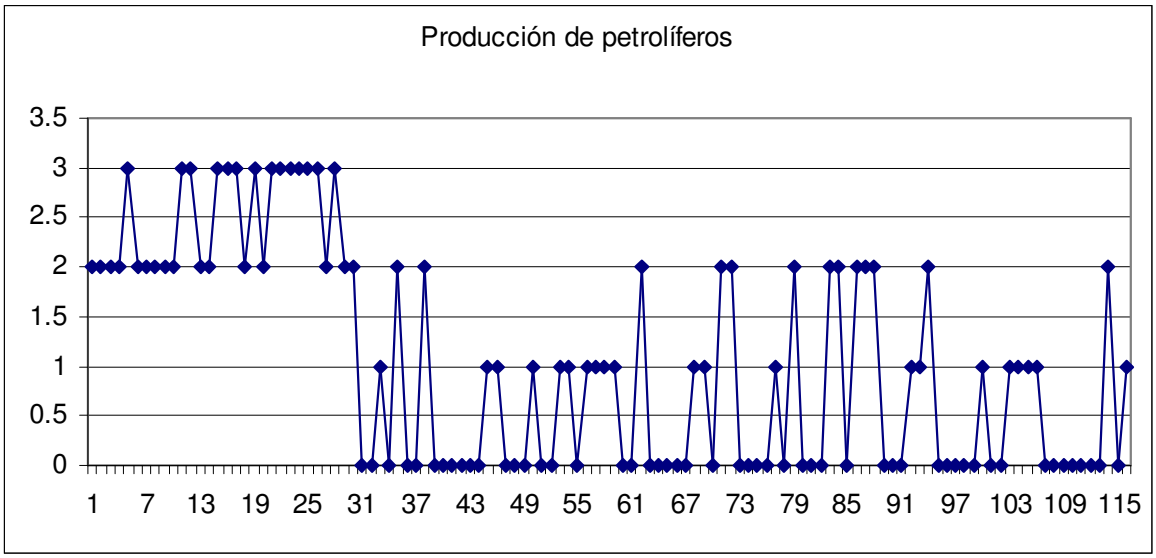


Figura 6.31. Discretización de *Producción de petrolíferos*

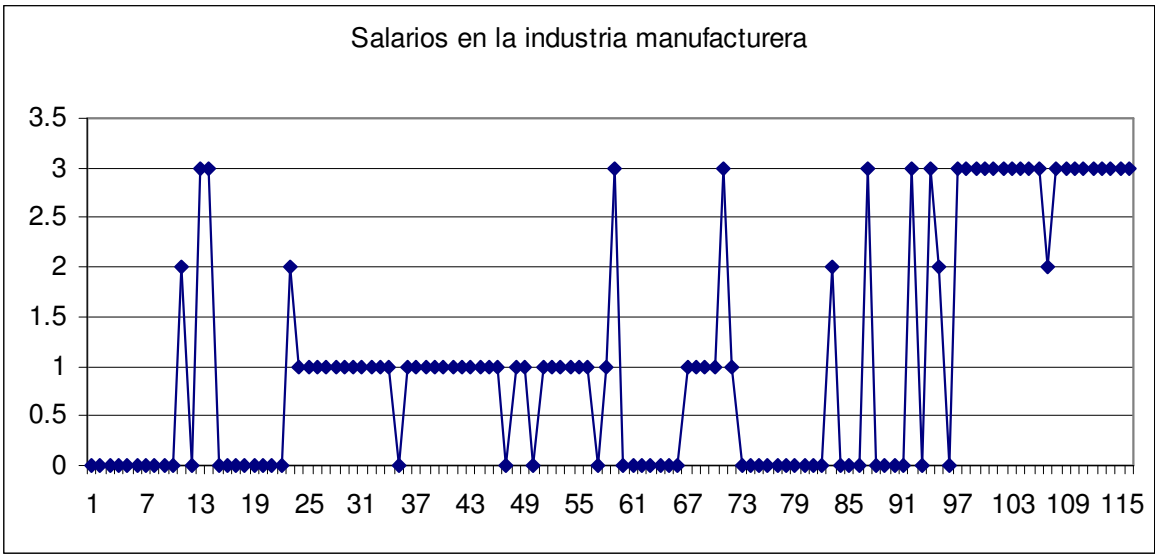


Figura 6.32. Discretización de *Salarios en la industria manufacturera*

Entre las variables utilizadas en pruebas anteriores, solamente *IPC* presenta una discretización distinta. Las variables restantes mantuvieron un valor de σ igual o muy similar durante los procesos de discretización-alineación, efectuados con distintos conjuntos de series de tiempo.

Entre las nuevas variables discretizadas, se observa que *Salarios en la industria manufacturera* presenta un cambio de comportamiento a partir del punto 24, correspondiente a Enero de 1995. Por su parte, *Actividad económica* presenta un cambio de comportamiento, a partir del punto 25 correspondiente a Febrero de 1995. Al alinear las diez series de tiempo se producen los desplazamientos que se presentan en la tabla 6.6.

Secuencia discreta	Desplazamiento
Desempleo abierto	16
IPC BMV	0
Inflación	3
Actividad económica	15
Exportaciones	0
Deuda pública valores	18
Costo captación MN	10
Importaciones	0
Producción petrolíferos	2
Salarios industria manufacturera	16

Tabla 6.6. Desplazamiento de las secuencias discretas

6.3.4 Prueba 4

En esta prueba se utilizaron seis series de tiempo. Las series *Actividad económica* (figura 6.3), *Inflación* (figura 6.6), *Deuda pública en valores* (figura 6.19) y *Producción de petrolíferos* (figura 6.21) se utilizaron en pruebas anteriores. Las otras dos series utilizadas

representan la Tasa de Interés Real Anual y el Gasto del Sector Público en Millones de Pesos a Precios Corrientes, mostradas en las figuras 6.33, y 6.34 respectivamente.

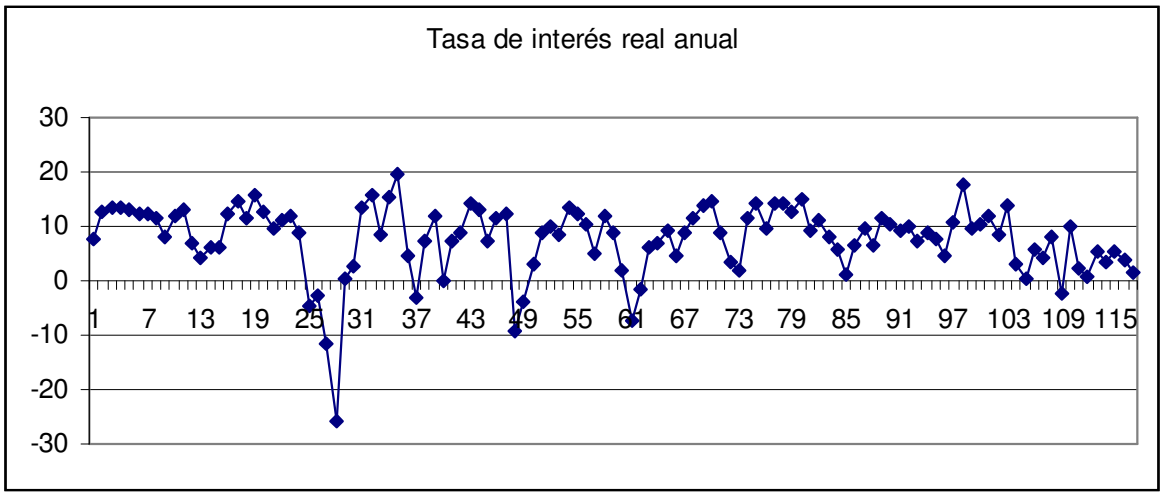


Figura 6.33. Serie de tiempo *Tasa de interés real anual*

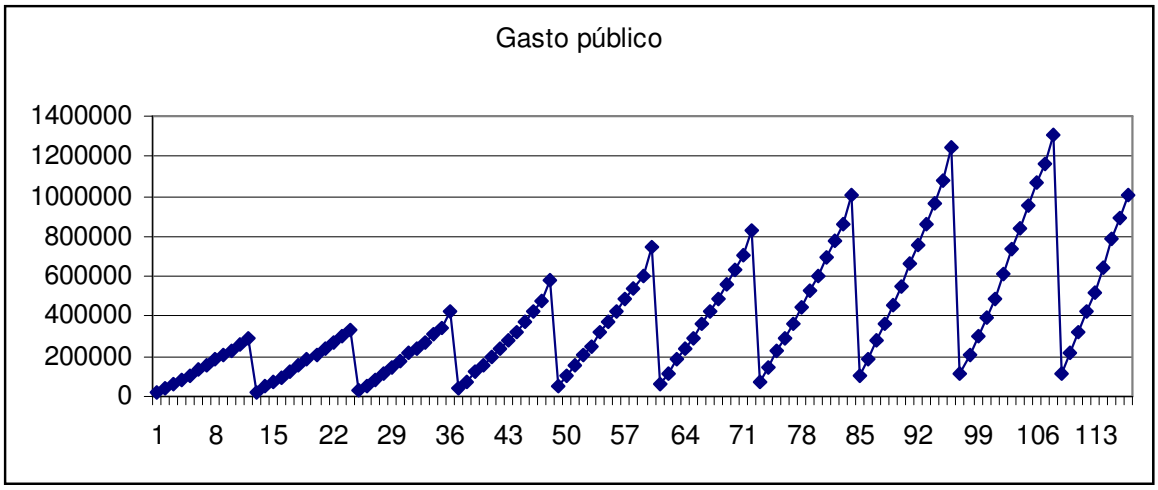


Figura 6.34. Serie de tiempo *Gasto público*

La tabla 6.7 muestra el valor de σ utilizado para discretizar cada una de estas series de tiempo.

Serie de tiempo	σ
Actividad económica	0
Inflación	0
Deuda pública en valores	7722.524
Producción de petrolíferos	0
Tasa de interés real anual	0
Gasto público	192.196

Tabla 6.7. Valores de σ utilizados para la discretización de las series de tiempo

Las figuras 6.35, 6.36, 6.37, 6.38, 6.39 y 6.40 muestran las secuencias discretas obtenidas de las series de tiempo recién presentadas.

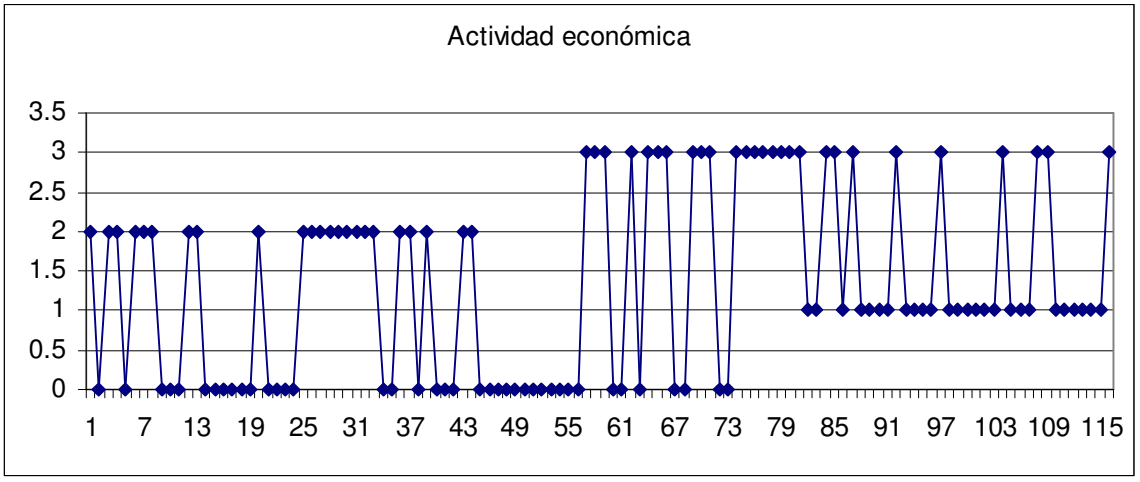


Figura 6.35. Discretización de *Actividad económica*

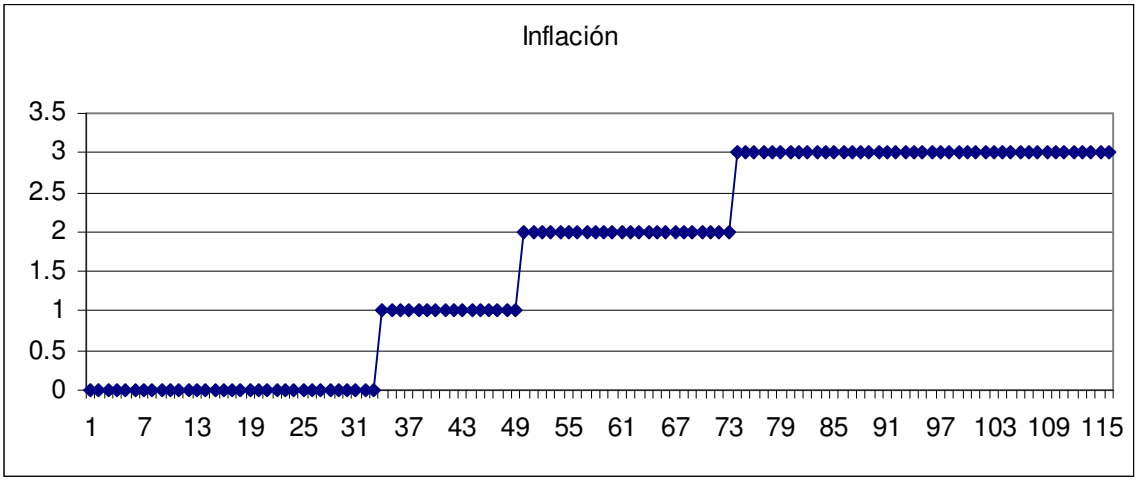


Figura 6.36. Discretización de *Inflación*

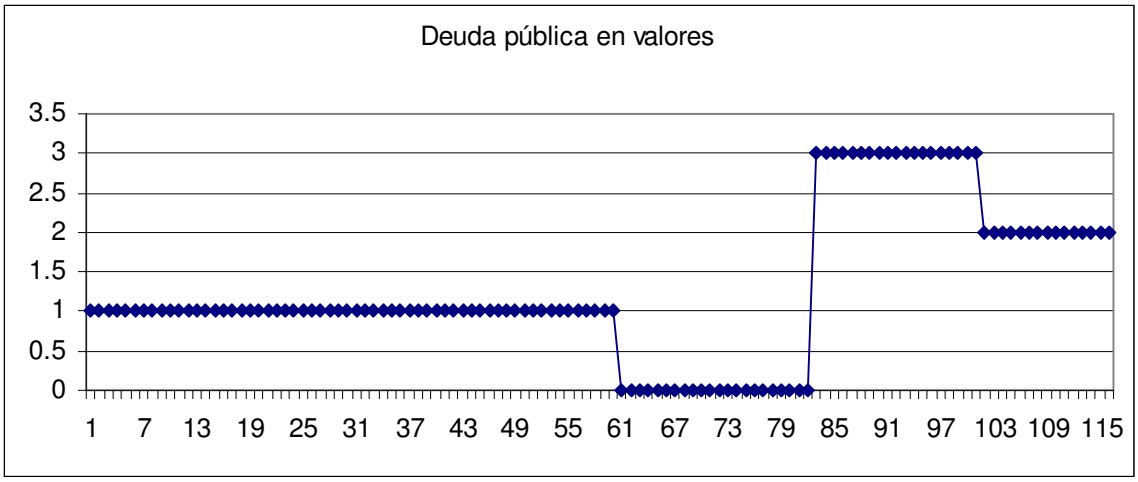


Figura 6.37. Discretización de *Deuda pública en valores*

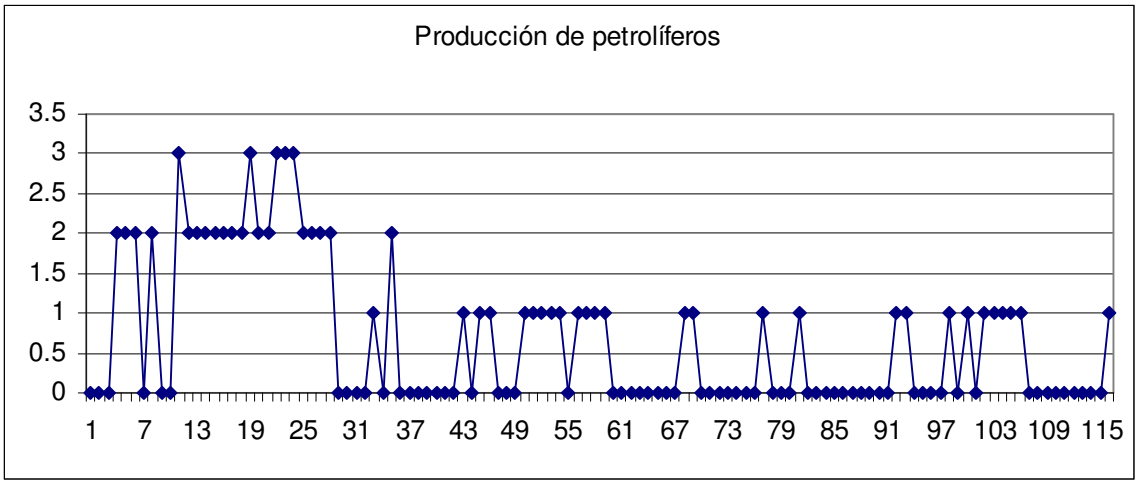


Figura 6.38. Discretización de *Producción de petrolíferos*

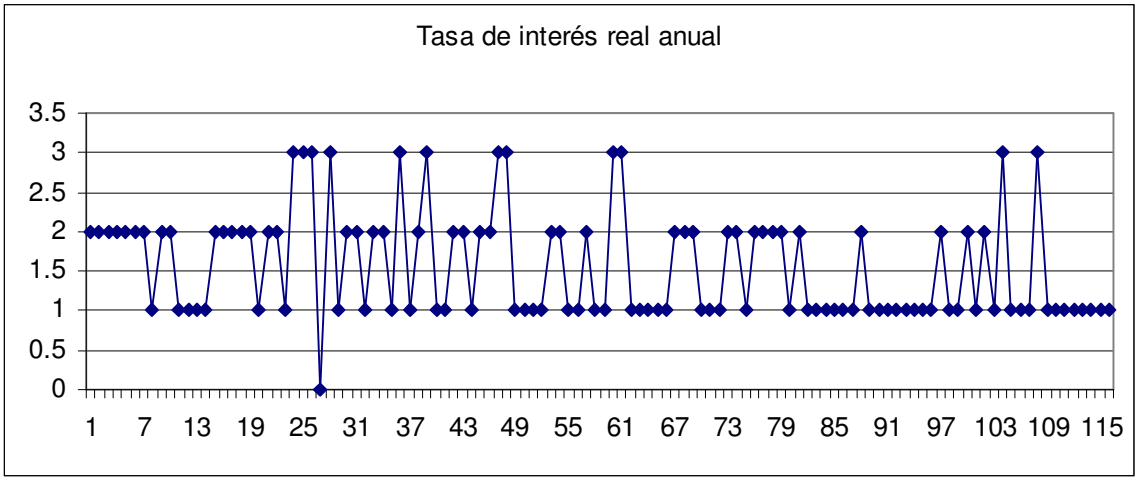


Figura 6.39. Discretización de *Tasa de interés real anual*

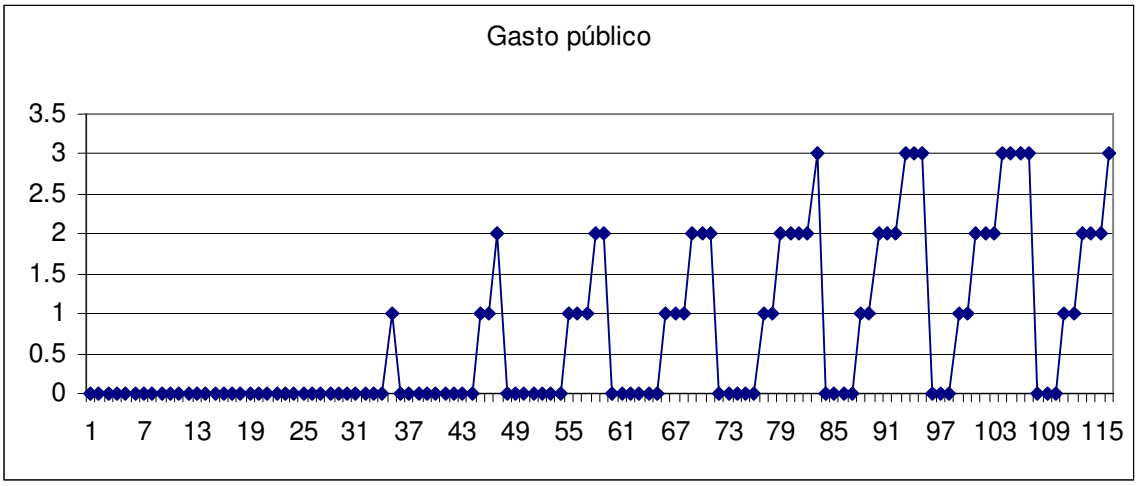


Figura 6.40. Discretización de *Gasto público*

En esta prueba, la discretización de la variable *Actividad económica* permanece igual que en ocasiones anteriores. El caso de la variable *Producción de petrolíferos* es interesante, ya que a pesar de que la serie original presenta valores en las 1500 unidades, una variación pequeña en el parámetro σ provocó que se generaran discretizaciones con variaciones importantes, especialmente en los primeros valores. Así, es evidente que aun el pequeño valor asignado al parámetro σ en pruebas anteriores obligó a tomar en cuenta las variaciones de la serie de tiempo.

La discretización de la variable *Deuda Pública Valores* presenta casi las mismas características que en pruebas anteriores. Aunque la variación del parámetro σ parece significativa, sigue siendo pequeña comparada con los valores que toma la serie original. Por otro lado, la variable *Inflación* presenta una discretización muy distinta, tomando en cuenta únicamente la magnitud de los valores de la serie de tiempo.

La tabla 6.8 muestra los desplazamientos obtenidos al alinear las seis secuencias discretas.

Secuencia discreta	Desplazamiento
Actividad económica	15
Inflación	5
Deuda pública en valores	18
Producción de petrolíferos	1
Tasa de interés real anual	18
Gasto público	3

Tabla 6.8. Desplazamiento de las secuencias discretas

7. EXTRACCIÓN DE REDES BAYESIANAS PREDICTIVAS

Una vez discretizadas y alineadas, las series de tiempo se consideran como un conjunto de casos, de los cuales se puede extraer la estructura de una Red Bayesiana utilizando algún método conocido que tome como entrada una base de datos. Un caso se construye tomando el valor de todas las secuencias en un instante dado, es decir, en una posición determinada, teniendo en cuenta el desplazamiento de cada secuencia.

7.1 Extracción de la estructura de la Red Bayesiana

En esta tesis, para la extracción de la Red Bayesiana a partir del conjunto de casos se utilizó tanto el método Maximum Likelihood Estimation (MLE) como un algoritmo de tres etapas, ambos mencionados en la sección 4.3.2.

El método MLE se implementó generando todas aquellas estructuras de red que no presenten ciclos dirigidos. Para cada una de estas estructuras se obtienen las densidades de probabilidad y se evalúa el likelihood (la probabilidad de los datos dada la Red Bayesiana) para esa estructura. Se elige la estructura que maximice el likelihood.

La detección de ciclos en la estructura se logra haciendo transitiva la relación padre-hijo, de modo que se conozcan todos los descendientes de cada nodo. Así, existirá un ciclo si y solo si un nodo es descendiente de sí mismo.

Para calcular el likelihood de cada estructura se utiliza el siguiente algoritmo:

```
Sea L_total=0
Para cada caso C_i de datos
     Para cada nodo N_j en la Red Bayesiana
          Sea v_ik el valor del caso C_i para el nodo N_k
          L_total = L_total + log(Pr(N_j=v_ij | N_k=v_ik ∀k≠j))
     End For
End For
```

En donde L_{total} es el likelihood resultante. Note que se maneja el logaritmo de la probabilidad para evitar problemas con números demasiado pequeños.

El algoritmo de tres etapas para extracción de Redes Bayesianas consiste en el bosquejo, el engrosamiento y el adelgazamiento de la estructura de la red. Dado un conjunto de *n* series de tiempo alineadas, la primera etapa (bosquejo) crea un grafo no dirigido comparando la información mutua existente entre cualesquiera dos nodos con un umbral definido:

Etapa 1. Bosquejo

1. Inicializar un grafo $G(V, E)$ en donde V es el conjunto de nodos asociados a sendas series de tiempo, y el conjunto de arcos E se inicializa como el conjunto vacío. Crear una lista vacía L.

2. Para cada par de nodos (v_i, v_j) en donde v_i, $v_j \in V$
   Calcular la información mutua $I(v_i, v_j)$.
   Para todos los pares de nodos cuya información mutua sea mayor o igual que un umbral ε
   Ordenarlos de manera descendente por su información mutua y colocar dichos pares en la lista L.
   Crear un apuntador p que señale al primer par de la lista.
3. Obtener los primeros dos pares de nodos en L y eliminarlos de la lista, añadiendo los arcos correspondientes a E. Mover p al próximo par de nodos.
4. Obtener el par de nodos en L que se encuentra en la posición apuntada por p.
   Si no existe un camino de adyacencia entre los dos nodos
   Añadir el arco correspondiente a E y eliminar éste par de nodos de L.
5. Mover el apuntador p al próximo par de nodos y regresar al paso 4 hasta que p apunte al final de L, o G contenga $n-1$ arcos.

Durante la etapa de engrosamiento se añaden los arcos correspondientes a aquellos pares de nodos no adyacentes que no son d-separables.

Etapa 2. Engrosamiento

6. Mover p al primer par de nodos en L.
7. Obtener el par de nodos que se encuentre en la posición p de L.
   Si este par de nodos es d-separable
   Ir al siguiente paso
   Else
   Conectar el par de nodos añadiendo el arco correspondiente a E.
8. Mover p al siguiente par de nodos y regresar al paso 7 hasta que p apunte al final de L.

La última etapa (adelgazamiento) elimina los arcos entre aquellos nodos que son d-separables.

Etapa 3. Adelgazamiento

9. Para cada arco que está en E
   Si existen otros caminos además de dicho arco entre los dos nodos
   Retirar temporalmente el arco de E y verificar si los nodos son d-separables.
   Si no son d-separables
   Eliminar el arco permanentemente.
   End For

10. Orientar los arcos

Por último, se orientan los arcos de acuerdo a las relaciones de dependencia entre ellos y a la forma del grafo. Específicamente, la base para orientar los arcos es la identificación de nodos de aristas convergentes. Algunos pasos siguen un razonamiento similar al utilizado por SGS y PC.

Orientar_arcos

1. Para cualesquiera dos nodos s_1 y s_2 que no estén directamente conectados y que tengan al menos un vecino en común
   Encontrar los vecinos de s_1 y s_2 que estén en el camino de adyacencia entre s_1 y s_2. Ponerlos en dos conjuntos N_1 y N_2.
   End For
2. Encontrar a los vecinos de los nodos en N_1 que estén en los caminos de adyacencia entre s_1 y s_2, y que no pertenezcan a N_1. Ponerlos en el conjunto N_1'.
3. Encontrar a los vecinos de los nodos en N_2 que estén en los caminos de adyacencia entre s_1 y s_2, y que no pertenezcan a N_2. Ponerlos en el conjunto N_2'.
4. Sea C un conjunto de nodos.
   Si $|N_1 \cup N_1'| < |N_2 \cup N_2'|$
   $C = N_1 \cup N_1'$
   Else
   $C = N_2 \cup N_2'$.
5. Sea $a = I(s_1, s_2|C)$.
   Si $a < \varepsilon$
   Ir al paso 8.
6. Sea $C' = C$.
   Para cada $i \in [1, |C|]$
   Sea $C_i = C \backslash \{$el i-ésimo nodo de $C\}$, $a_i = I(s_1, s_2|C_i)$.
   Si $a_i < a + \varepsilon$
   $C' = C' \backslash \{$el i-ésimo nodo de $C\}$
   Sean s_1 y s_2 los padres del i-ésimo nodo de C.
   Si $a_i < \varepsilon$
   Ir al paso 8. (ε es un valor pequeño).
   End For
7. Si $|C'| < |C|$ entonces $C = C'$
   Ir al paso 5.
8. Regresar al paso 1 hasta que todos los pares de nodos hayan sido examinados.
9. Para cada tres nodos a, b y c
   Si a es padre de b, b y c son adyacentes, a y c no son adyacentes y el arco (b, c) no está orientado
   b es el padre de c

```
  End For
10.    Para cualquier arco (a, b) que no esté orientado
           Si hay un camino dirigido de a a b
               a es el padre de b.
  End For
11.    Regresar al paso 9 hasta que no se puedan orientar
  más arcos.
```

Durante la etapa de alineación se asignó un desplazamiento d_i a cada serie de tiempo X_i, creando así un ordenamiento parcial sobre las mismas. De acuerdo a la regla que establece que una consecuencia no puede preceder a su causa, el ordenamiento parcial sirve para orientar algunos de los arcos en el grafo de la Red Bayesiana. Asimismo, el arco dirigido que conecta a los nodos X_i y X_j tiene asociada una diferencia de tiempo $\delta_{ij}=d_i-d_j$, de modo que se espera que los cambios en el nodo X_i se vean reflejados en el nodo X_j después de δ_{ij} unidades de tiempo.

7.2 Recuperación de la serie de tiempo

Para recuperar una serie de tiempo discretizada ${}_D\hat{X}_R$ a partir de una Red Bayesiana, es necesario muestrear el valor de dicha serie sobre la densidad marginal de probabilidad f_R, asociada a ${}_DX_R$. Si $Ancestros({}_DX_R)\neq\varnothing$ o $Descendientes({}_DX_R)\neq\varnothing$ en la Red Bayesiana, entonces f_R depende del valor de un conjunto de k variables $P=\{{}_DX_1, {}_DX_2, ..., {}_DX_k\}$ en la red.

Sea $S\subseteq P$, es obvio que f_R se verá alterada al asignar distintos valores a cada una de las variables contenidas en S. Sean $\{{}_Dx_{i1}, {}_Dx_{i2}, ..., {}_Dx_{iw}\}$ w valores asignados a la variable ${}_DX_i\in S$, la densidad de probabilidad f_R es una función de la posición t ($1\leq t\leq w$), por lo que puede escribirse como $f_R(t)$. Para un valor dado de t, $f_R(t)$ puede expresarse como:

$$p\left({}_DX_R={}_Dx_1 \mid {}_DX_i={}_Dx_{it},\ldots,{}_DX_j={}_Dx_{jt}\right)=p_{1,t}$$
$$p\left({}_DX_R={}_Dx_2 \mid {}_DX_i={}_Dx_{it},\ldots,{}_DX_j={}_Dx_{jt}\right)=p_{2,t}$$
$$\vdots$$
$$p\left({}_DX_R={}_Dx_{r_R} \mid {}_DX_i={}_Dx_{it},\ldots,{}_DX_j={}_Dx_{jt}\right)=p_{r_R,t}$$

en donde r_R es el número de valores que puede tomar la variable ${}_DX_R$. Al muestrear sobre $f_R(t)$ se obtiene un valor estimado ${}_D\hat{x}_R(t)$ para la variable ${}_DX_R$ en la posición t. Si se muestrea para una secuencia de valores de t entre 1 y w, se obtiene una estimación ${}_D\hat{X}_R$ de la serie de tiempo discretizada ${}_DX_R$:

$${}_D\hat{X}_R={}_D\hat{x}_R(1),{}_D\hat{x}_R(2),\ldots,{}_D\hat{x}_R(t)$$

Dado que cada valor ${}_D\hat{x}_R(t)$ ha sido tomado como una muestra aleatoria, la estimación ${}_D\hat{X}_R$ representa un resultado entre varios posibles. En consecuencia, para cada posición t

se pueden tomar u muestras ${}_D x_R^1(t), {}_D x_R^2(t), \mathrm{K}, {}_D x_R^u(t)$, obteniéndose u posibles estimaciones para cada posición. Al aplicar el mismo razonamiento para cada posición t se obtienen u estimaciones para la serie de tiempo discretizada:

$$
\begin{aligned}
{}_D \hat{X}_R^1 &= \left({}_D x_R^1(1), {}_D x_R^1(2), \mathrm{K}, {}_D x_R^1(w) \right) \\
{}_D \hat{X}_R^2 &= \left({}_D x_R^2(1), {}_D x_R^2(2), \mathrm{K}, {}_D x_R^2(w) \right) \\
&\vdots \\
{}_D \hat{X}_R^u &= \left({}_D x_R^u(1), {}_D x_R^u(2), \mathrm{K}, {}_D x_R^u(w) \right)
\end{aligned}
$$

Al recuperar la serie continua $\hat{X}_R^i$ a partir de los valores discretos de la estimación ${}_D \hat{X}_R^i$, se obtiene una trayectoria posible para la serie de tiempo. Es decir, debido a que durante la discretización de la serie de tiempo se pudo tomar en cuenta la variación de cada punto respecto al anterior, la variación en un valor discreto ${}_D \hat{x}_R^i(t)$ podría afectar la recuperación de los valores continuos posteriores $\hat{x}_R^i(t), \hat{x}_R^i(t+1), \mathrm{K}, \hat{x}_R^i(w)$.

La distribución de los valores continuos recuperados en un punto t, $\hat{x}_R^1(t), \hat{x}_R^2(t), \mathrm{K}, \hat{x}_R^u(t)$ no puede asumirse como normal debido a que se desconocen, entre otros, la naturaleza de la serie de tiempo y el valor de σ utilizado durante la discretización. Este hecho descalifica el uso de la media de los valores $\hat{x}_R^1(t), \hat{x}_R^2(t), \mathrm{K}, \hat{x}_R^u(t)$ para la estimación de un valor final $\hat{x}_R(t)$ para la posición t.

Inicialmente, se pensó en utilizar la mediana de los valores $\hat{x}_R^1(t), \hat{x}_R^2(t), \mathrm{K}, \hat{x}_R^u(t)$ como el valor $\hat{x}_R(t)$ para cada posición debido a su robustez ante la presencia de valores extremos (outliers). No obstante, se observó que es necesario tomar en cuenta la densidad de las trayectorias para los valores dados, es decir, tomar un valor cercano a aquel valor en torno al cual se agrupan la mayoría de los valores $\hat{x}_R^i(t)$.

Por ejemplo, suponga que se tienen nueve valores: $\hat{x}_R^1(t) = 2$, $\hat{x}_R^2(t) = 2.3$, $\hat{x}_R^3(t) = 2$, $\hat{x}_R^4(t) = 3$, $\hat{x}_R^5(t) = 6$, $\hat{x}_R^6(t) = 5.2$, $\hat{x}_R^7(t) = 10.4$, $\hat{x}_R^8(t) = 7.3$, $\hat{x}_R^9(t) = 2$,. Al ordenarlos se obtendría la secuencia 2, 2, 2, 2.3, 3, 5.2, 6, 7.3 y 10.4. Es claro que existe una agrupación de valores en torno a 2, y que los demás, aunque son mayoría, están dispersos y por lo tanto no son tan representativos. La mediana de este conjunto de valores es 3, lo que muestra lo inapropiada que puede resultar esta medida.

Pese al ejemplo anterior, la mediana resultaría apropiada en el caso de valores cuya distribución fuera simétrica, especialmente en el caso de una distribución uniforme. En otro

caso, sería preferible tomar un valor cercano a aquel en torno al cual la distancia entre los valores recuperados fuera menor, para lo cual se requeriría dar algún peso a ciertos valores en base a la cercanía con sus vecinos.

Dado que es más fácil calcular una medida de distancia que una medida de cercanía entre valores, y utilizando una idea de dualidad, se ha optado por elegir el valor que se encuentre simétrico a aquel en el cual la acumulación de distancias iguale o supere la mitad de la suma total de distancias. Es decir, sea $\hat{x}_R^{1_s}(t), \hat{x}_R^{2_s}(t), \mathrm{K}, \hat{x}_R^{u_s}(t)$ una secuencia ordenada de los valores $\hat{x}_R^1(t), \hat{x}_R^2(t), \mathrm{K}, \hat{x}_R^u(t)$, sea la distancia entre valores $d_i = \hat{x}_R^{i_s+1}(t) - \hat{x}_R^{i_s}(t)$, y sea la suma de las distancias $d_T = \sum_{i=1}^{u-1} d_i$ en donde *u* es el número de valores estimados, se elige $\hat{x}_R(t) = \hat{x}_R^{(u-k)_s}(t)$, en donde *k* es el menor entero tal que $\sum_{i=1}^{k} d_i \geq \frac{d_T}{2}$.

Retomando el ejemplo anterior, la secuencia ordenada sería $\hat{x}_R^{1_s}(t) = 2$, $\hat{x}_R^{2_s}(t) = 2$, $\hat{x}_R^{3_s}(t) = 2$, $\hat{x}_R^{4_s}(t) = 2.3$, $\hat{x}_R^{5_s}(t) = 3$, $\hat{x}_R^{6_s}(t) = 5.2$, $\hat{x}_R^{7_s}(t) = 6$, $\hat{x}_R^{8_s}(t) = 7.3$, $\hat{x}_R^{9_s}(t) = 10.4$, mientras que las distancias serían $d_1 = 0$, $d_2 = 0$, $d_3 = 0.3$, $d_4 = 0.7$, $d_5 = 1.2$, $d_6 = 0.8$, $d_7 = 1.3$ y $d_8 = 3.1$, de donde se calcula $d_T = 7.4$. Como $\frac{d_T}{2} = 3.7$ y $\sum_{i=1}^{7} d_i = 4.3$, entonces *k*=7, *u-k* = 9-7 =2, y el valor elegido es $\hat{x}_R(t) = \hat{x}_R^{2_s}(t) = 2$.

Es fácil notar que, si la distribución de valores es simétrica, el valor elegido por este método será igual a la mediana.

Una vez calculado el valor $\hat{x}_R(t)$ para el punto *t*, se define el error $e_R^i(t) = \left| \hat{x}_R^i(t) - \hat{x}_R(t) \right|$ como la distancia entre el valor de la serie $\hat{X}_R^i$ para la posición *t* y el valor elegido para dicha posición. Al calcular la mediana de los valores $e_R^1(t), e_R^2(t), \mathrm{K}, e_R^u(t)$ se obtiene una estimación $e_R(t)$ del error de $\hat{x}_R(t)$ respecto al valor real $x_R(t)$ de la serie de tiempo para cada posición *t* [Medina & Figueroa, 2003].

7.3 Pruebas y resultados

En esta sección se retoman los resultados mostrados en la sección 6.3, generándose para cada prueba los modelos a partir de las secuencias discretas alienadas. En cada prueba se obtiene, cuando es posible, tanto el modelo generado por MLE como el obtenido a partir del algoritmo de tres etapas. Para cada red generada se recupera al menos una serie de tiempo y se compara con la serie original. En los casos en los que es posible predecir, se muestra la predicción a corto plazo utilizando cada una de las estructuras generadas.

Durante la extracción de Redes Bayesianas mediante el algoritmo de tres etapas, se utilizó un valor de ε igual al 20% de la información mutua máxima entre los pares de series de tiempo (ver sección 7.1).

Los valores obtenidos para σ afectan la manera en que se recupera una serie de tiempo a partir de la red. Un valor relativamente pequeño genera secuencias discretas tales que la recuperación la serie de tiempo resulta estable, es decir, el error de recuperación no crece conforme transcurre el tiempo. En contraste, un valor mayor para σ genera recuperaciones más precisas, pero cuyo error aumenta conforme transcurre el tiempo.

La estructura obtenida para un conjunto de secuencias discretas depende en gran medida del método utilizado para su obtención. El método Maximum Likelihood Estimation tiende a generar grafos completos, mientras que el método de tres etapas basado en información mutua genera grafos con pocas aristas.

7.3.1 Prueba 1

Las secuencias discretas alineadas obtenidas en la sección 6.3.1 son utilizadas para extraer la Red Bayesiana. Para esto se puede utilizar el método MLE o el algoritmo de tres etapas descrito en la sección 7.1. La figura 7.1 muestra la estructura de la Red Bayesiana extraída utilizando el método MLE.

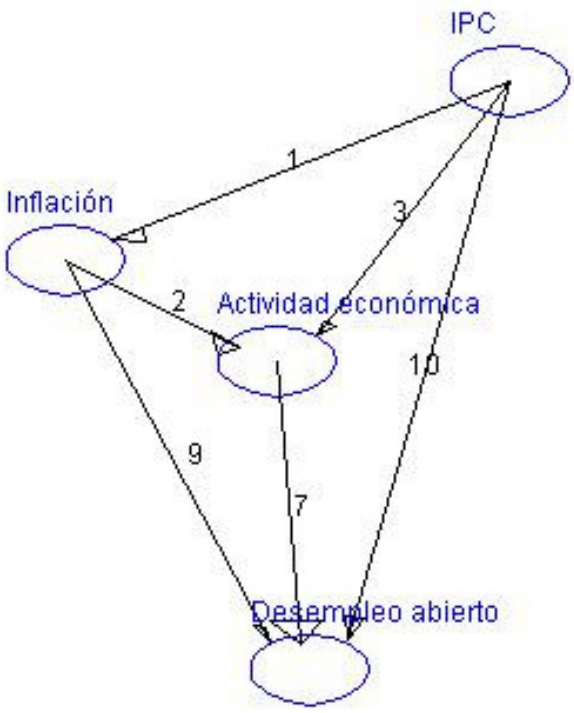


Figura 7.1. Estructura de la Red Bayesiana obtenida utilizando MLE

Observe que la estructura obtenida no corresponde con la observación realizada en la sección 6.3.1 acerca de la manera en que afectó la crisis del “Efecto Tequila” a la inflación

y al desempleo. Sin embargo, al observar las secuencias discretas es notorio que esta sincronización es muy singular, y que para las secuencias completas existen otras alineaciones que producen una mejor coincidencia.

La figura 7.2 muestra la Red Bayesiana obtenida aplicando el algoritmo de extracción de tres etapas al conjunto de secuencias discretas con los mismos desplazamientos. Aquí se observa a la variable *IPC* como el padre común a las otras tres variables. Dada esta relación, la red muestra que la relación entre las demás variables puede ser obtenida por medio de la variable padre.

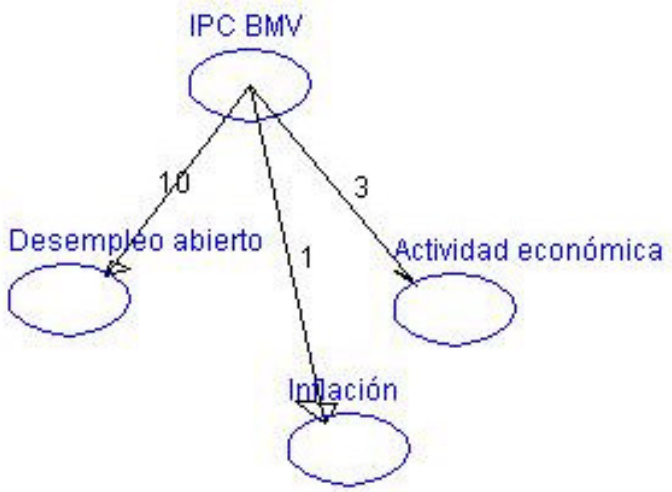


Figura 7.2. Estructura de la Red Bayesiana obtenida utilizando el algoritmo de tres etapas

A partir de las Redes Bayesianas mostradas en las figuras 7.1 y 7.2, es posible recuperar alguna serie de tiempo con el objetivo de evaluar la relevancia de las relaciones mostradas en cada red sobre dicha serie. En este caso, se ha elegido recuperar la serie *Desempleo abierto*, utilizando los valores de aquellas series que aparecen como padres en cada red.

Durante la recuperación de una serie de tiempo, se obtiene una estimación del error absoluto para cada punto. Este error puede ser comparado con el error real de la serie recuperada respecto a la serie original.

La figura 7.3 muestra la serie de tiempo *Desempleo abierto* y su recuperación a partir de la Red Bayesiana de la figura 7.1. La recuperación se efectuó asignando los valores originales a las series *Inflación*, *IPC* y *Actividad económica*. En la gráfica resultante se observa como la serie recuperada sigue a la original durante la subida de la misma después del punto 25. Es importante tener en mente las series de tiempo originales a partir de las cuales se logró esta recuperación.

La figura 7.4 muestra el error absoluto y su estimación, obtenidos a partir de esta recuperación. Como se podría esperar, la estimación del error es más acertada en aquellos puntos en los que, durante la recuperación, se eligió algún valor que difería para otras trayectorias, como es el caso de los puntos 85 al 96.

En este caso, la gráfica de estimación del error puede dar una idea, aunque poco precisa, acerca de que partes de la recuperación son más o menos confiables. Sin embargo, la estimación del error suele ser mucho más útil cuando la serie recuperada se ha discretizado con un valor de σ comparable con su intervalo dinámico.

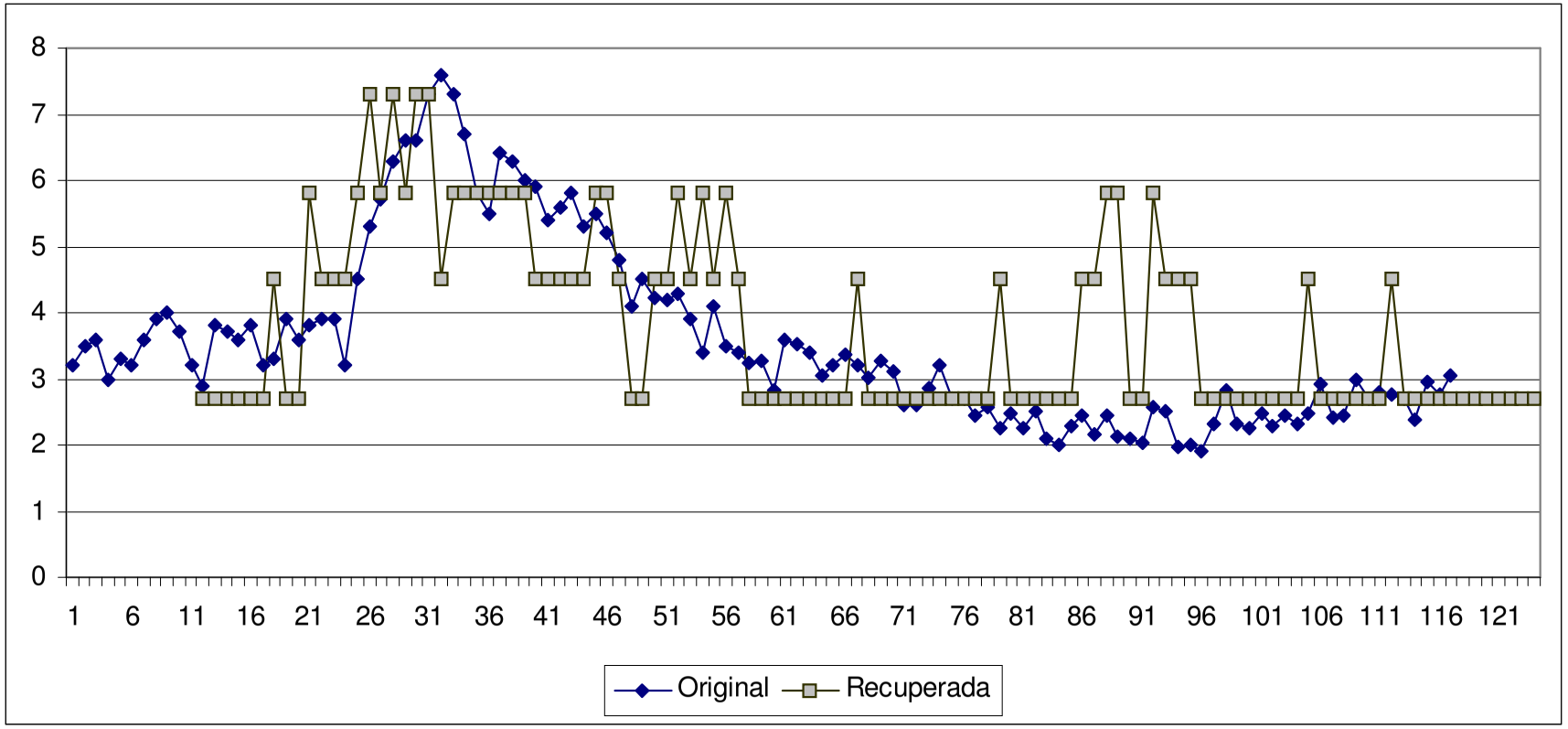


Figura 7.3. Serie de tiempo *Desempleo abierto*, original y recuperada a partir de la red obtenida utilizando MLE (figura 7.1)

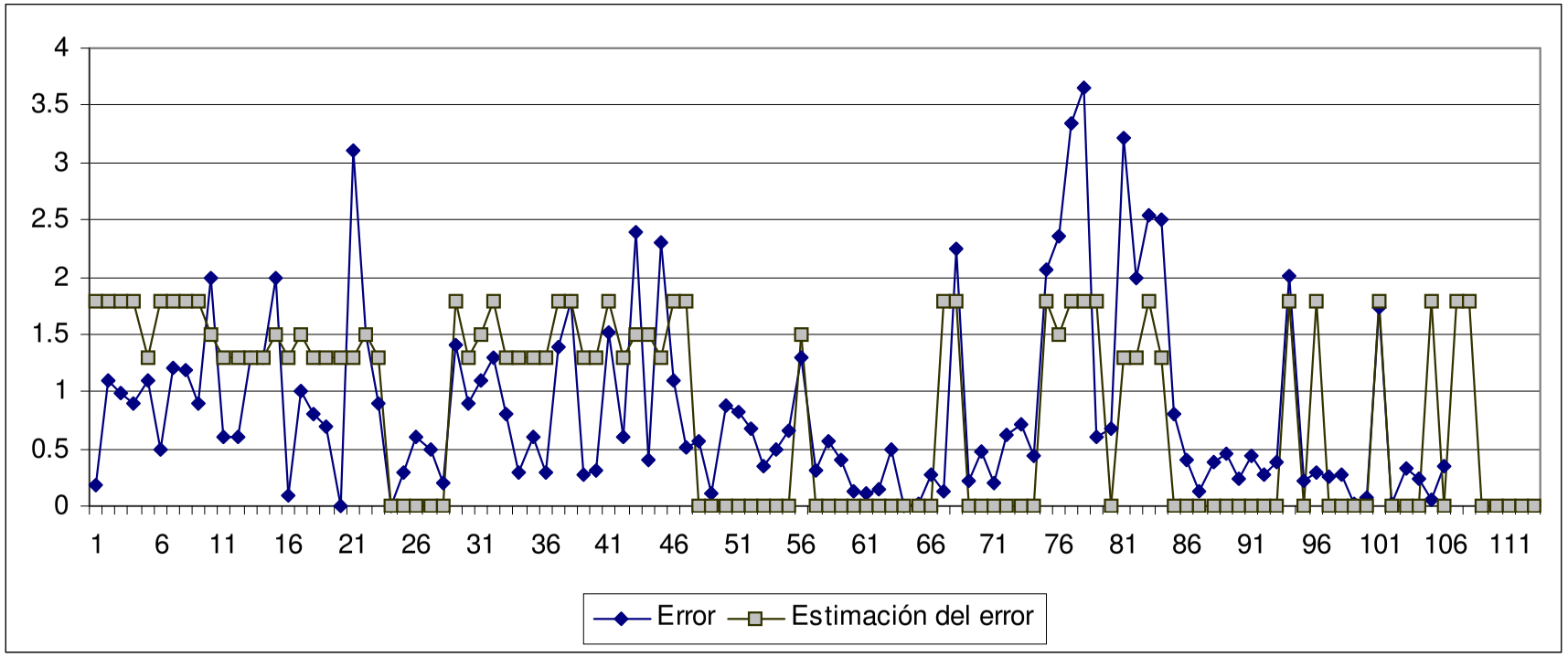


Figura 7.4. Error y estimación del error de la recuperación de la serie *Desempleo abierto*, mostrada en la figura 7.3

La figura 7.5 muestra los siete valores siguientes al último mostrado en la serie original de la figura 7.3, es decir, aquellos correspondientes a los meses que se encuentran entre Octubre del 2002 y Abril del 2003. Dado que estos valores no fueron introducidos durante la etapa de aprendizaje de la red, se pueden ver como una predicción de la misma. La figura 7.6 muestra el error de predicción y su estimación previa.

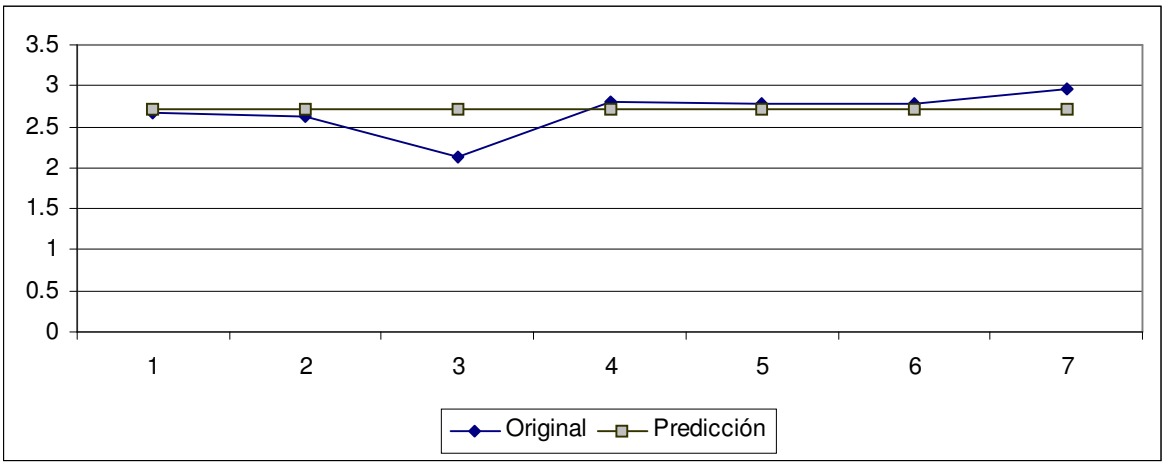


Figura 7.5. Predicción de la serie *Desempleo abierto* a partir de la red generada con MLE

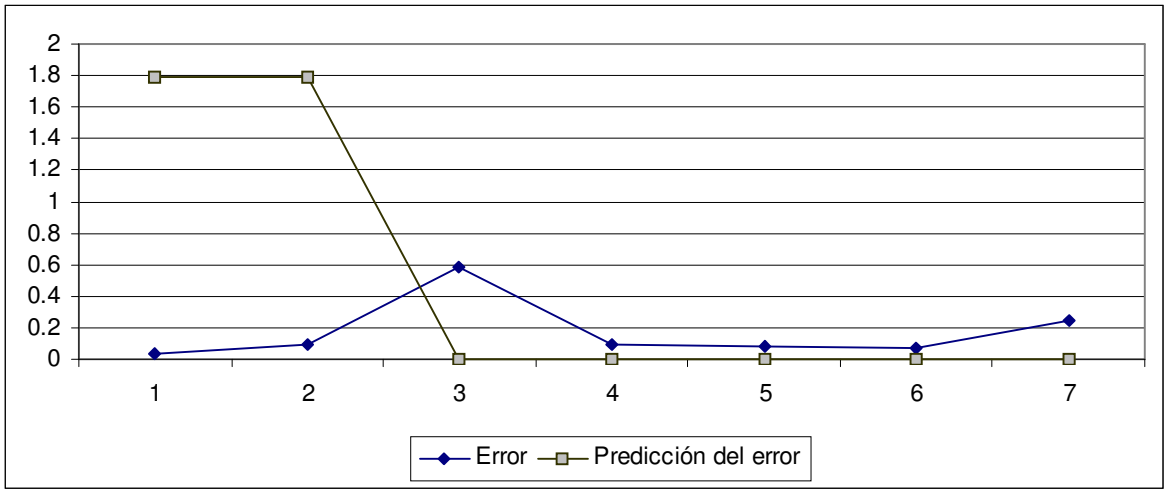


Figura 7.6. Error de predicción de la serie *Desempleo abierto* a partir de la red generada con MLE

La figura 7.7 muestra la serie de tiempo *Desempleo abierto* y su recuperación al asignar los valores originales a la serie *IPC* en la red de la figura 7.2. En este caso, la recuperación muestra mucho menos detalle que en el anterior, como se esperaría de una estructura de red más sencilla. También se observa que la serie recuperada solamente contiene tres valores distintos, lo cual se entiende debido a que el proceso de recuperación obtiene los valores a utilizar mediante un muestreo aleatorio.

La figura 7.8 muestra el error absoluto y su estimación, la cual se encuentra en el mismo caso que en la red anterior.

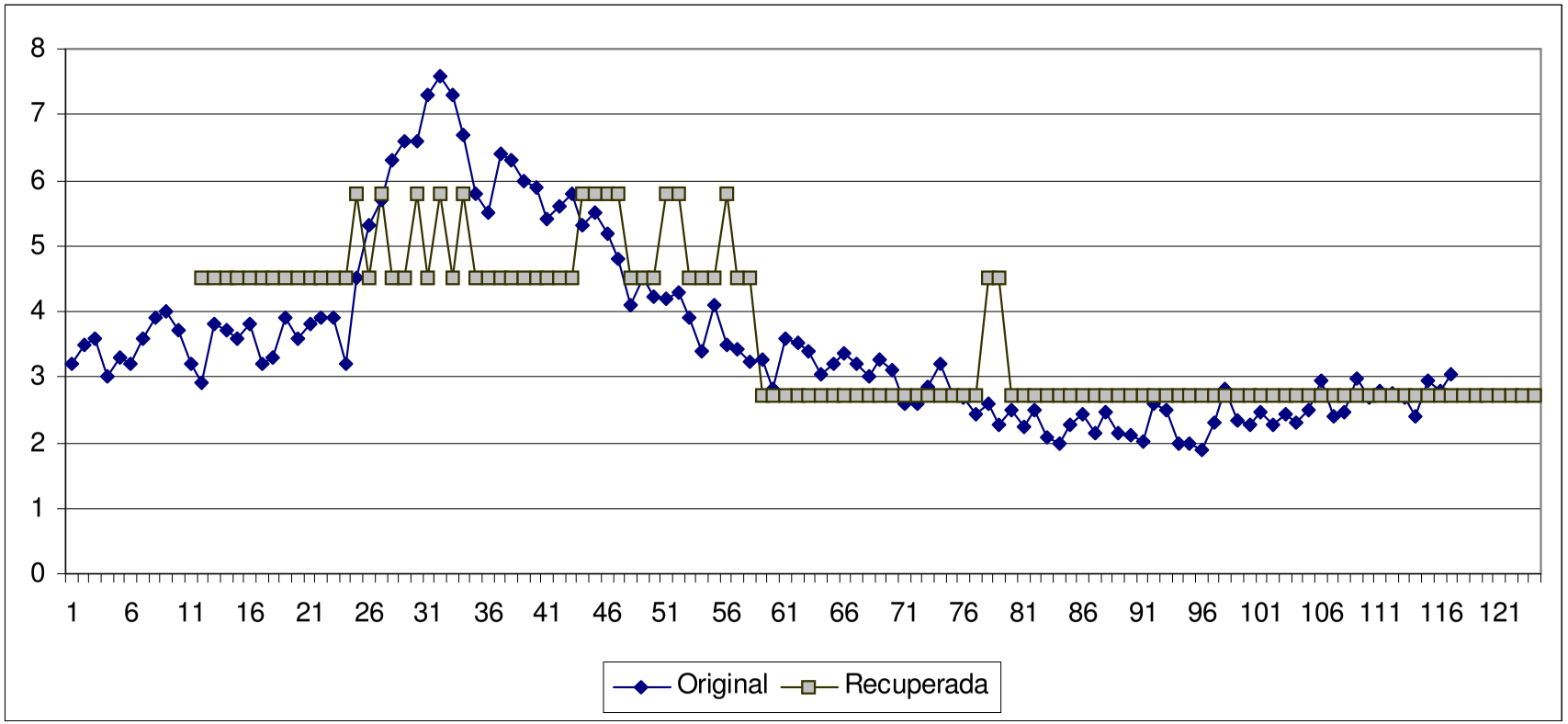


Figura 7.7. Serie de tiempo *Desempleo abierto*, original y recuperada a partir de la red obtenida utilizando el algoritmo de tres etapas (figura 7.2)

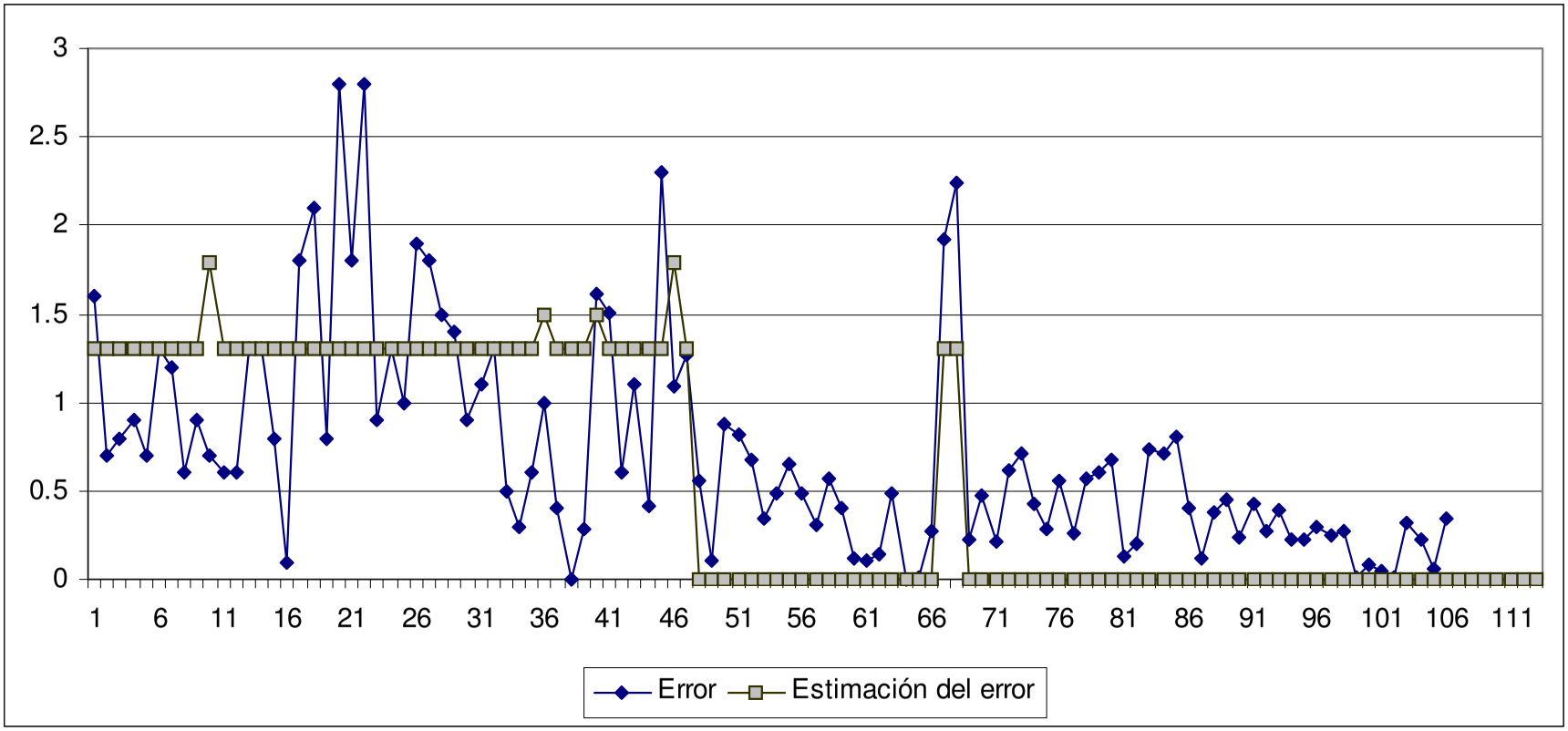


Figura 7.8. Error y estimación del error de la recuperación de la serie *Desempleo abierto*, mostrada en la figura 7.7

La figura 7.9 muestra la predicción de diez valores de la serie *Desempleo abierto*, posteriores a aquellos mostrados en la serie original de la figura 7.7. La figura 7.10 muestra el error y su estimación previa.

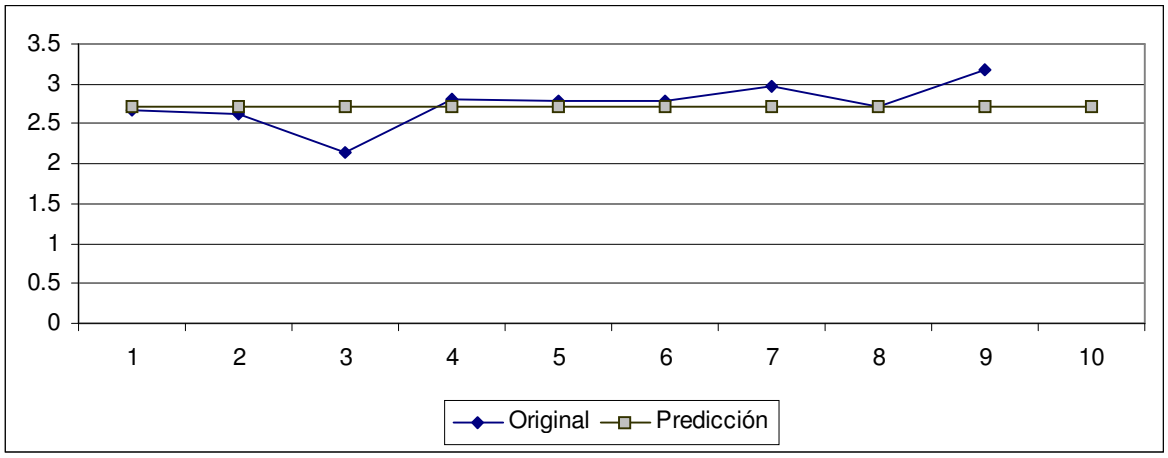


Figura 7.9. Predicción de la serie *Desempleo abierto* a partir de la red generada con el algoritmo de tres etapas

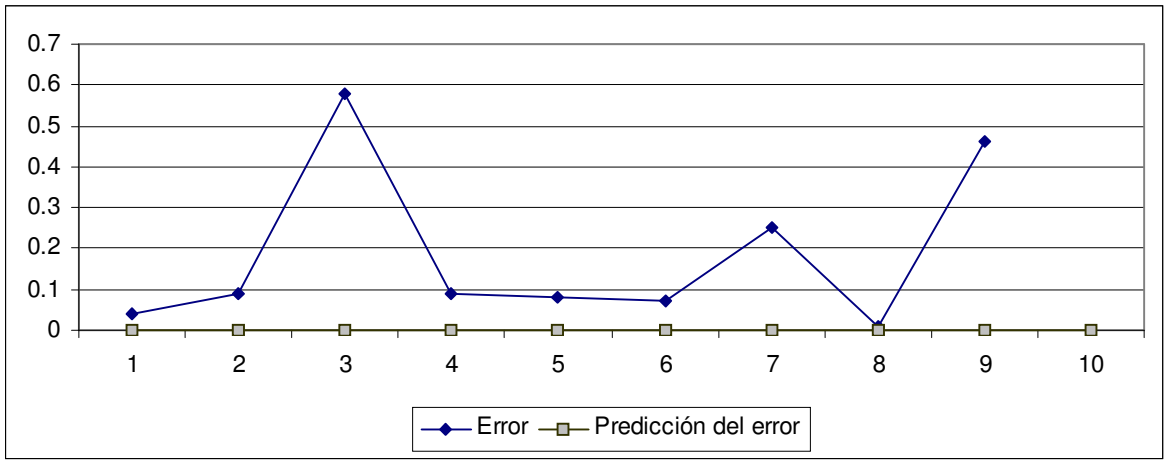


Figura 7.10. Error de predicción de la serie *Desempleo abierto* a partir de la red generada con el algoritmo de tres etapas

7.3.2 Prueba 2

Para esta prueba se utilizan las series de tiempo y secuencias discretas obtenidas en la sección 6.3.2. Utilizando el método MLE se obtiene la Red Bayesiana mostrada en la figura 7.11. Al igual que en la prueba anterior, esta red no presenta las relaciones que se esperarían por la reacción de las variables a la crisis de 1995, pero observando de nuevo las secuencias discretas se confirma que tales coincidencias no son constantes en el tiempo.

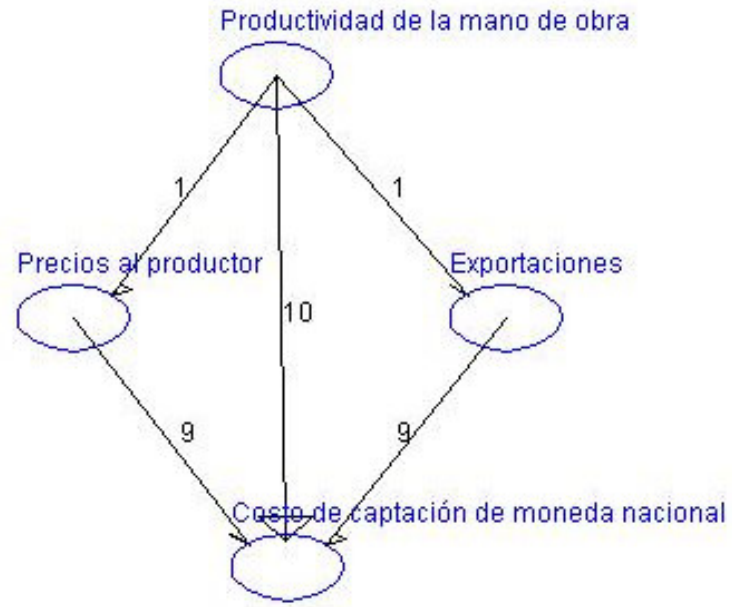


Figura 7.11. Estructura de la Red Bayesiana obtenida utilizando MLE

La figura 7.12 muestra la Red Bayesiana obtenida utilizando el algoritmo de extracción de tres etapas. De nuevo y como es de esperarse, se obtiene una estructura más sencilla, en la que las relaciones entre algunas variables están representadas por su dependencia de un solo padre.

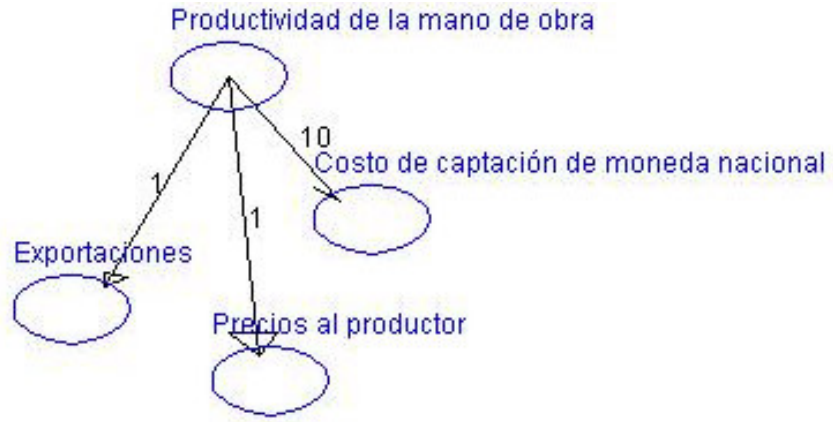


Figura 7.12. Estructura de la Red Bayesiana obtenida utilizando el algoritmo de tres etapas

La figura 7.13 muestra la recuperación de la serie de tiempo *Costo de captación de moneda nacional* a partir de la red mostrada en la figura 7.11 (obtenida utilizando MLE), dados los valores de las series *Productividad de la mano de obra*, *Precios al productor* y *Exportaciones*. Al igual que en la recuperación de la serie *Desempleo abierto* (sección 7.3.1), después del punto 24 los valores recuperados presentan un comportamiento difícil de describir a partir de las series de tiempo originales, las cuales fueron utilizadas como datos de entrada para la recuperación.

La figura 7.14 muestra el error de esta recuperación y su estimación. Dado que la serie *Costo de captación de moneda nacional* fue discretizada utilizando un valor de $\sigma = 0$, esta gráfica contiene poca información.

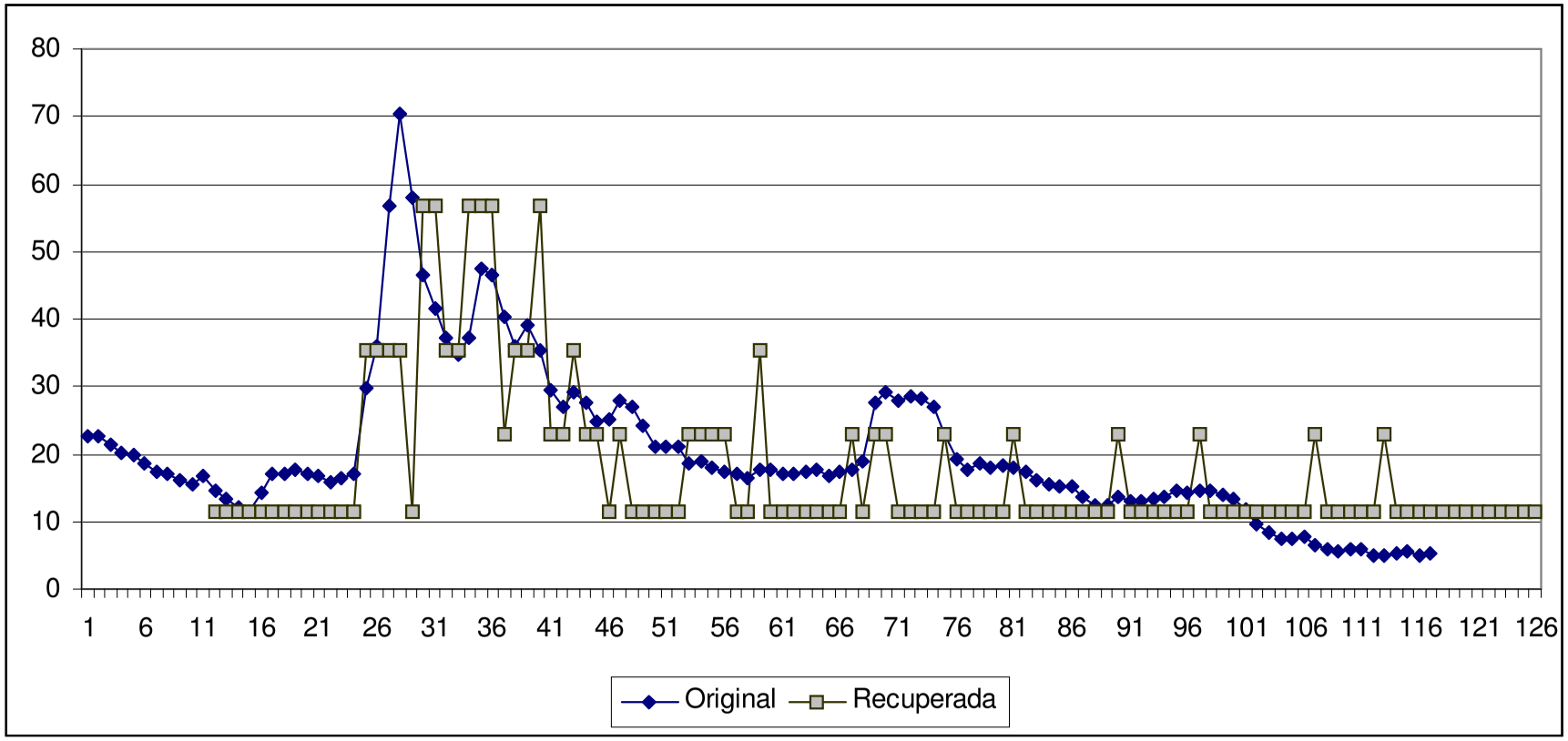


Figura 7.13. Serie de tiempo *Costo de captación de moneda nacional* original y recuperada a partir de la red obtenida con MLE (figura 7.11)

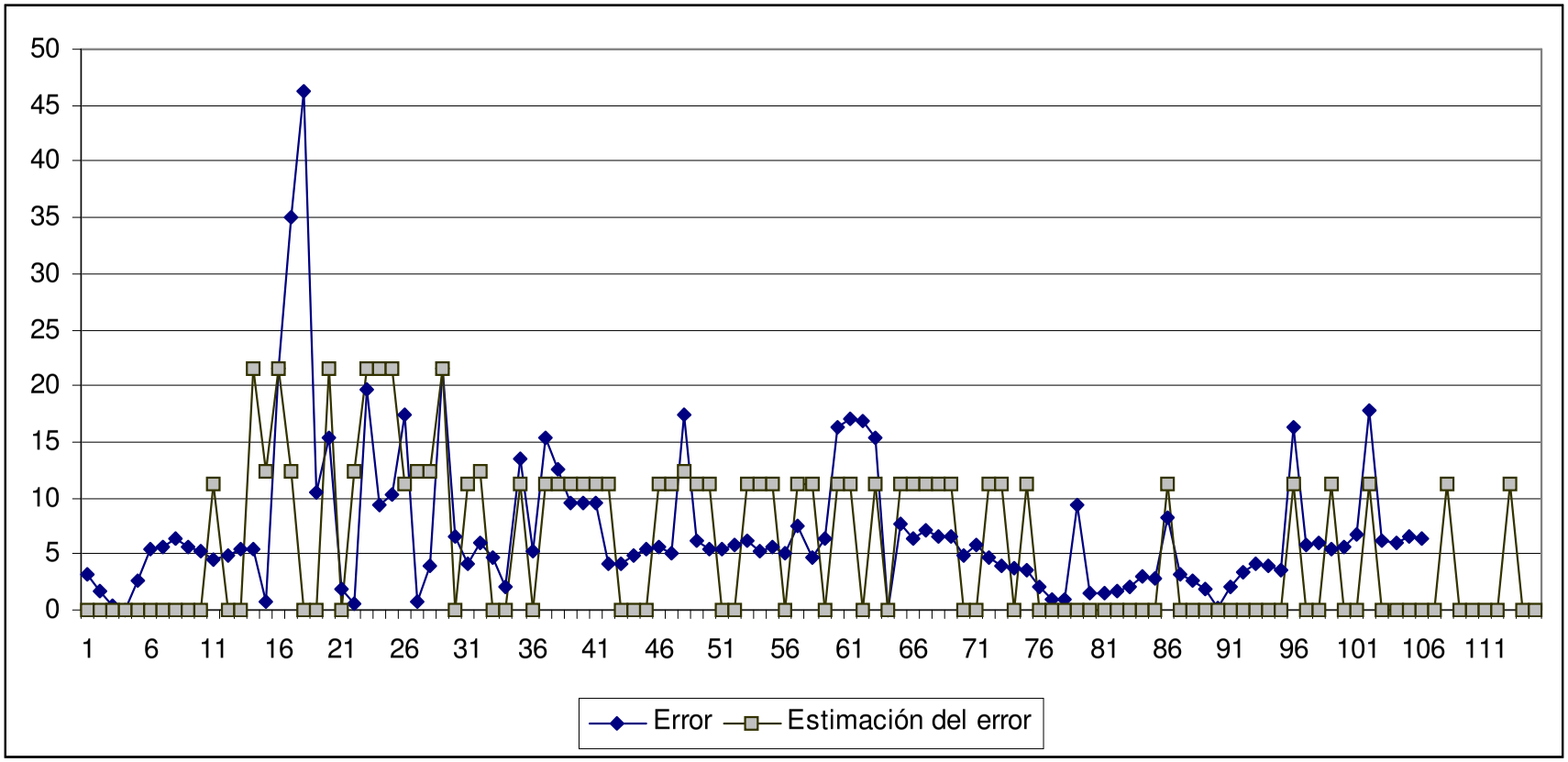


Figura 7.14. Error y estimación del error de la recuperación de la serie *Costo de captación de moneda nacional*, mostrada en la figura 7.13

La figura 7.15 muestra la predicción de los nueve valores siguientes a los últimos mostrados en la serie original de la figura 7.13. Note que el error en el primer valor de la predicción, originado por haber utilizado una continuación de la recuperación de la variable, es el que ocasiona el mayor impacto en el resto de la gráfica. La figura 7.16 muestra el error de predicción y su estimación previa.

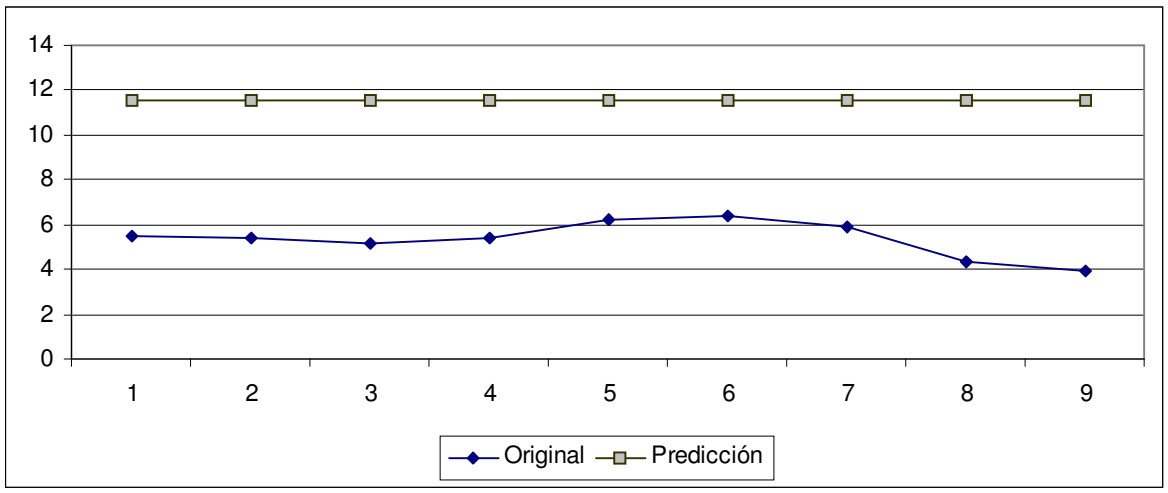


Figura 7.15. Predicción de la serie *Costo de captación de moneda nacional* a partir de la red generada con el algoritmo MLE

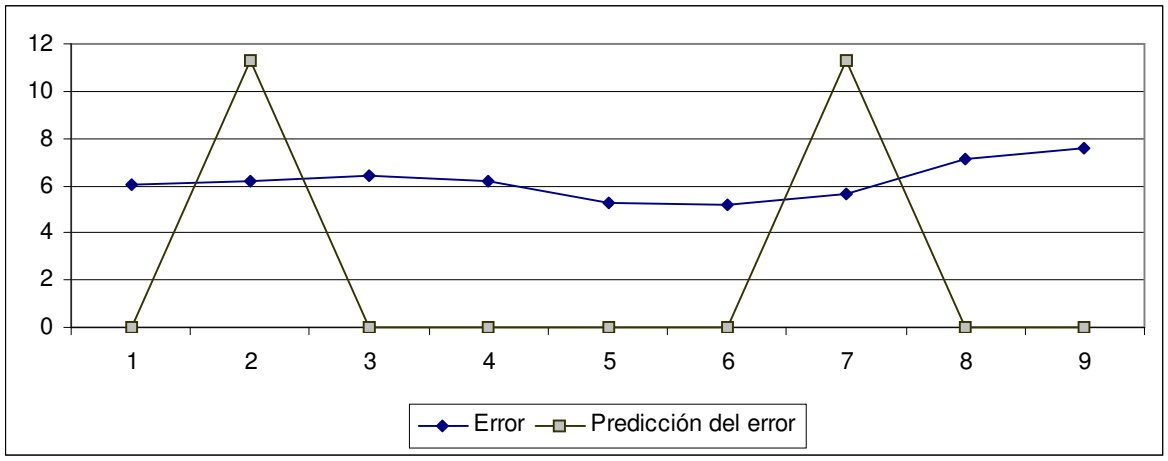


Figura 7.16. Error de predicción de la serie *Costo de captación de moneda nacional* a partir de la red generada con el algoritmo MLE

En la red de la figura 7.12, obtenida utilizando el algoritmo de tres etapas, se observa que la variable *Productividad de la mano de obra* afecta directamente a *Costo de captación de moneda nacional*. La recuperación de esta última variable se muestra en las figuras 7.17 y su error en la figura 7.18.

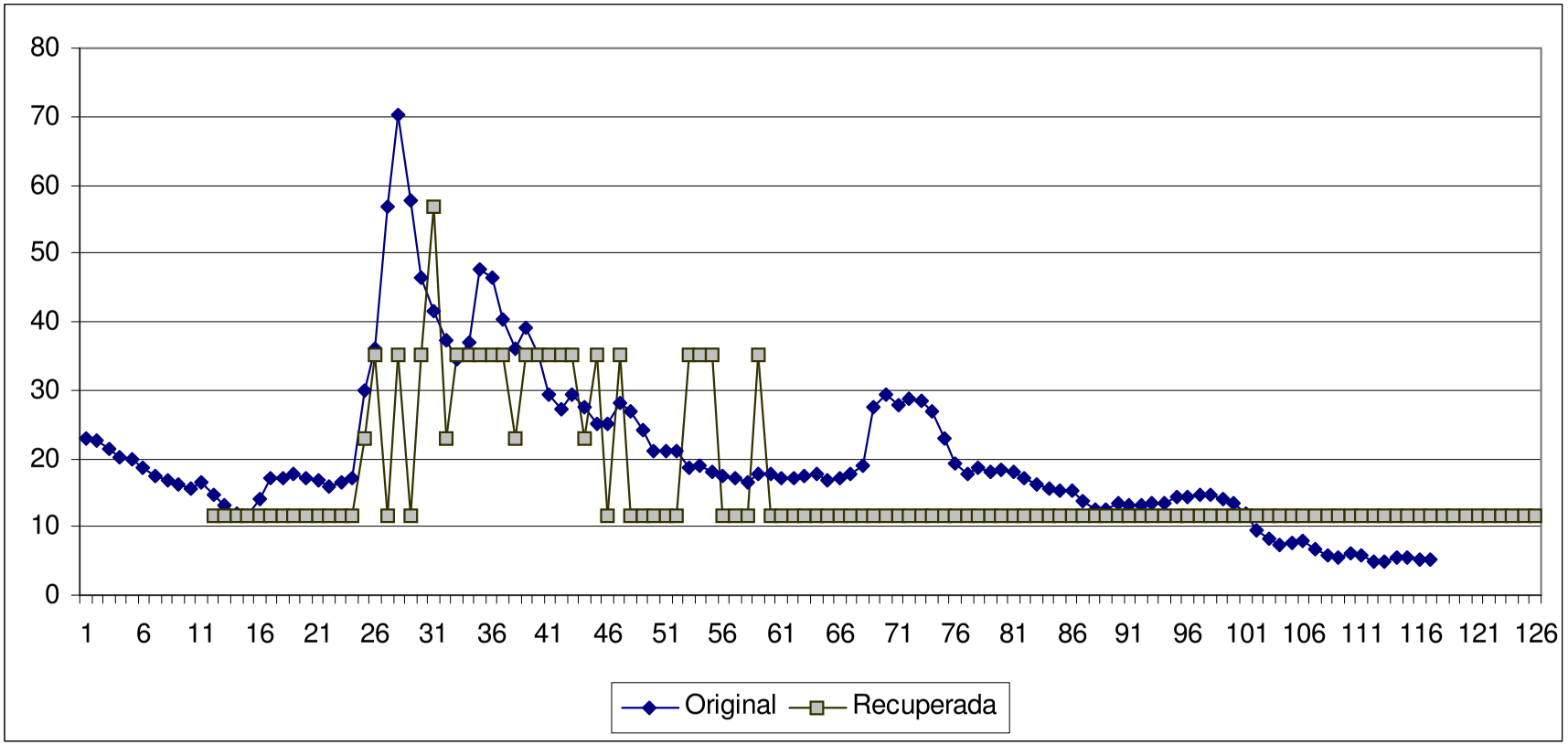


Figura 7.17. Serie de tiempo *Costo de captación de moneda nacional*, original y recuperada, a partir de la red obtenida utilizando el algoritmo de tres etapas (figura 7.12)

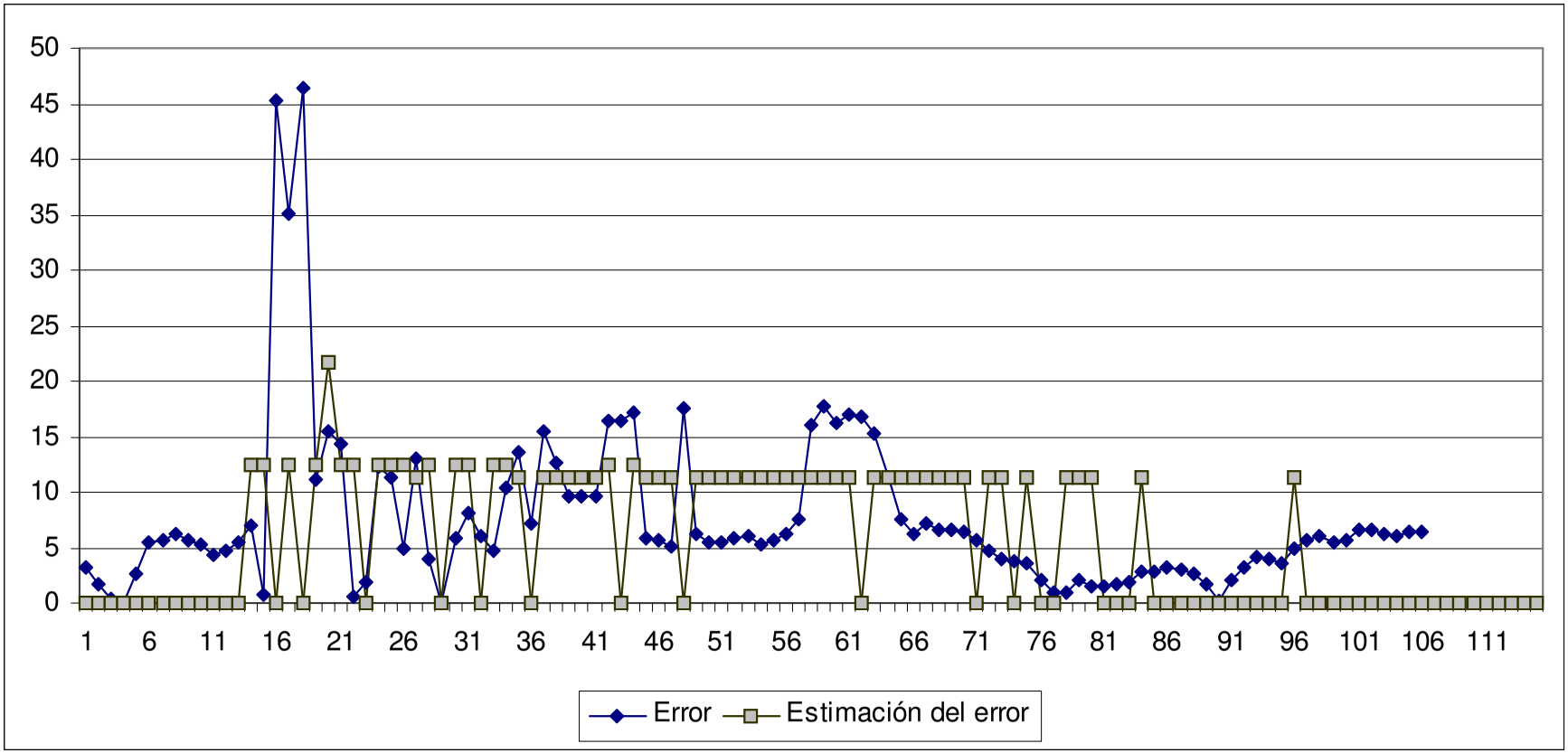


Figura 7.18. Error y estimación del error de la recuperación de la serie *Costo de captación de moneda nacional*, mostrada en la figura 7.17

La figura 7.19 muestra los diez valores predichos en la figura 7.17. La figura 7.20 muestra el error y su estimación previa.

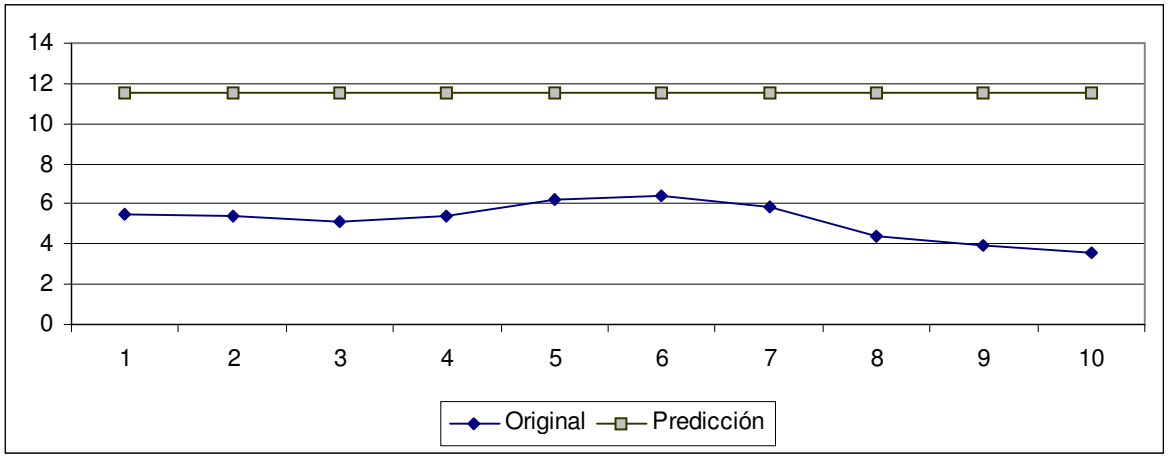


Figura 7.19. Predicción de la serie *Costo de captación de moneda nacional* a partir de la red generada con el algoritmo de tres etapas

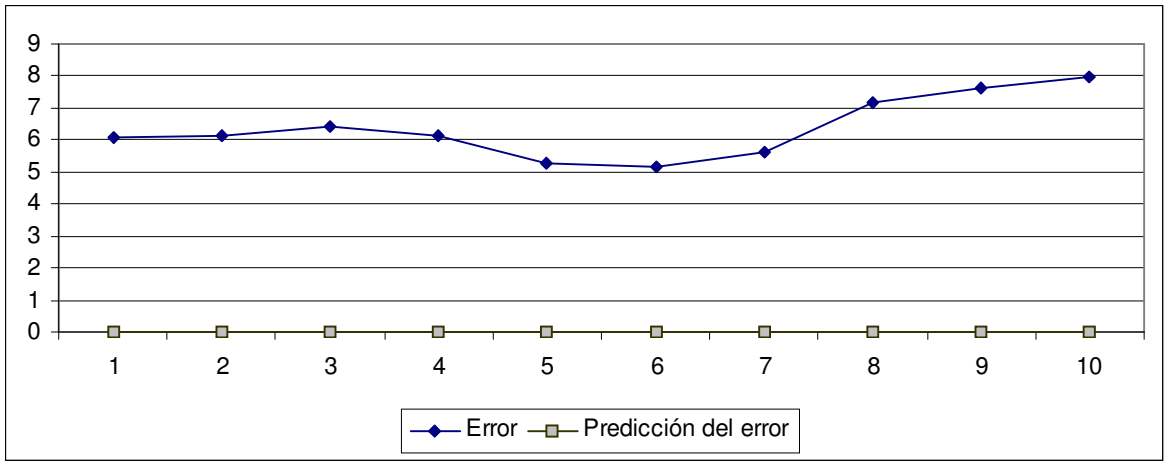


Figura 7.20. Error de predicción de la serie *Costo de captación de moneda nacional* a partir de la red generada con el algoritmo de tres etapas

7.3.3 Prueba 3

En esta prueba se utilizaron los datos mostrados en la sección 6.3.3. Para extraer la estructura de la Red Bayesiana se utilizó únicamente el algoritmo de tres etapas, debido a que la utilización del método MLE hubiera requerido la generación de todas las posibles estructuras y, dado que el número de nodos es relativamente grande (10 nodos), el número de posibles estructuras es enorme. La figura 7.21 muestra la red obtenida.

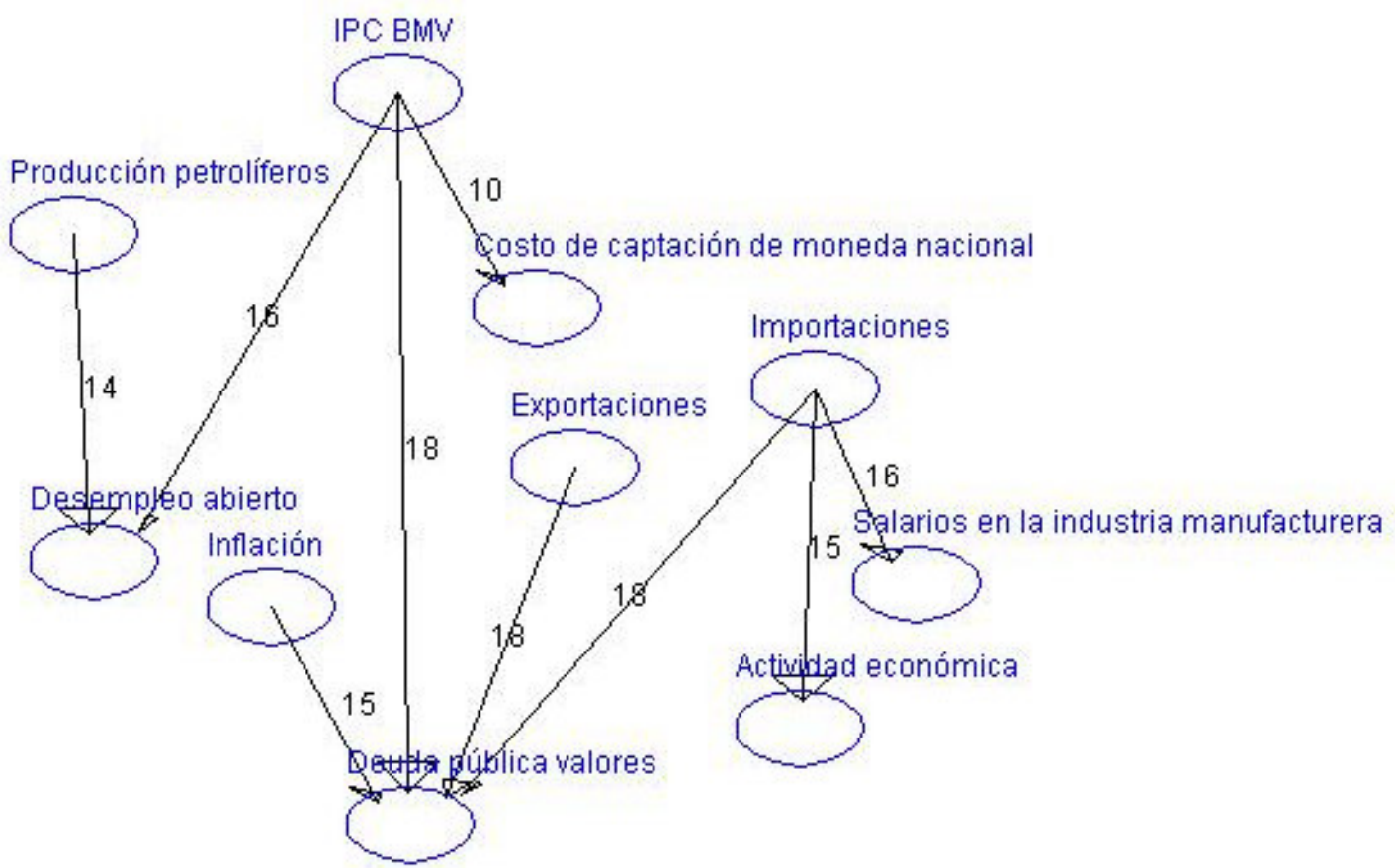


Figura 7.21. Red Bayesiana extraída utilizando el algoritmo de tres etapas

A partir de esta red se han recuperado las series de tiempo *Actividad económica*, *Deuda pública en valores* y *Salarios en la industria manufacturera*.

La recuperación de la serie *Actividad económica* a partir de la serie *Importaciones* se muestra en la figura 7.22. Aun cuando la relación entre estas dos variables no parece obvia, la recuperación obtenida es similar a la serie original. El error de recuperación se muestra en la figura 7.23. La figura 7.24 muestra los quince valores predichos en esta recuperación. La figura 7.25 muestra el error de predicción y su estimación previa.

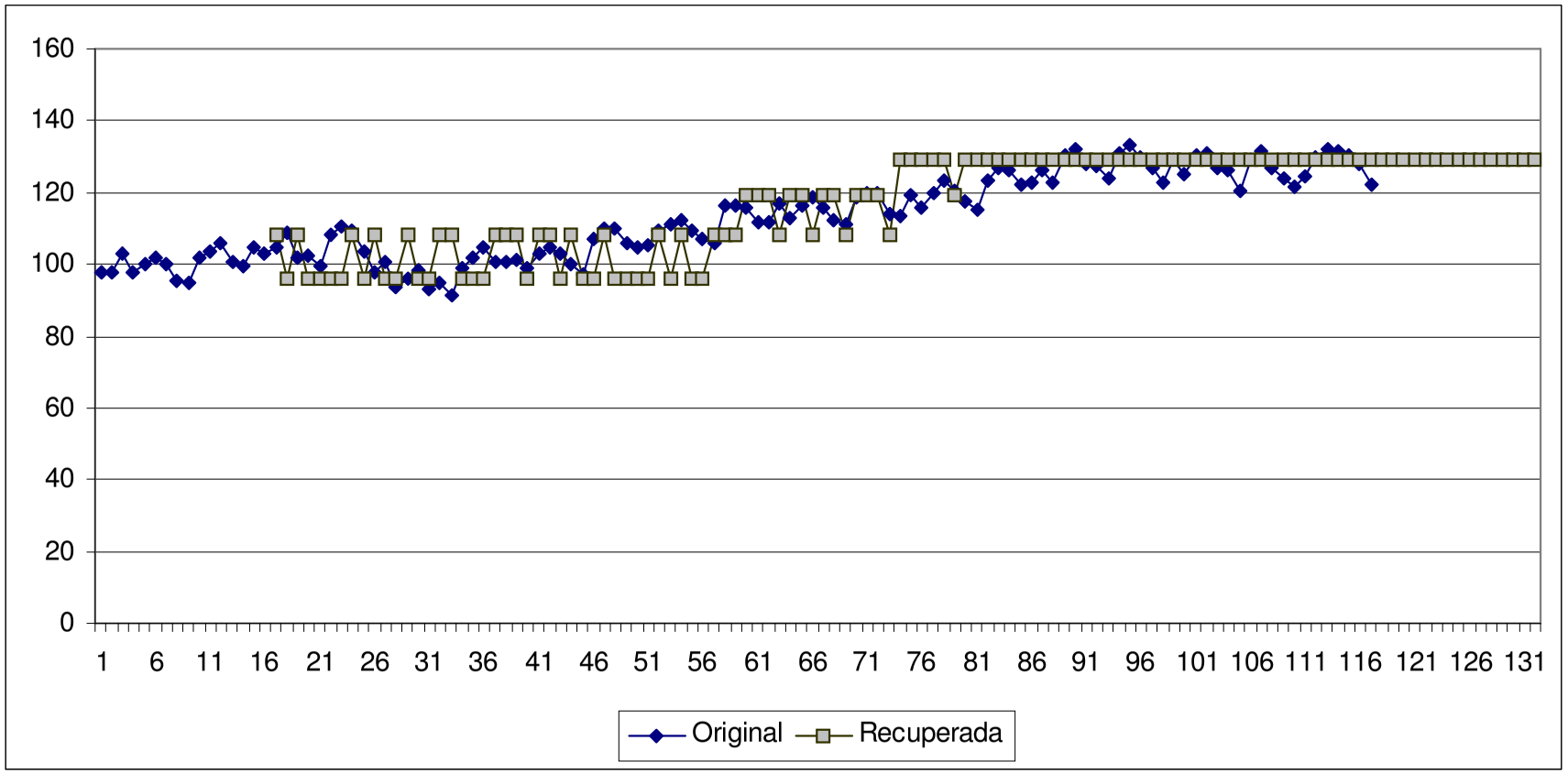


Figura 7.22. Serie de tiempo *Actividad económica* original y recuperada a partir de la red de la figura 7.21

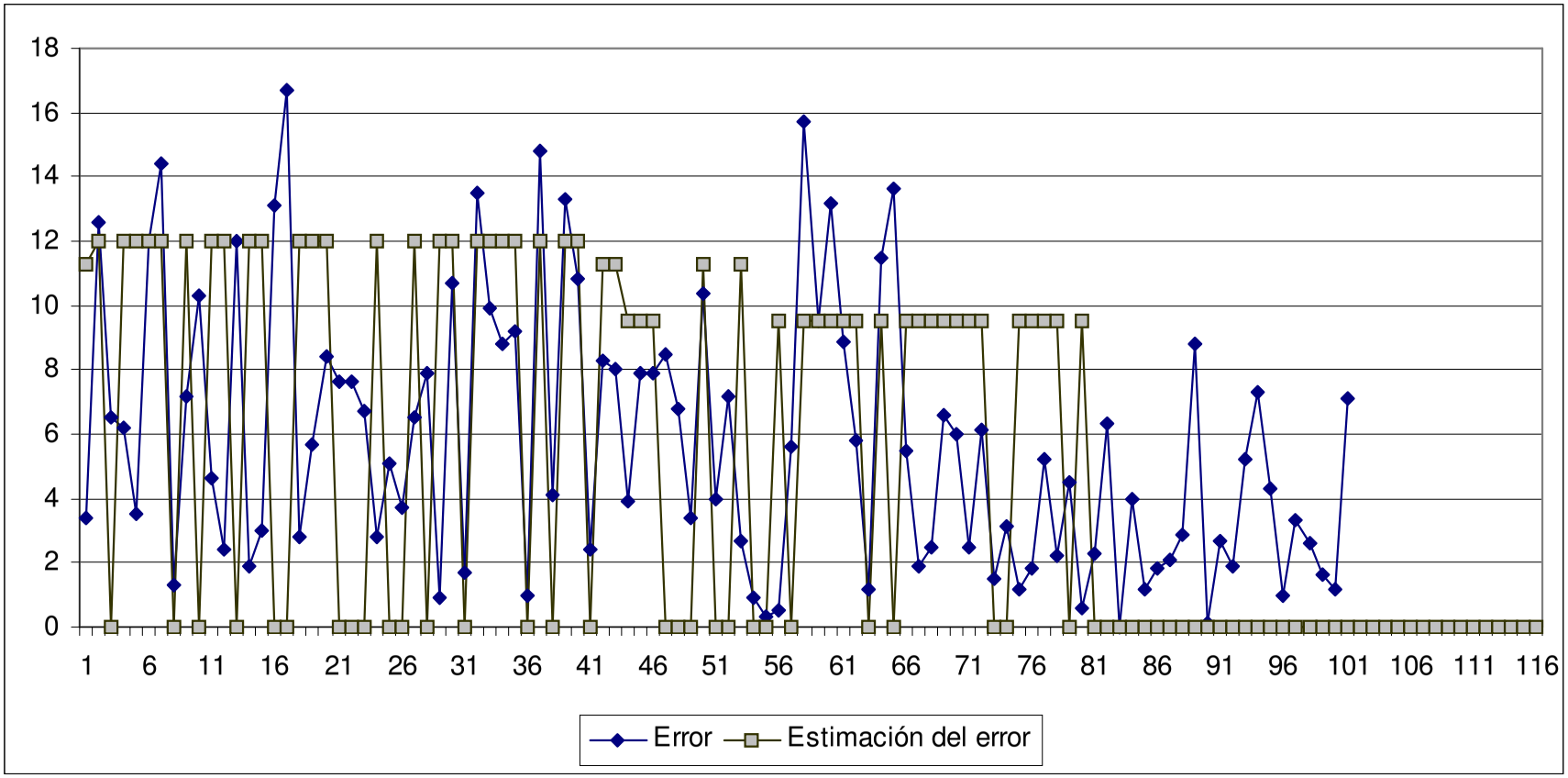


Figura 7.23. Error y estimación del error de la recuperación de la variable *Actividad económica*, mostrada en la figura 7.22

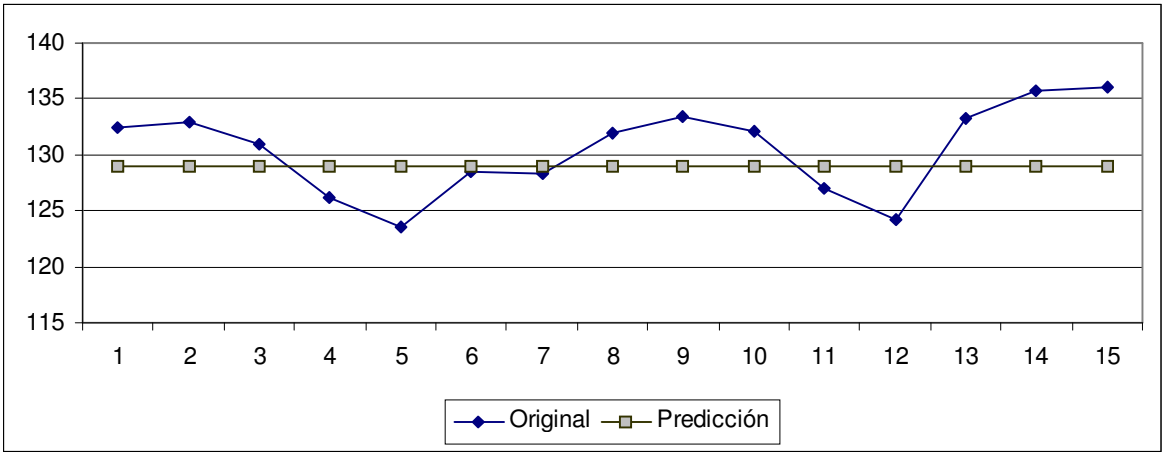


Figura 7.24. Predicción de la serie *Actividad económica*

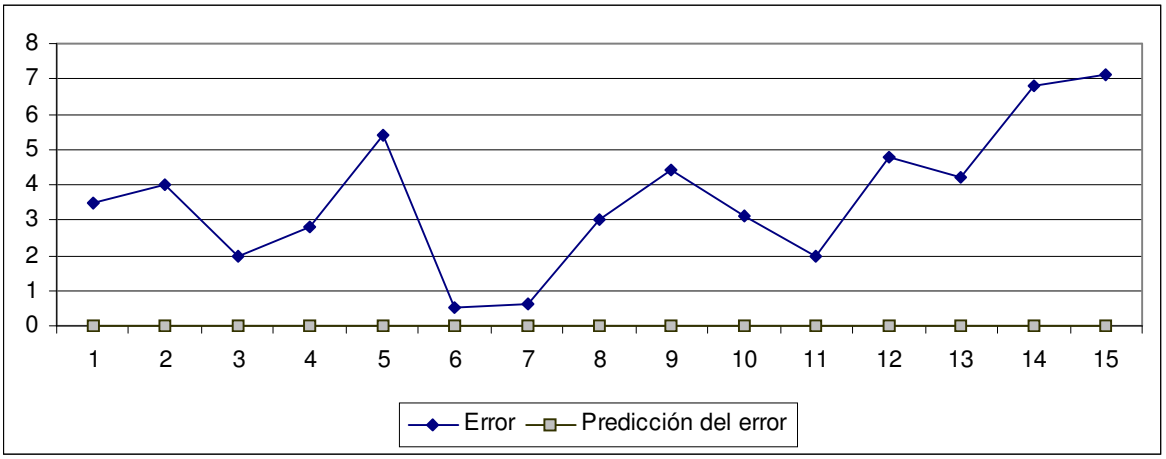


Figura 7.25. Error de predicción de la serie *Actividad económica*

La variable *Deuda pública valores* se recuperó dadas las series *Inflación*, *IPC BMV*, *Importaciones* y *Exportaciones*. La recuperación se muestra en la figura 7.26, y el error de recuperación en la figura 7.27. Los valores predichos se muestran en la figura 7.28, y el error de predicción en la figura 7.29.

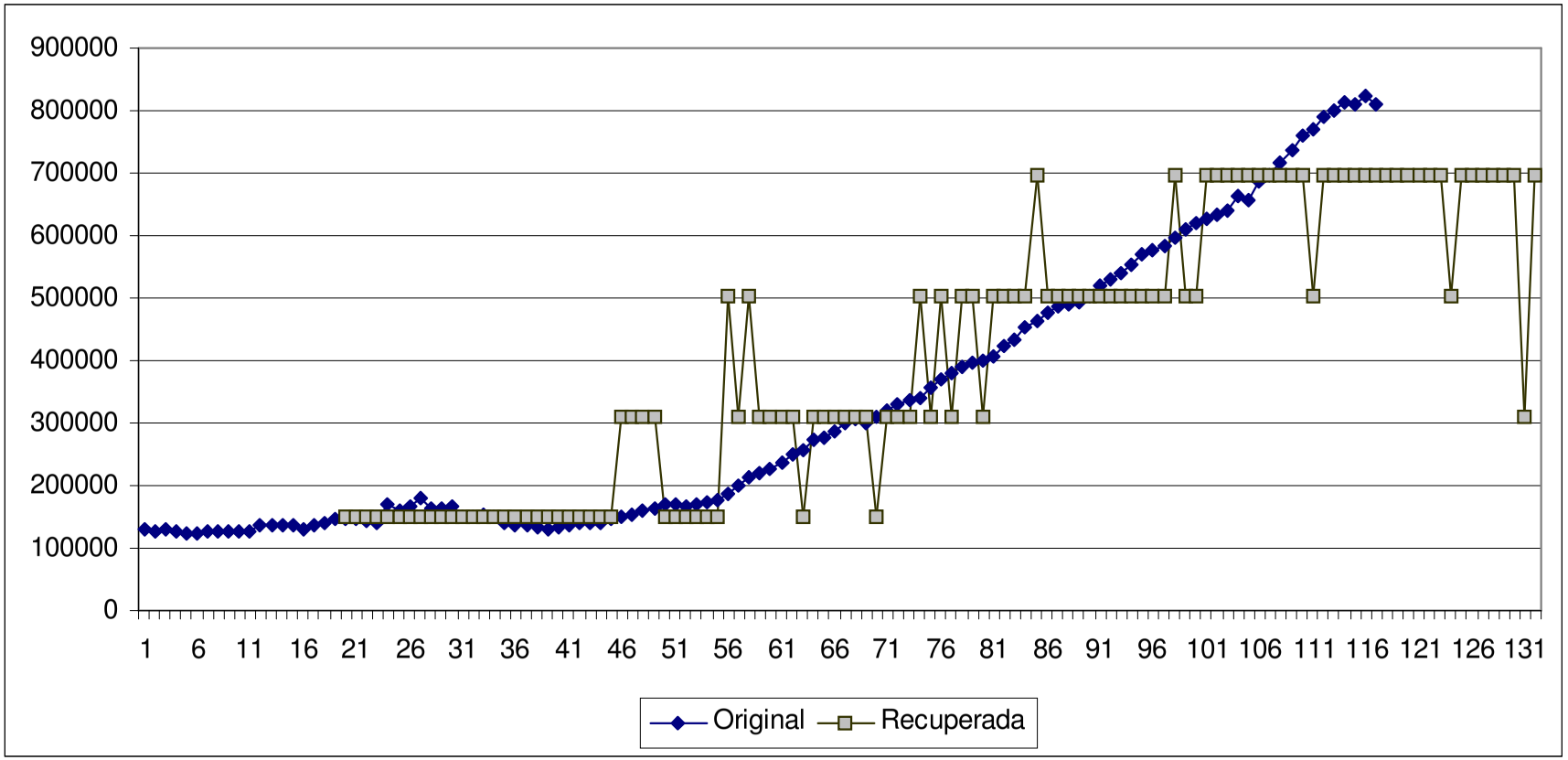


Figura 7.26. Serie de tiempo *Deuda pública valores* original y recuperada a partir de la red de la figura 7.21

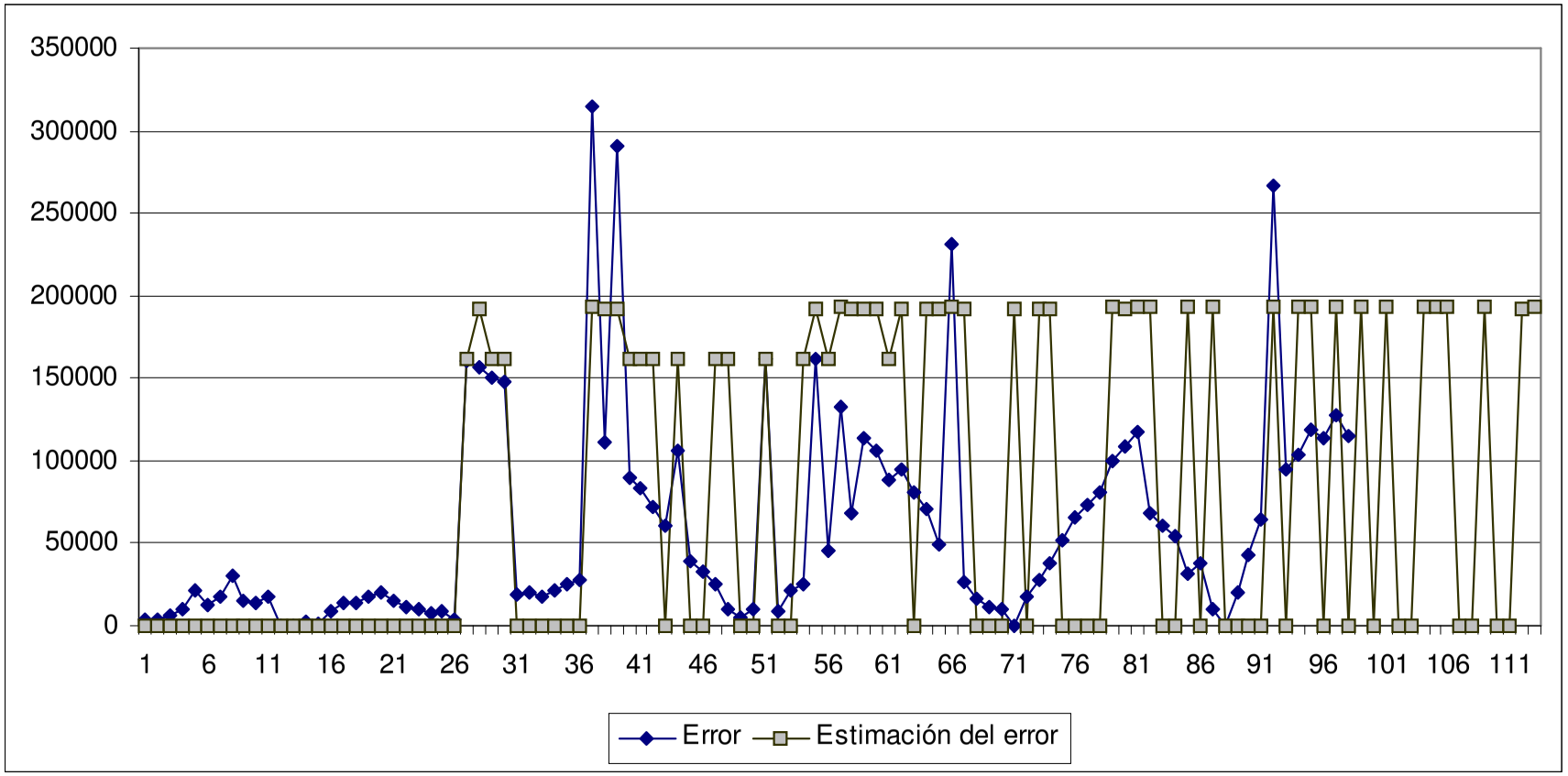


Figura 7.27. Error y estimación del error de la recuperación de la variable *Deuda pública valores*, mostrada en la figura 7.26

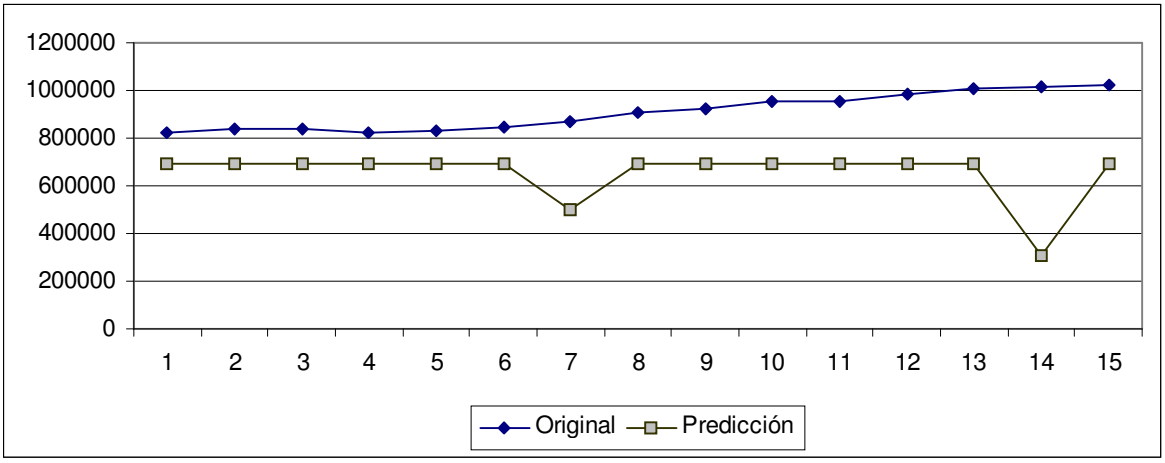


Figura 7.28. Predicción de la serie *Deuda pública valores*

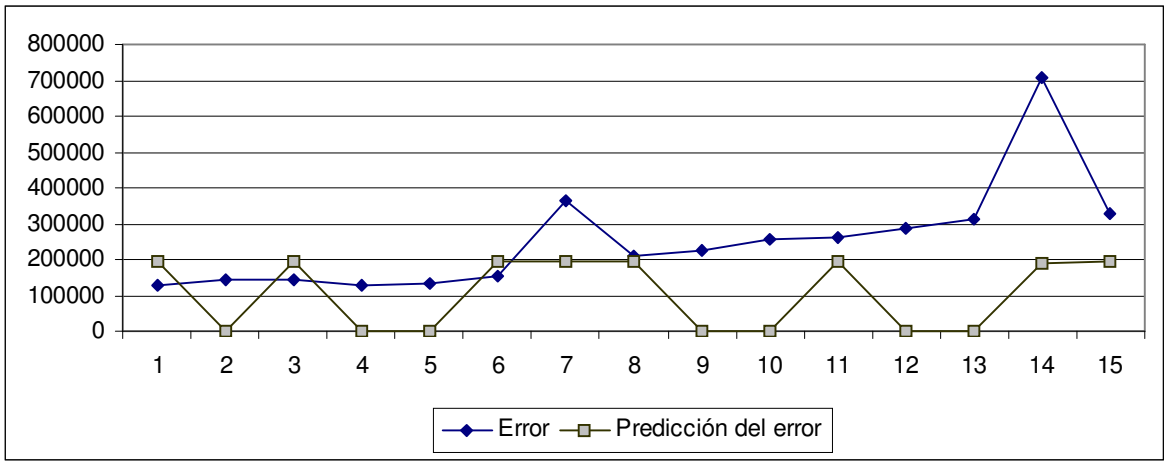


Figura 7.29. Error de predicción de la serie *Deuda pública valores*

La variable *Salarios en la industria manufacturera* se recuperó a partir de la serie *Importaciones*. La recuperación se muestra en la figura 7.30 y el error de recuperación en la figura 7.31. La predicción se muestra en la figura 7.32 y el error de predicción en la figura 7.33. En este último se observa una cercanía notoria entre el error verdadero y su estimación, excepto en los puntos correspondientes a Diciembre del 2002 y Diciembre del 2003.

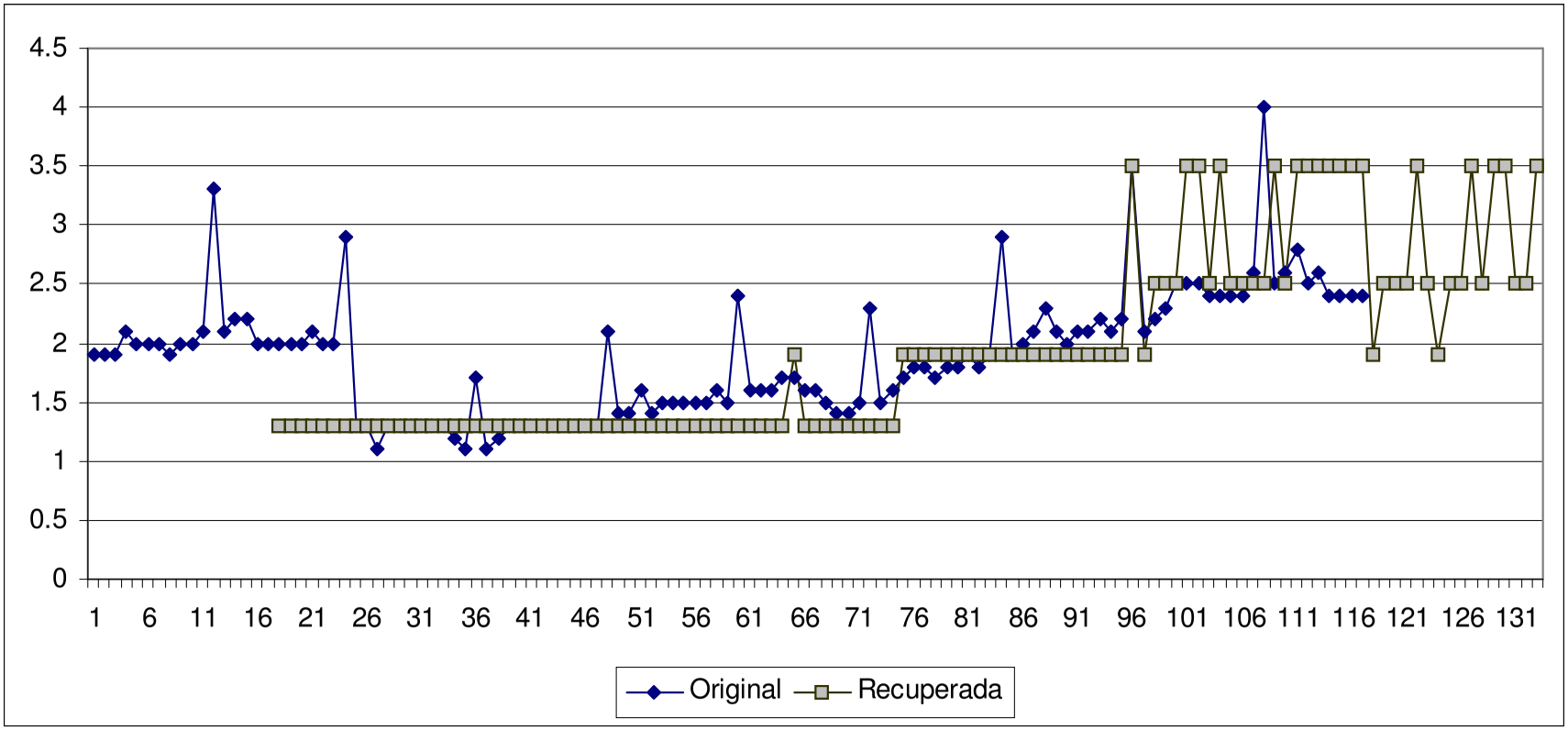


Figura 7.30. Serie de tiempo *Salarios industria manufacturera* original y recuperada a partir de la red de la figura 7.21

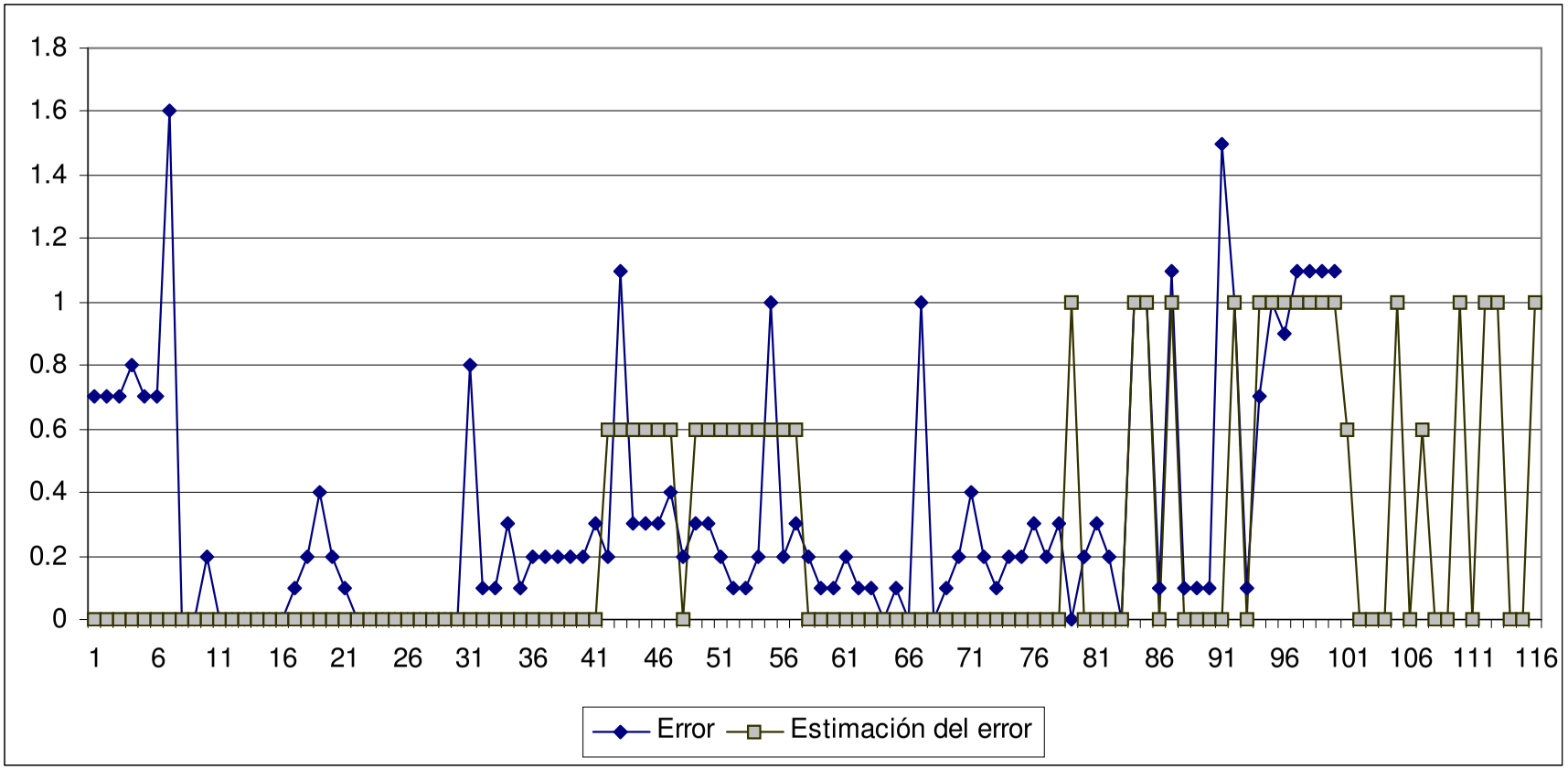


Figura 7.31. Error y estimación del error de la recuperación de la serie *Salarios industria manufacturera*, mostrada en la figura 7.30

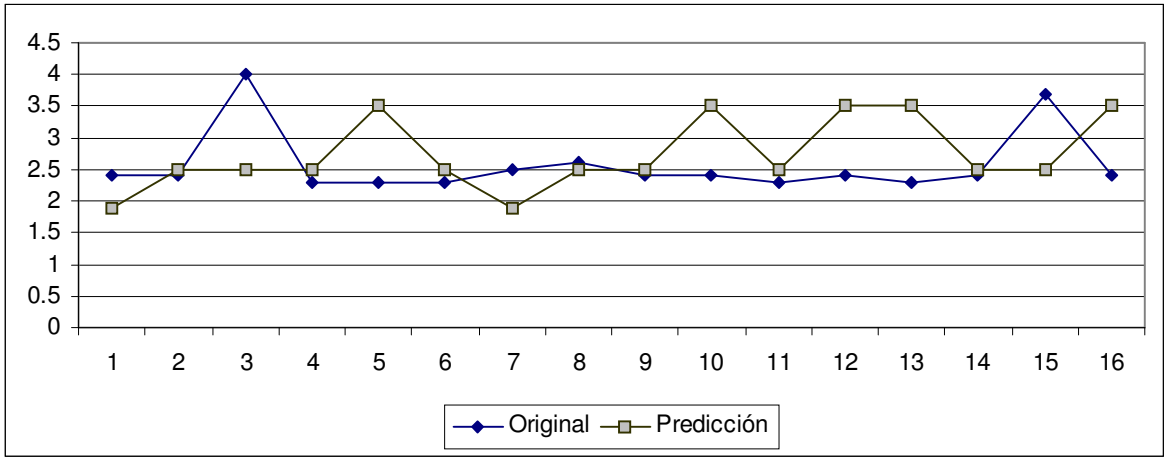


Figura 7.32. Predicción de la serie *Salarios industria manufacturera*

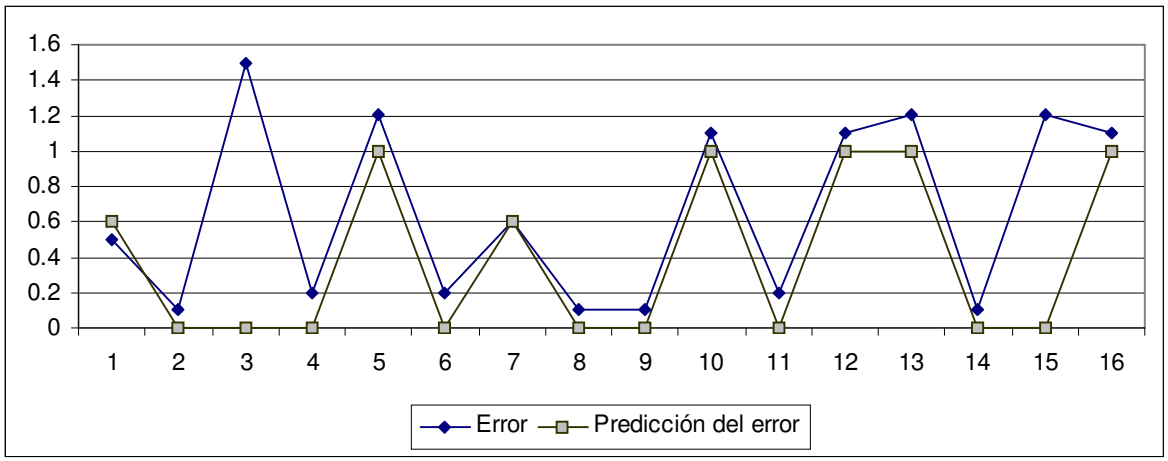


Figura 7.33. Error de predicción de la serie *Salarios industria manufacturera*

7.3.4 Prueba 4

La figura 7.34 muestra la Red Bayesiana obtenida a partir de las secuencias alineadas obtenidas en la sección 6.3.4 utilizando el algoritmo MLE. A partir de esta red se han recuperado las series *Deuda pública en valores*, *Inflación* y *Actividad económica*.

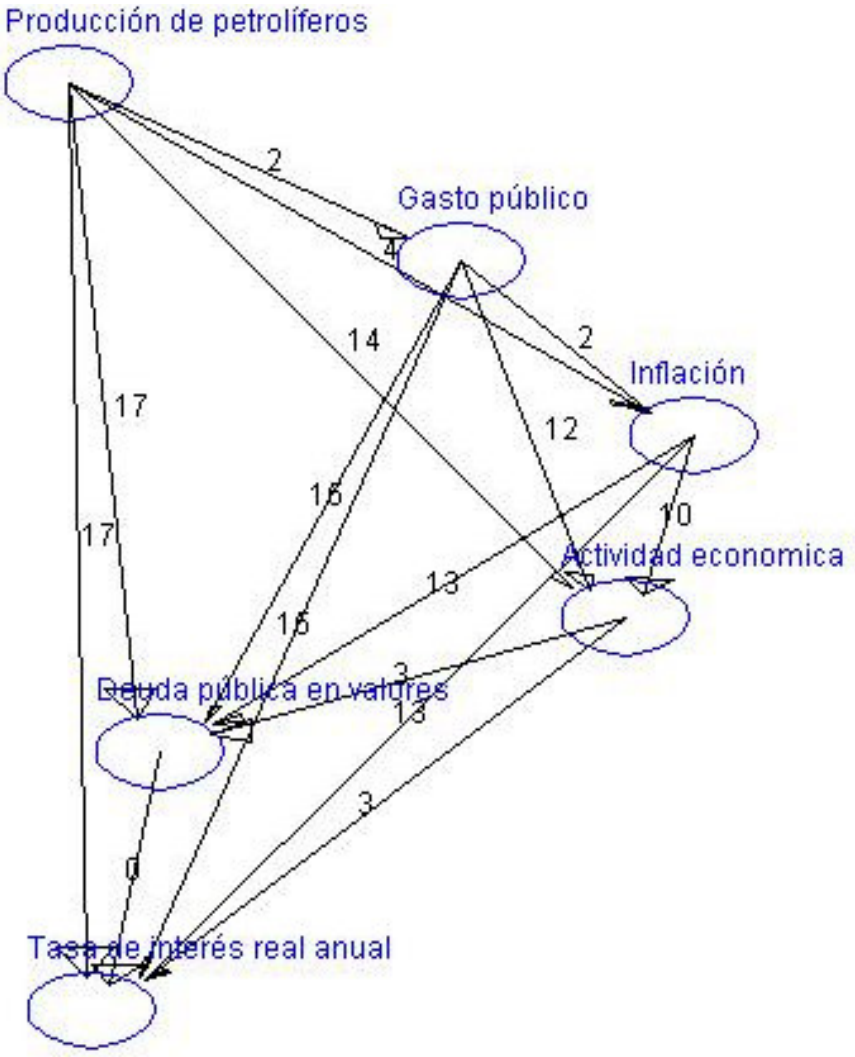


Figura 7.34. Red Bayesiana obtenida utilizando el algoritmo MLE

La recuperación de la serie *Deuda pública en valores* se muestra en la figura 7.35, mientras que el error de discretización y su estimación se muestran en la figura 7.36. Para la recuperación de la serie se utilizaron los valores originales de *Inflación*, *Producción de petrolíferos*, *Gasto público* y *Actividad económica*.

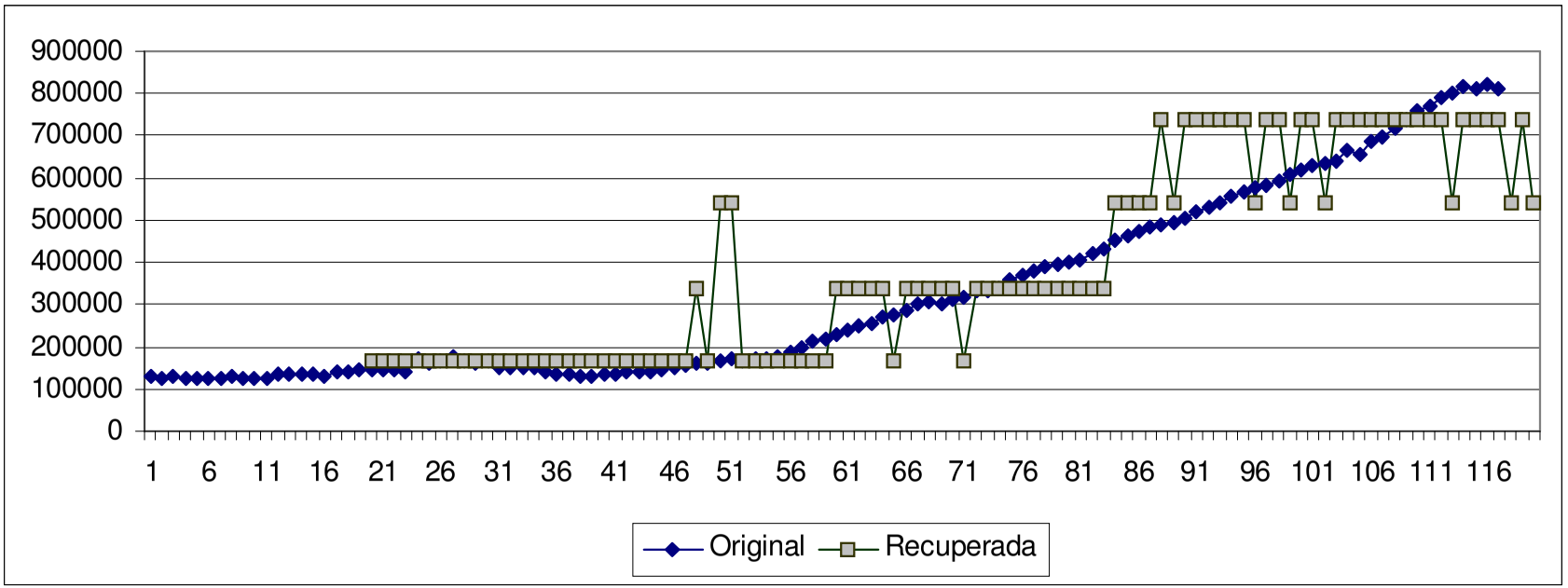


Figura 7.35. Recuperación de la serie *Deuda pública en valores* a partir de la red obtenida con MLE (figura 7.34)

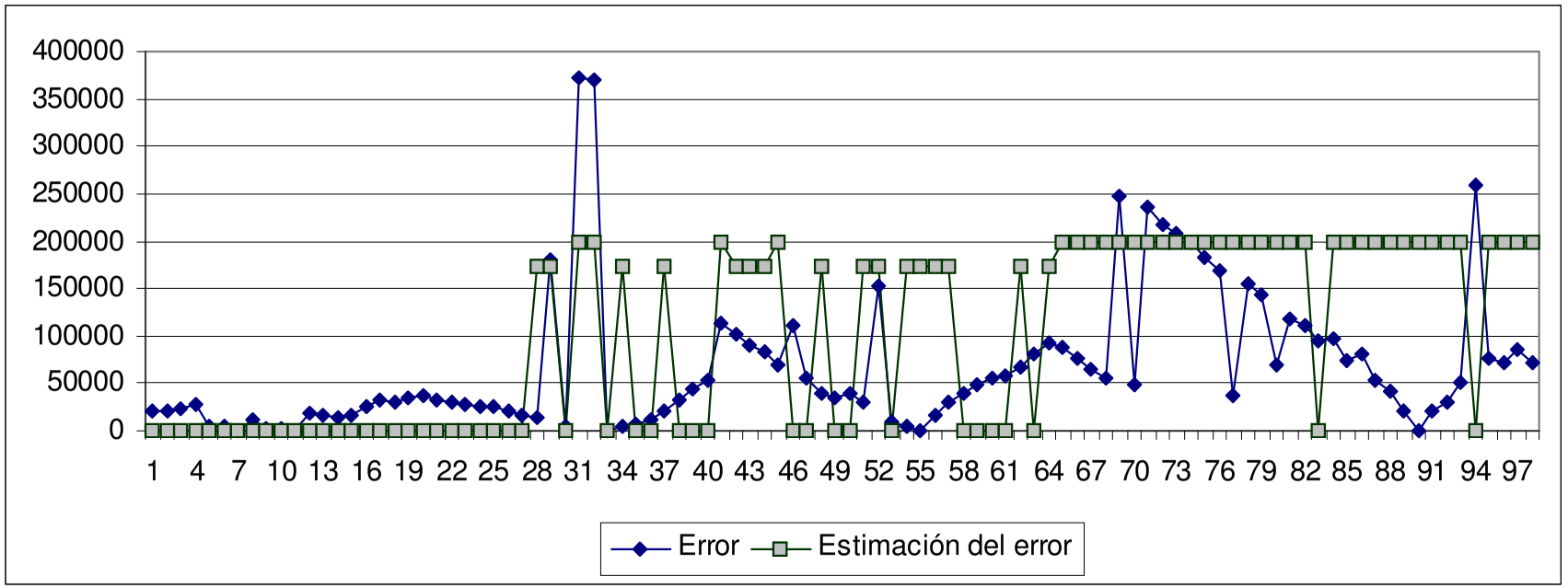


Figura 7.36. Error y estimación del error de la recuperación de la serie *Deuda pública en valores*, mostrada en la figura 7.35

Como se puede observar en la recuperación mostrada en la figura 7.35, a partir de la Red Bayesiana se han obtenido tres valores posteriores a los datos de entrenamiento para la serie de tiempo *Deuda pública en valores*. La figura 7.37 muestra tanto los valores predichos como los reales. La figura 7.38 muestra el error de predicción y su estimación.

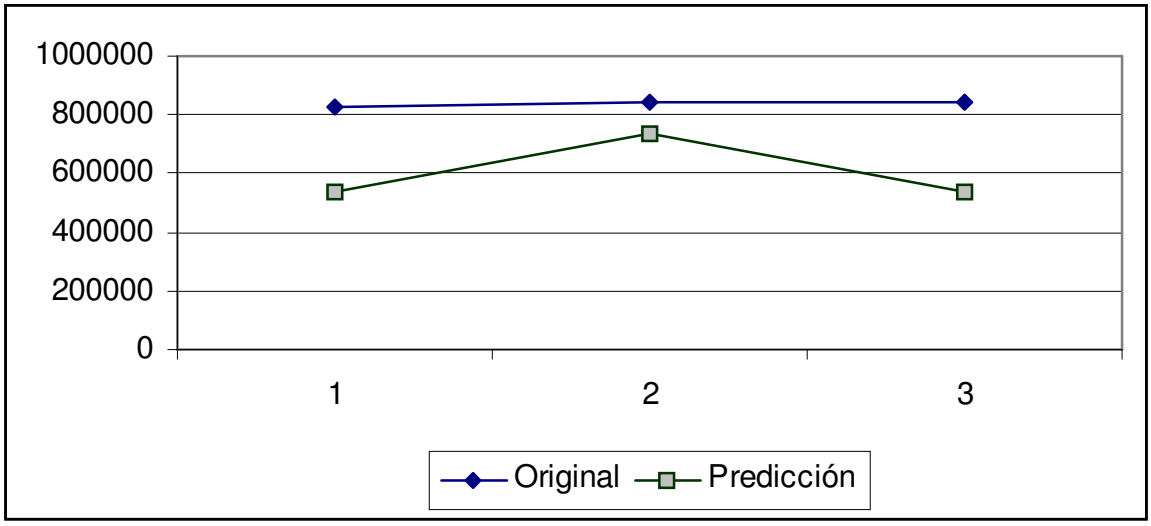


Figura 7.37. Predicción de la serie *Producción de petrolíferos* a partir de la red generada con MLE

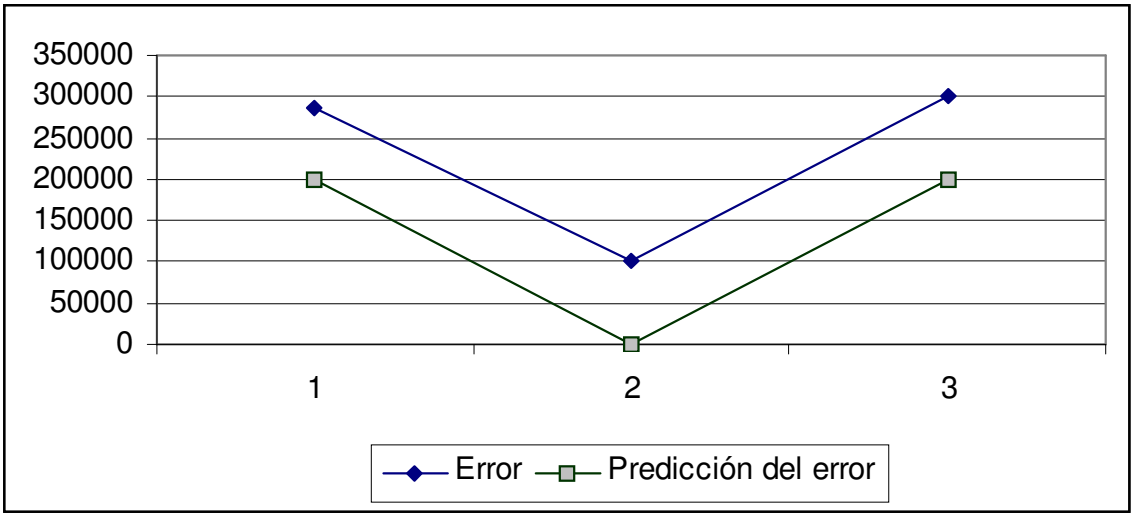


Figura 7.38. Error de predicción de la serie *Producción de petrolíferos* a partir de la red generada con MLE

La recuperación de la serie *Inflación* se muestra en la figura 7.39. Para esta recuperación se utilizaron los valores originales de *Producción de petrolíferos y Gasto público*. El error de recuperación y su estimación se muestran en la figura 7.40.

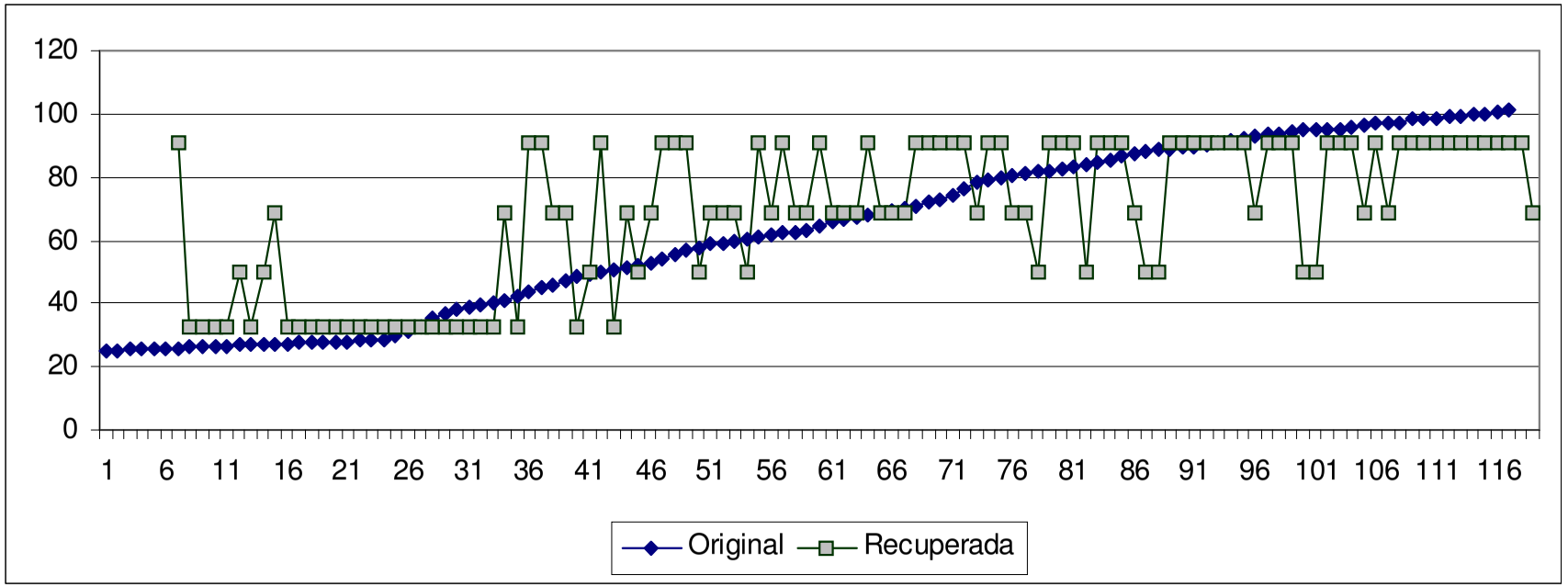


Figura 7.39. Recuperación de la serie *Inflación* a partir de la red obtenida con MLE (figura 7.34)

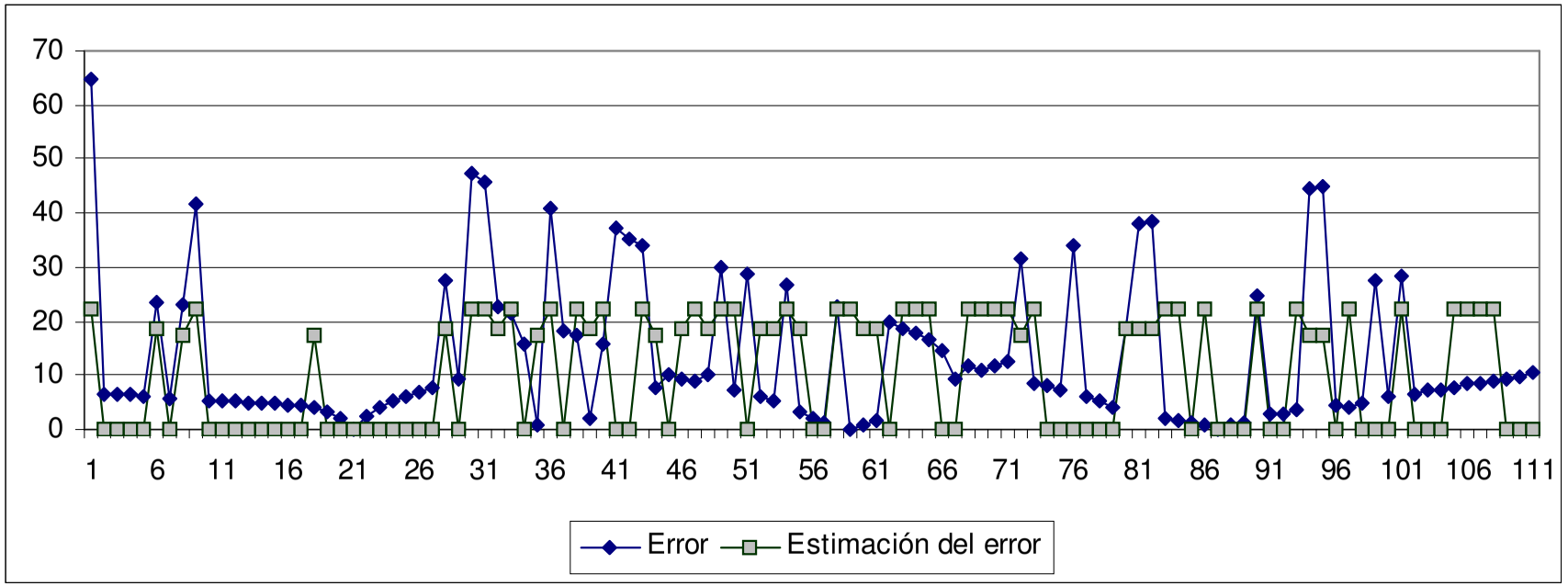


Figura 7.40. Error y estimación del error de la recuperación de la serie *Inflación*, mostrada en la figura 7.39

Durante la recuperación de la serie de tiempo *Inflación* se obtuvieron dos valores predichos, los cuales se muestran en la figura 7.41. El error de predicción y su estimación se muestran en la figura 7.42.

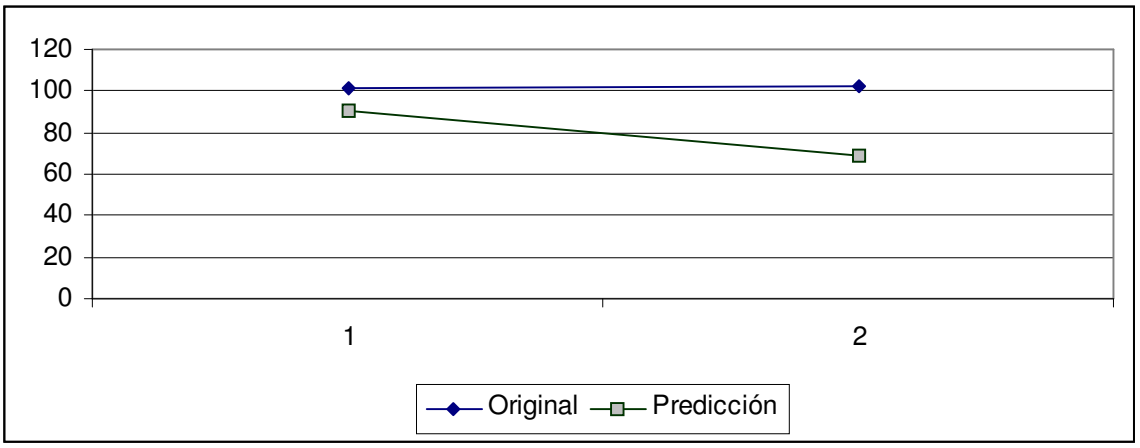


Figura 7.41. Predicción de la serie *Inflación* a partir de la red generada con MLE

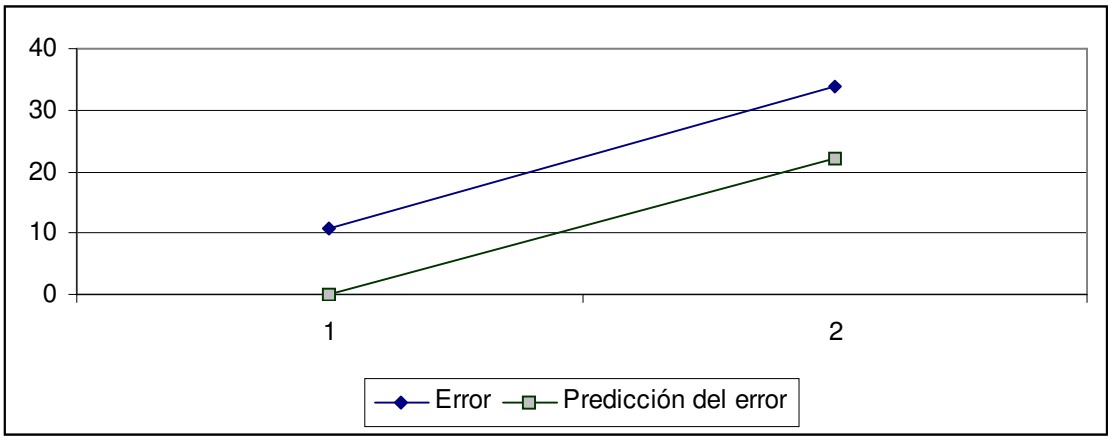


Figura 7.42. Error de predicción de la serie *Inflación* a partir de la red generada con MLE

La recuperación de la serie *Actividad económica* se muestra en la figura 7.43. Esta recuperación se realizó en base a los valores originales de *Producción de petrolíferos*, *Gasto público* e *Inflación*. El error y su estimación se muestran en la figura 7.44.

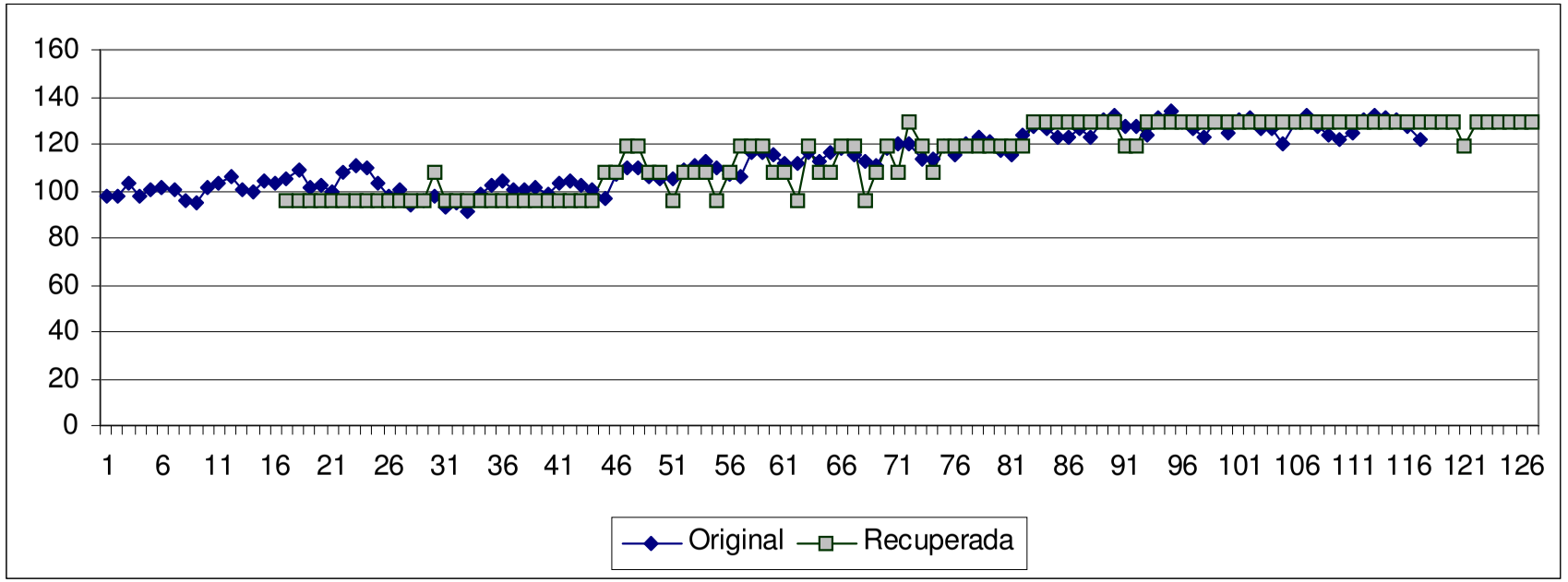


Figura 7.43. Recuperación de la serie *Actividad económica* a partir de la red obtenida con MLE (figura 7.34)

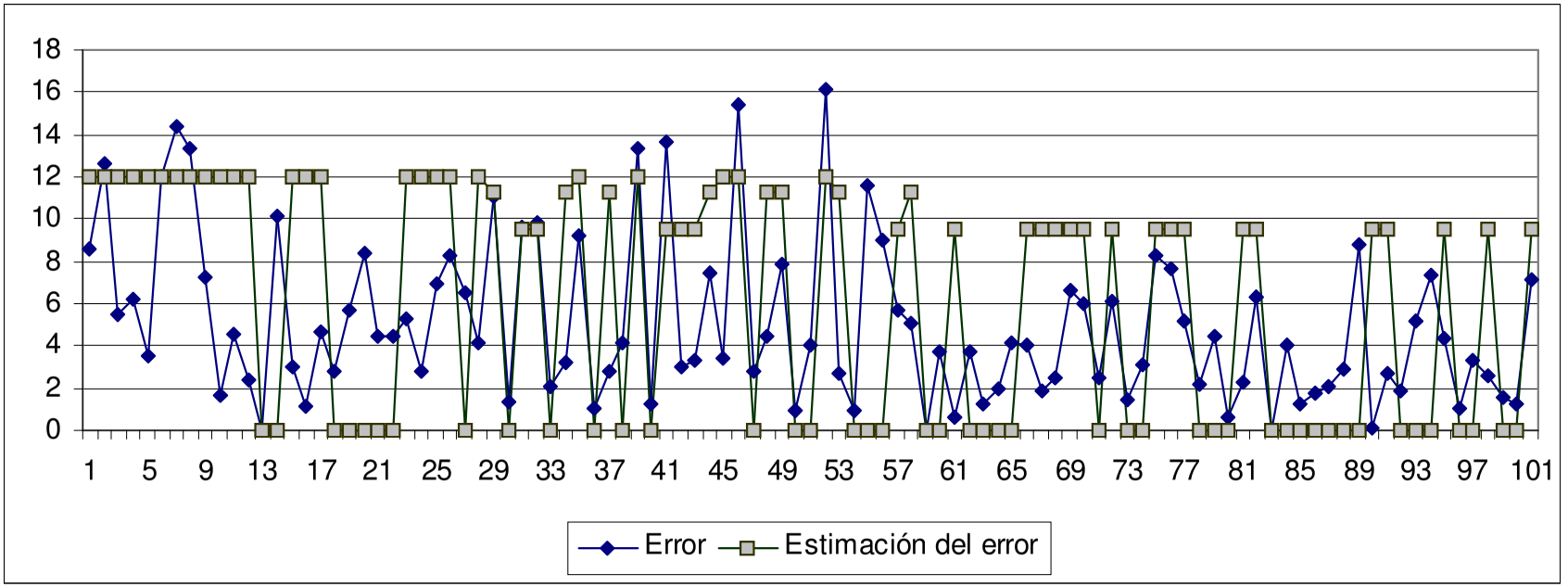


Figura 7.44. Error y estimación del error de la recuperación de la serie *Actividad económica*, mostrada en la figura 7.43

La figura 7.45 muestra los diez valores predichos para la serie de tiempo *Actividad económica*. La figura 7.46 muestra el error de predicción y su estimación.

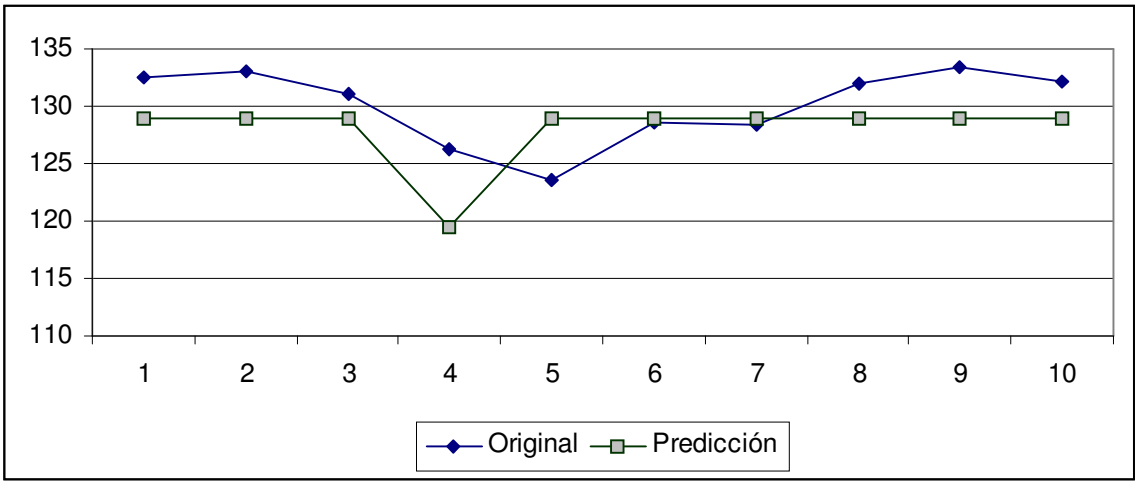


Figura 7.45. Predicción de la serie *Actividad económica* a partir de la red generada con MLE

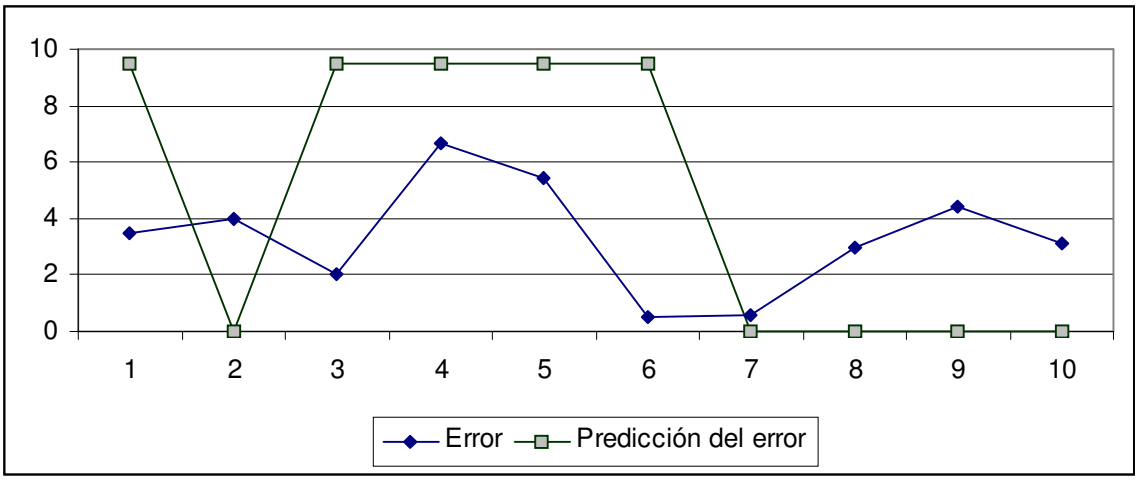


Figura 7.46. Error de predicción de la serie *Actividad económica* a partir de la red generada con MLE

La figura 7.47 muestra la Red Bayesiana obtenida utilizando el algoritmo de tres etapas. Como era de esperarse, esta red presenta menos arcos que la generada utilizando MLE.

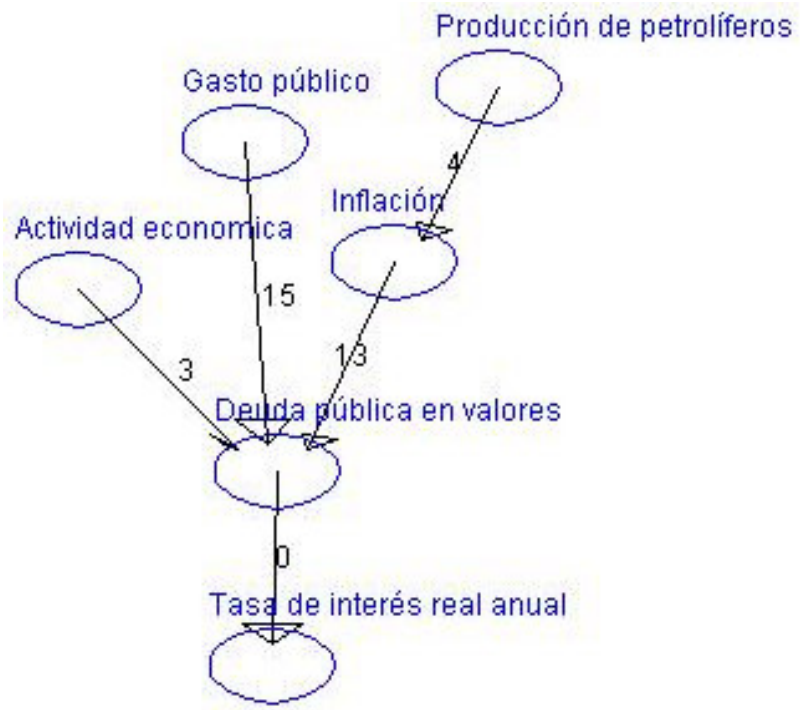


Figura 7.47. Red Bayesiana obtenida utilizando el algoritmo de tres etapas

La recuperación de la serie de tiempo *Deuda pública en valores* se muestra en la figura 7.48. El error de esta recuperación y su estimación se muestran en la figura 7.49. Para recuperar esta serie de tiempo se utilizaron los valores originales de las series de tiempo *Inflación*, *Actividad económica* y *Gasto público*.

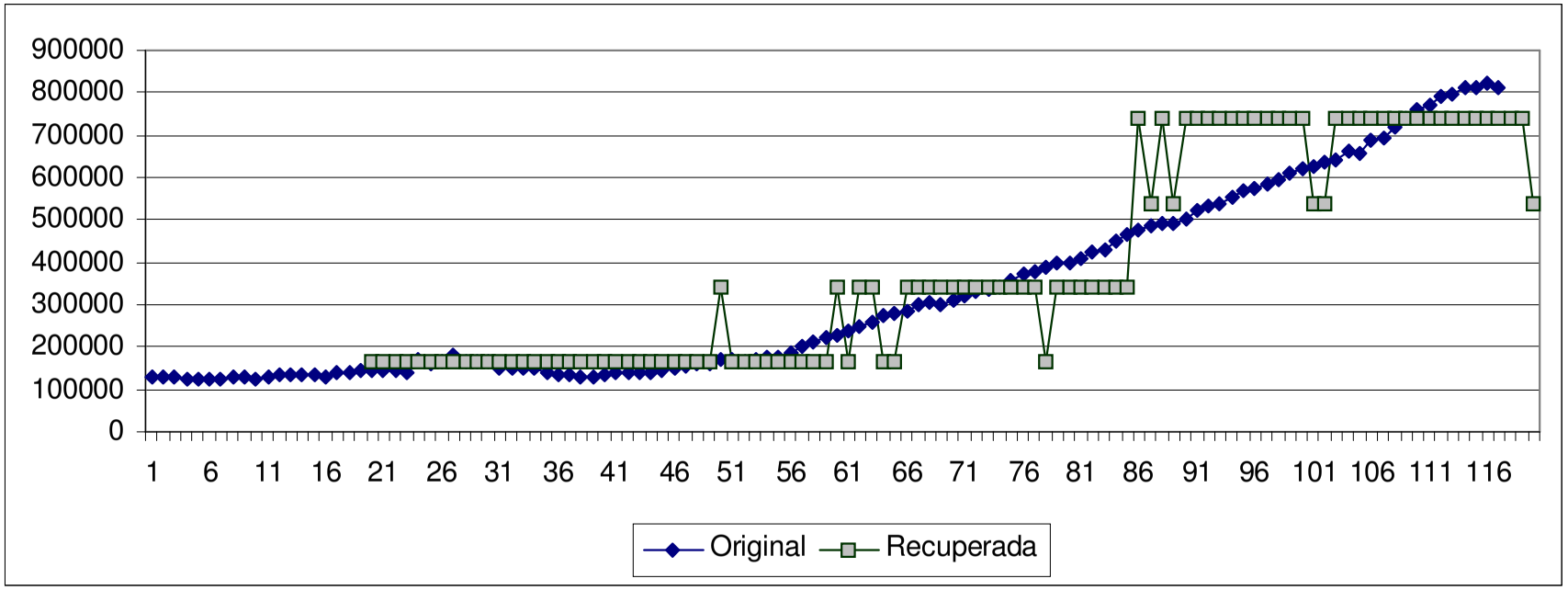


Figura 7.48. Recuperación de la serie *Deuda pública en valores* a partir de la red obtenida utilizando el algoritmo de tres etapas (figura 7.47)

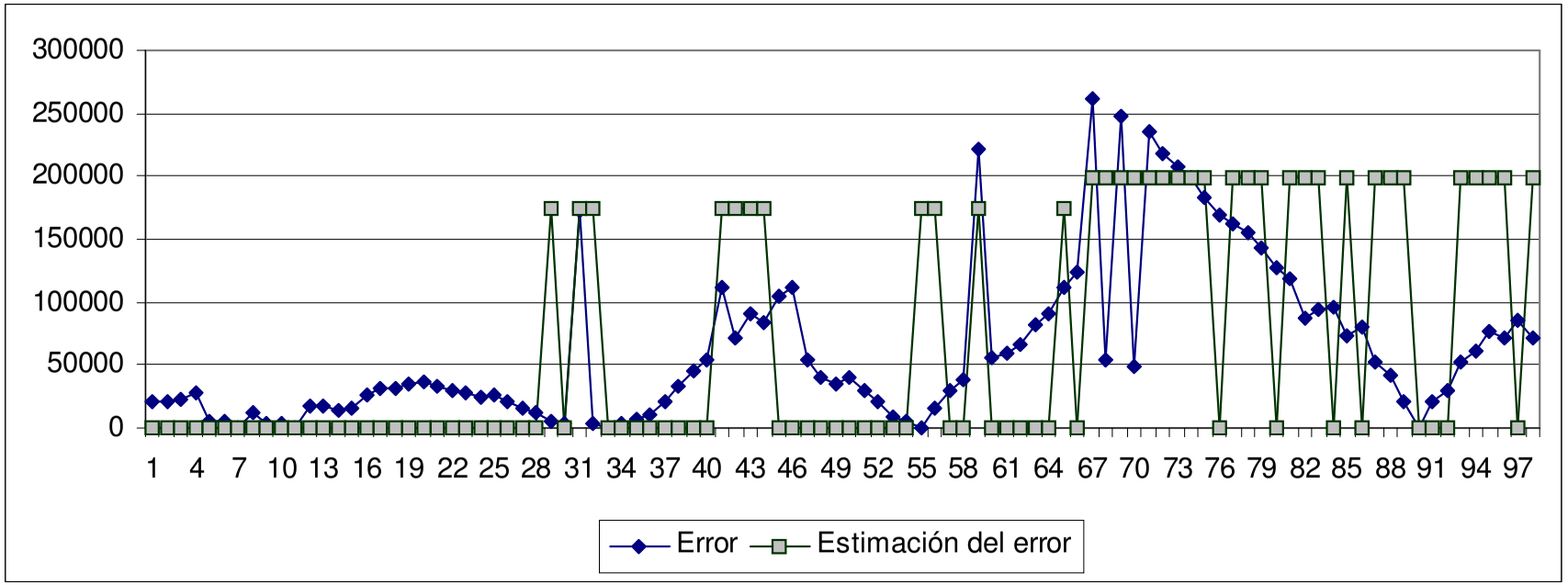


Figura 7.49. Error de la recuperación de la serie *Deuda pública en valores*, mostrada en la figura 7.48

La predicción de tres valores de la serie *Deuda pública en valores* se muestran en la figura 7.50. El error de predicción y su estimación previa se muestran en la figura 7.51.

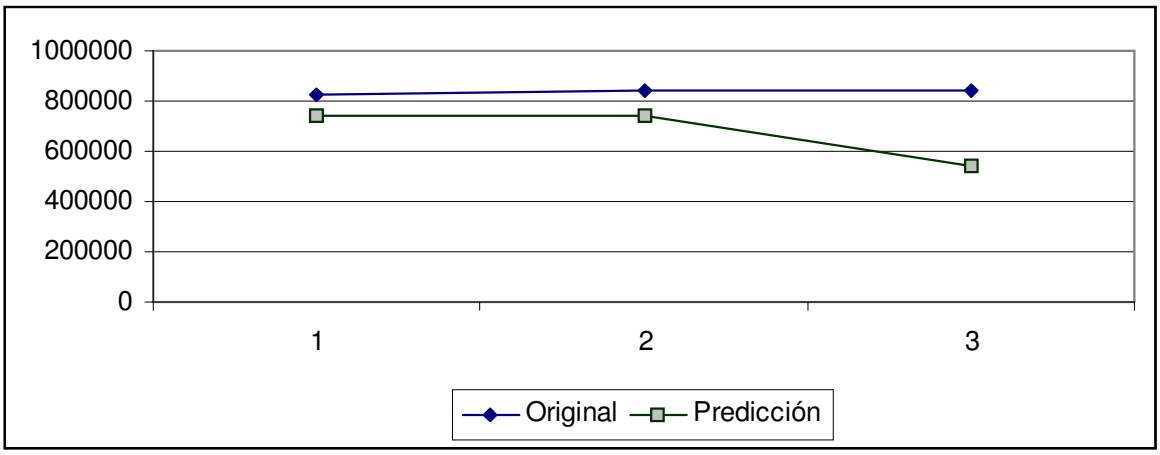


Figura 7.50. Predicción de la serie *Deuda pública en valores* a partir de la red generada con el algoritmo de tres etapas

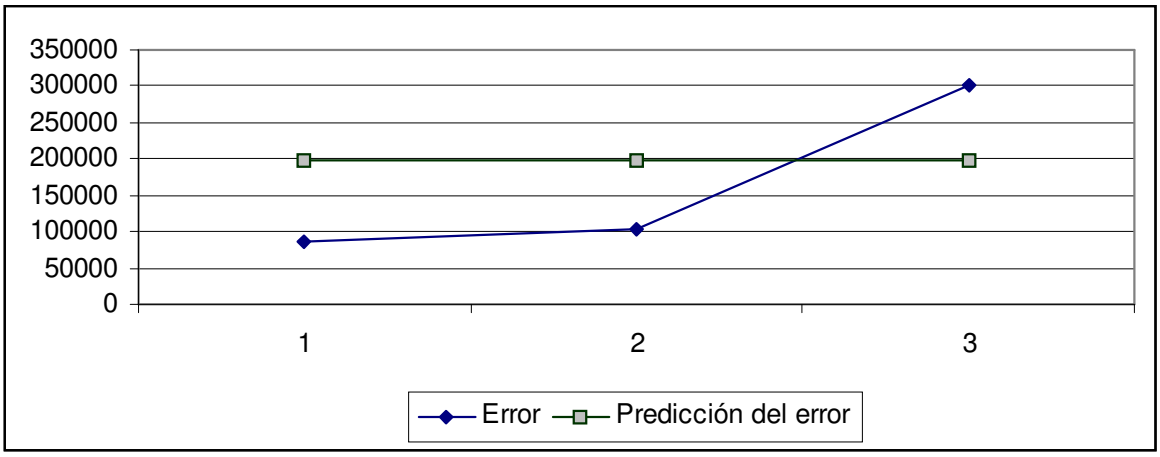


Figura 7.51. Error de predicción de la serie *Deuda pública en valores* a partir de la red generada con el algoritmo de tres etapas

La figura 7.52 muestra la recuperación de la serie *Inflación*. La figura 7.53 muestra el error de recuperación y su estimación. Esta serie fue recuperada valiéndose de los valores originales de *Producción de petrolíferos*.

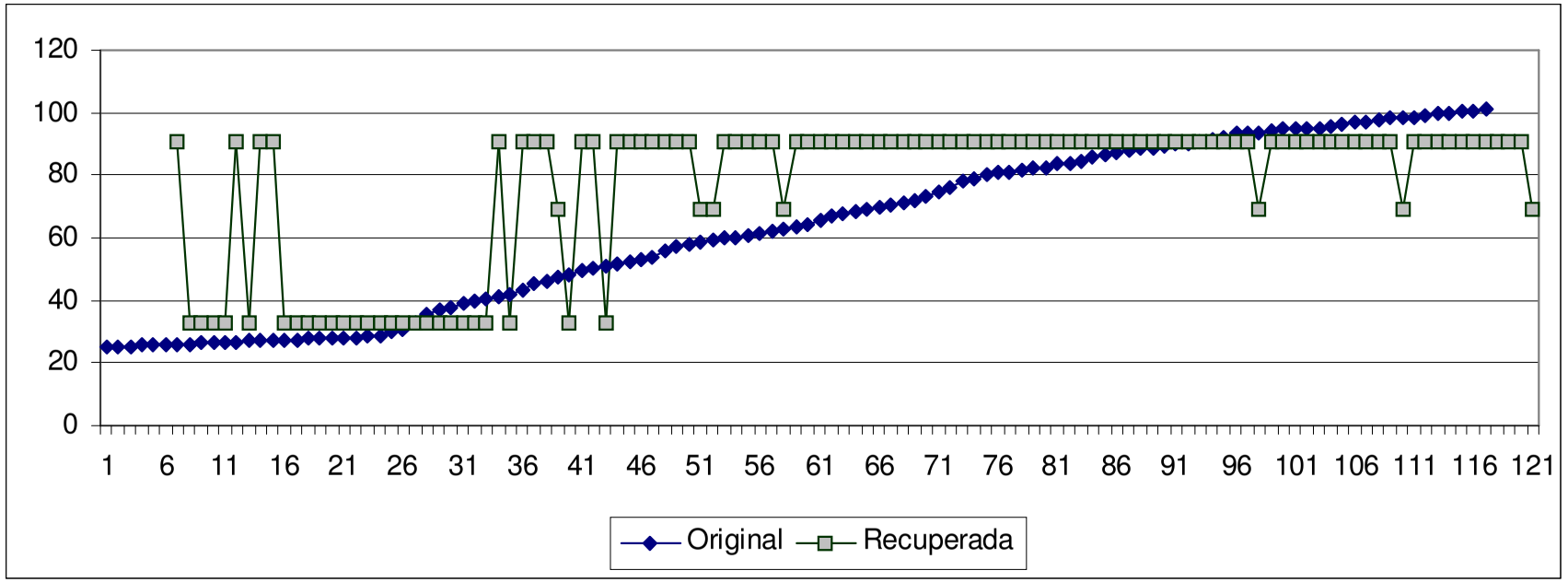


Figura 7.52. Recuperación de la serie *Inflación* a partir de la red obtenida utilizando el algoritmo de tres etapas (figura 7.47)

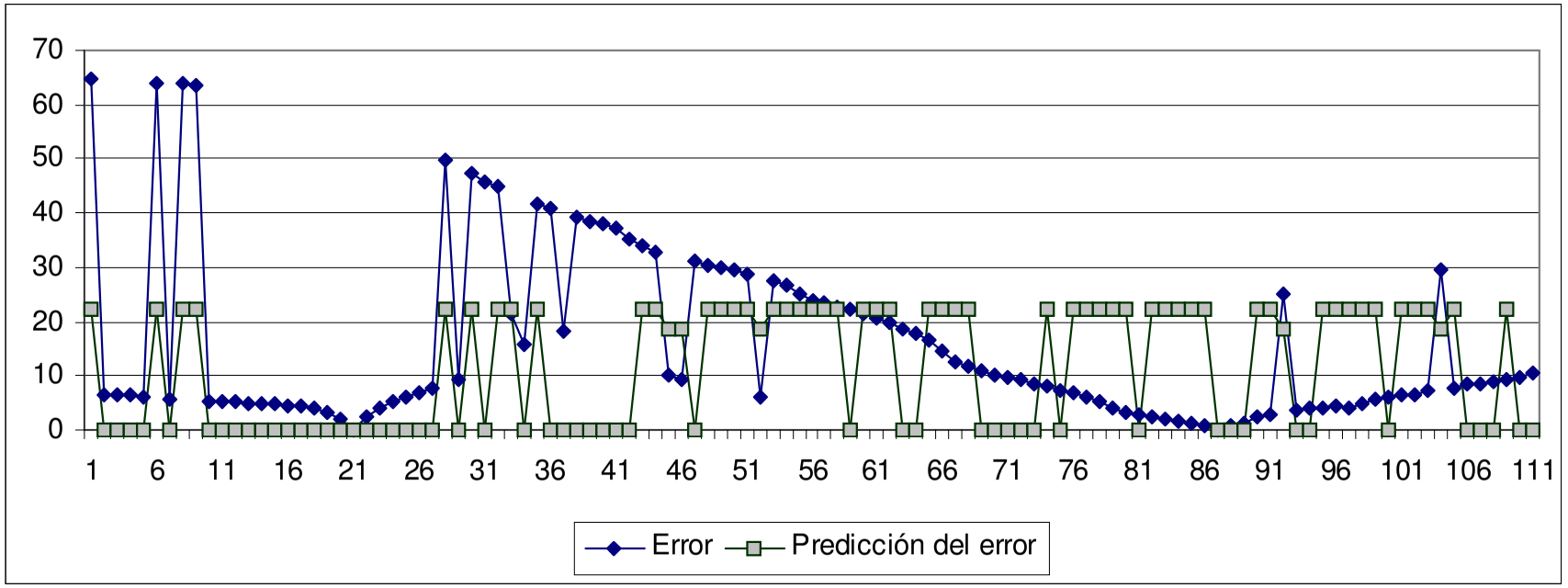


Figura 7.53. Error de la recuperación de la serie *Actividad económica*, mostrada en la figura 7.52

La predicción de cuatro valores de la serie *Inflación* se muestra en la figura 7.54. El error de predicción se muestra en la figura 7.55.

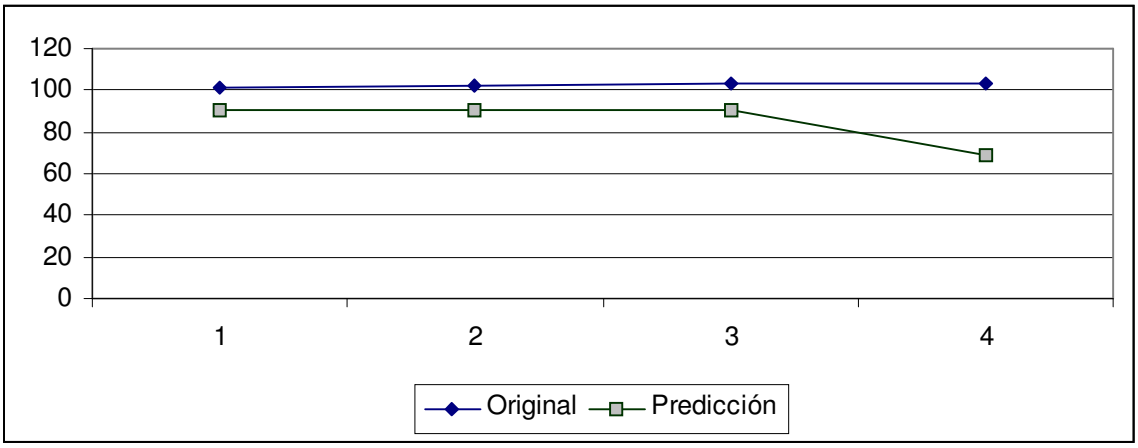


Figura 7.54. Predicción de la serie *Inflación* a partir de la red generada con el algoritmo de tres etapas

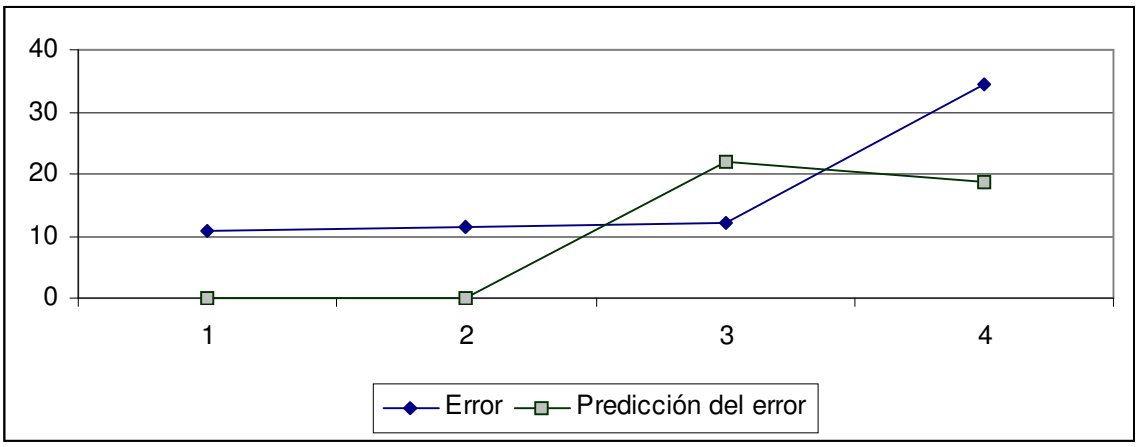


Figura 7.55. Error de predicción de la serie *Inflación*
a partir de la red generada con el algoritmo de tres etapas

7.3.5 Sumario de resultados

Las series de tiempo, las pruebas en que fueron incluidas y el valor de σ utilizado para su discretización se muestran en la tabla 7.1. En las columnas referentes a las pruebas, una *U* indica que la serie de tiempo fue utilizada en esa prueba, mientras que una *R* indica que fue recuperada a partir de la Red Bayesiana, y una *P* indica que fue predicha. Para cada prueba, en cada serie utilizada se indica el valor de σ con el que fue discretizada. El número de figura mostrado para cada serie se refiere a la serie de tiempo original.

Serie de tiempo	Figura	Prueba 1		Prueba 2		Prueba 3		Prueba 4	
		Uso	σ	Uso	σ	Uso	σ	Uso	σ
Actividad económica	6.3	U	0			URP	0	URP	0
Desempleo abierto	6.4	URP	0			U	0		
IPC BMV	6.5	U	302.145			U	9.023		
Inflación	6.6	U	4128.888			U	11000.041	URP	0
Exportaciones	6.11			U	21572.473	U	26377.109		
Precios al productor	6.12			U	1337.463				
Productividad de la mano de obra	6.13			U	0				
Costo de captación de moneda nacional	6.14			URP	0	U	0		
Deuda pública en valores	6.19					URP	17500.781	URP	7722.524
Importaciones	6.20					U	40.823		
Producción de petrolíferos	6.21					U	1.742	U	0
Salarios en la industria manufacturera	6.22					URP	0		
Tasa de interés real anual	6.33							U	0
Gasto público	6.34							U	192.196

Tabla 7.1. Información sobre las series de tiempo utilizadas durante las pruebas.
U = Utilizada, R = Recuperada, P = Predicha

Una de las medidas de error más utilizadas es el error cuadrático medio (Mean Square Error, MSE), que es igual a la media de los cuadrados de los errores

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(x_i - y_i)^2$$

en donde m es el número de valores, x_1, x_2, ..., x_m son los valores obtenidos y y_1, y_2, ..., y_m son los valores objetivo.

Para medir la calidad de las recuperaciones se utilizó una medida conocida como MSER [Battaglia, 1996], que es básicamente una normalización del error cuadrático medio. Se calcula como

$$MSER = \frac{MSE}{(NSL - T)^2}$$

en donde T es el valor objetivo (Target) y el parámetro *NSL* (Nearer Specification Limit) especifica un límite de cercanía a T.

La tabla 7.2 muestra el error MSER asociado a cada una de las recuperaciones en las diferentes pruebas. Se ha asignado el valor cero al parámetro MSL. En esta medición se incluyeron también los datos predichos.

Prueba	**Red Bayesiana**		**Serie de tiempo recuperada**	**Error MSER**
	Figura	**Extracción**		
1	7.1	MLE	Desempleo abierto	0.16847859
	7.2	Algoritmo de tres etapas		0.05972780
2	7.11	MLE	Costo de captación de moneda nacional	0.48921255
	7.12	Algoritmo de tres etapas		0.40941197
3	7.21	Algoritmo de tres etapas	Actividad económica	0.00382237
			Deuda pública en valores	0.11460205
			Salarios en la industria manufacturera	0.05343067
4	7.34	MLE	Deuda pública en valores	0.14904649
			Inflación	0.17011338
			Actividad económica	0.00297761
	7.47	Algoritmo de tres etapas	Deuda pública en valores	0.05516805
			Inflación	0.31905607

Tabla 7.2. MSER de la recuperación de series de tiempo

En esta tabla se observa que la mejor recuperación se obtuvo en la prueba 4 para la serie *Actividad económica*, cuando se recuperó a partir de la red obtenida con el algoritmo MLE. La peor recuperación se obtuvo en la prueba 2, al recuperar la serie *Costo de captación de moneda nacional* mediante el algoritmo MLE. Estas recuperaciones se presentaron en las figuras 7.43 y 7.13 respectivamente.

8. DISCUSIÓN

8.1 Valor de los parámetros

Indudablemente, el parámetro más importante durante el proceso de extracción de Redes Bayesianas Predictivas es σ, el cual define la forma de discretización para cada serie de tiempo. Dado que la alineación y la extracción de las Redes Bayesianas utilizan los valores discretos, un cambio en este parámetro para alguna serie de tiempo modifica la Red Bayesiana resultante.

Existen varios criterios que se deben tomar en cuenta al definir el valor de σ. El primero y más importante es el grado de importancia de la pendiente en la discretización. En la ecuación 5.1 se observa que la recuperación de una serie de tiempo discretizada se basa en el valor de σ e I, en donde I representa el intervalo dinámico de la serie de tiempo. Así, si $\sigma < I$, la serie de tiempo es discretizada dando más peso a su amplitud. Opuestamente, si $\sigma > I$, la pendiente de la serie de tiempo tiene más peso. Si $\sigma \approx I$, la discretización toma en cuenta la amplitud y la pendiente de la serie de manera equilibrada.

Otro criterio importante para elegir un valor de σ es el de maximizar la relación señal a ruido (SNR) de la serie de tiempo recuperada respecto a la original. Generalmente el comportamiento de la SNR al variar σ es una función que presenta varios máximos y mínimos locales. Como ejemplo, en la figura 8.1 se muestra una serie de tiempo calculada a partir del atractor de Lorenz. La figura 8.2 muestra el comportamiento del SNR al variar σ cuando la serie de tiempo se discretiza utilizando cuatro símbolos.

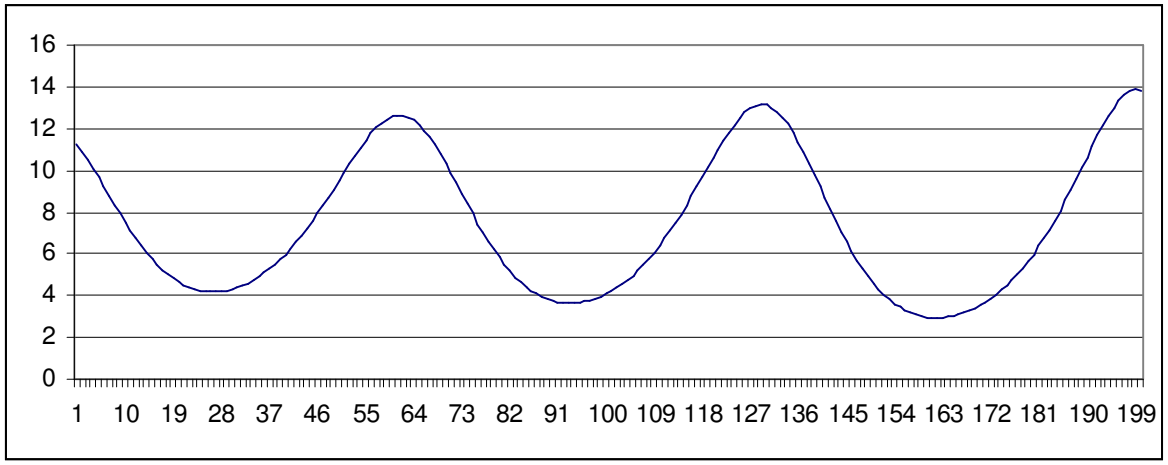


Figura 8.1. Serie de tiempo original

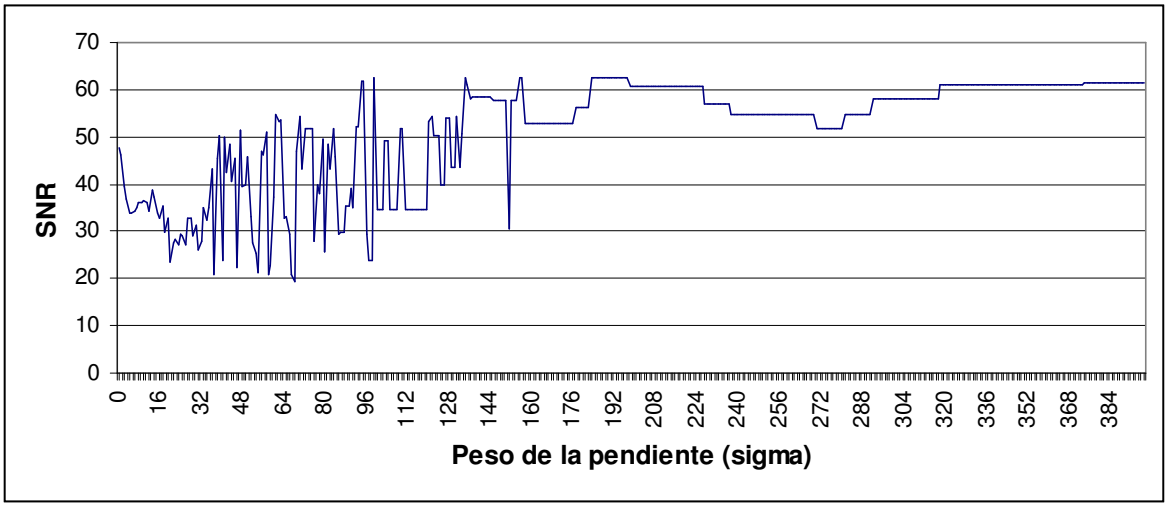


Figura 8.2. Comportamiento de la SNR al variar σ

Las figuras 8.3, 8.4 y 8.5 muestran la recuperación de la serie de tiempo *Lorenz* cuando se utiliza σ=12.95, σ=70.45 y σ=389.65 respectivamente.

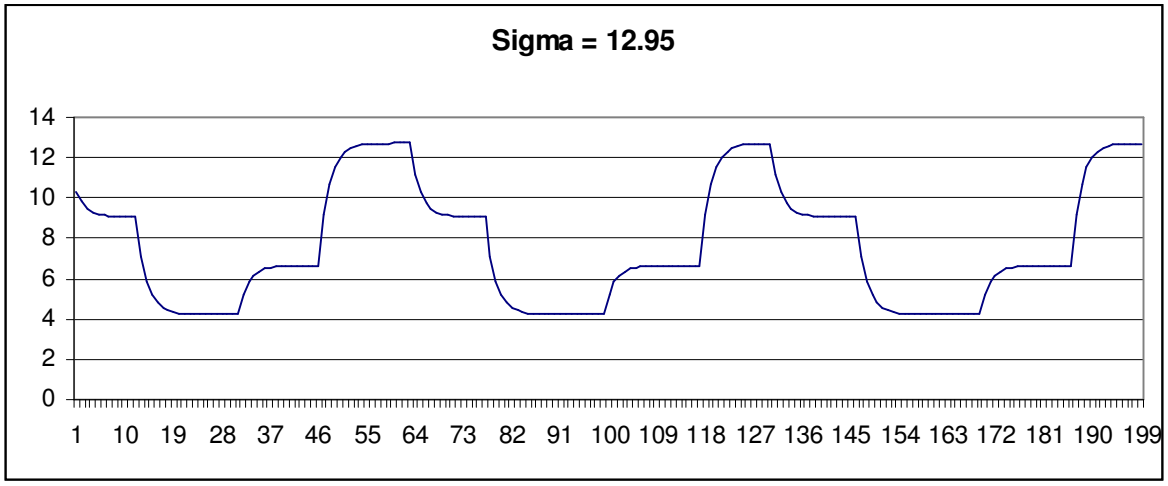


Figura 8.3. Recuperación de la serie *Lorenz* cuando se discretiza con σ = 12.95

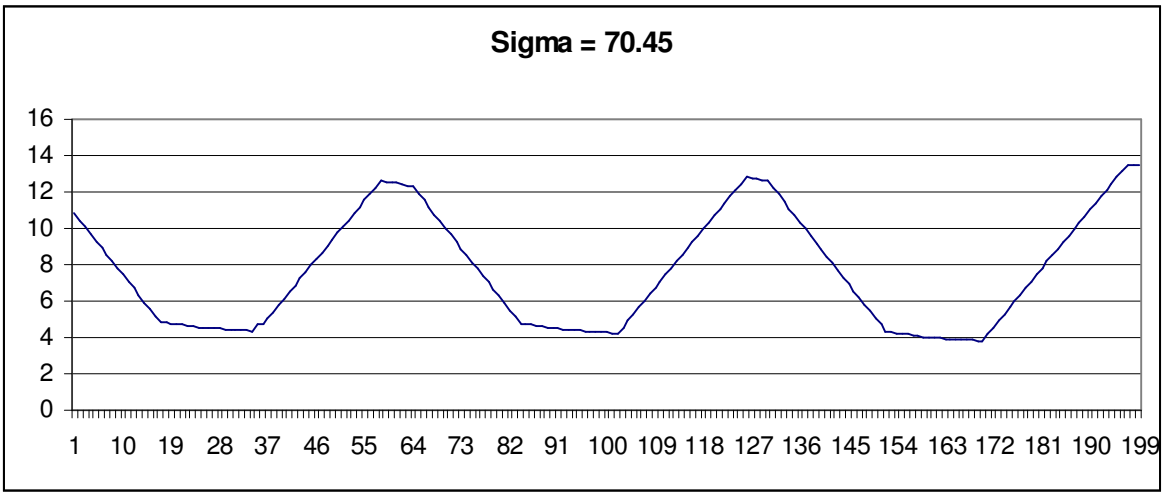


Figura 8.4. Recuperación de la serie *Lorenz* cuando se discretiza con σ = 70.45

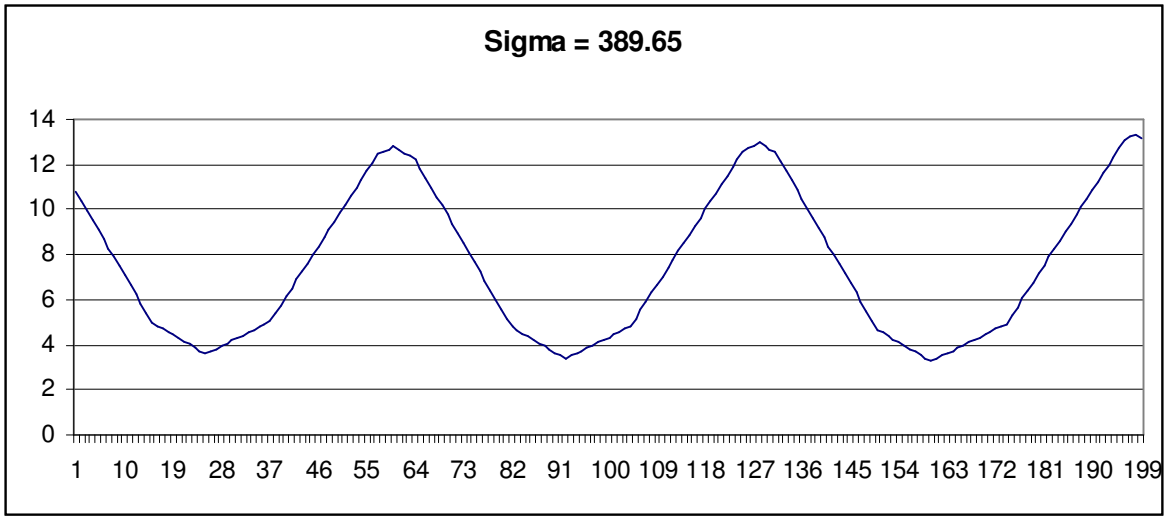


Figura 8.5. Recuperación de la serie *Lorenz* cuando se discretiza con σ = 389.65

Un valor pequeño para σ permite recuperar la serie de tiempo de manera poco precisa pero estable, es decir, el error es en ocasiones alto, pero permanece estable conforme aumenta el número de valores recuperados. Por el contrario, un valor de σ relativamente grande permite recuperaciones más precisas, con el costo de que, al recuperar cada punto respecto al anterior, la recuperación se desvía respecto a la serie original conforme aumenta el número de valores recuperados.

Además de σ, cuando se utiliza el algoritmo de tres etapas para la extracción de la Red Bayesiana, es necesario especificar un parámetro ε que influye sobre la conectividad del grafo subyacente a la estructura de la red. Para todos los experimentos mostrados se ha utilizado un valor igual al 20% de la información mutua existente entre las dos secuencias discretas mejor acopladas.

8.2 Confiabilidad de los resultados

Al elegir determinadas variables de entrada es posible que los algoritmos de extracción de Redes Bayesianas generen estructuras espurias, en cuyo caso los arcos entre variables no reflejarán relaciones de causalidad o dependencia. Sin embargo, es posible realizar un tipo de validación al recuperar una serie de tiempo asignando valores conocidos a otras variables que se muestran conectadas en la red. De esta manera, si la serie de tiempo recuperada no semeja a la esperada, esto indicará que la estructura obtenida no es del todo válida, o al menos no refleja relaciones suficientemente fuertes entre las variables.

Cabe hacer notar que una estructura de Red Bayesiana generada automáticamente por cualquier método está basada en los valores discretos, y por tanto solo es válida respecto a los valores de σ utilizados durante la discretización. Por tanto, al variar el valor de σ para alguna serie de tiempo, es de esperarse que se modifique la estructura de la red.

Por ejemplo, si se toman las series de tiempo *Actividad económica*, *IPC* y *Salarios industria manufacturera* (mostradas en las figuras 6.3, 6.5 y 6.22 respectivamente) y se discretizan de acuerdo al valor de σ mostrado en la tabla 8.1, al alinear las secuencias discretas y utilizar el método MLE se obtiene la estructura de la Red Bayesiana mostrada en la figura 8.6.

Serie de tiempo	σ
Actividad económica	0
IPC	48.046
Salarios industria manufacturera	0

Tabla 8.1. Valores de σ utilizados para la generación de la red mostrada en la figura 8.6

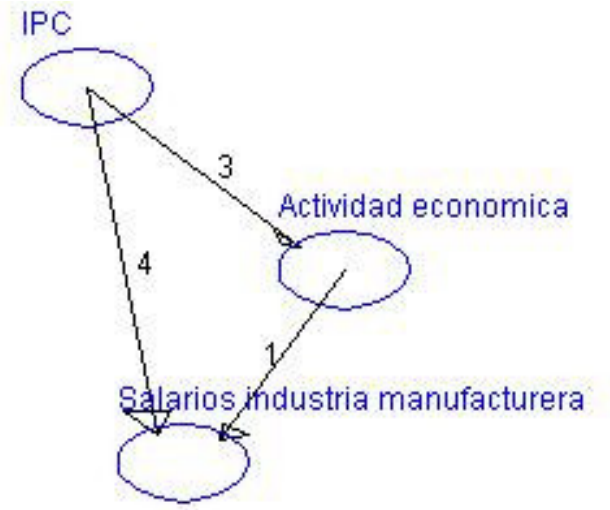


Figura 8.6. Estructura de la Red Bayesiana obtenida utilizando σ=0 para discretizar *Salarios industria manufacturera*

Con la Red Bayesiana mostrada en la figura 8.6 se recupera la variable *Salarios industria manufacturera* como se muestra en la figura 8.7.

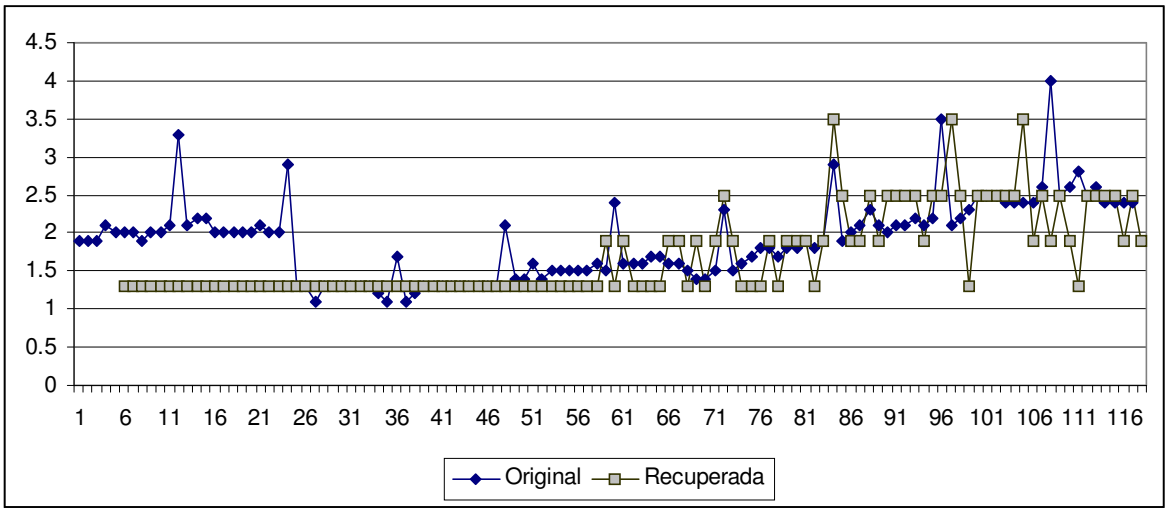


Figura 8.7. Recuperación de *Salarios industria manufacturera* a partir de la red de la figura 8.6 con σ=0

Si la variable *Salarios industria manufacturera* se discretiza dando al parámetro σ un valor de 215.55, respetando la estructura de la red mostrada en la figura 8.6, se obtiene la recuperación que se observa en la figura 8.8.

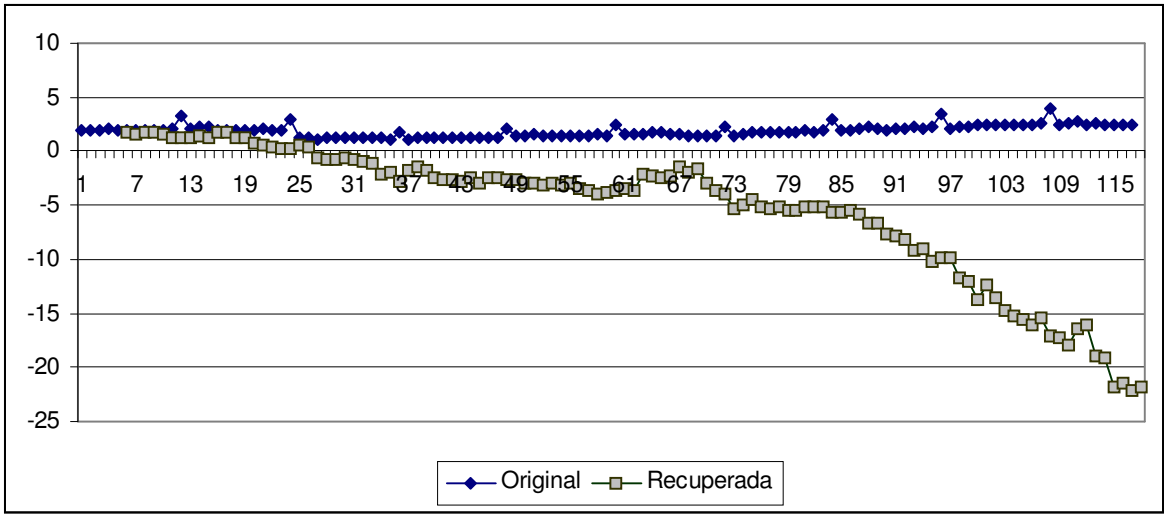


Figura 8.8. Recuperación de *Salarios industria manufacturera* a partir de la red de la figura 8.6 con σ=215.55

Aunque se nota que es posible utilizar la misma Red Bayesiana con diferentes parámetros de discretización, la calidad de los resultados disminuye debido a la pérdida de dependencia entre los datos discretizados. Al modificar el parámetro σ en *Salarios industria manufacturera*, se encuentra que la estructura más apropiada para representar a los datos cambia. En este caso, la estructura encontrada con el método MLE se muestra en la figura 8.9.

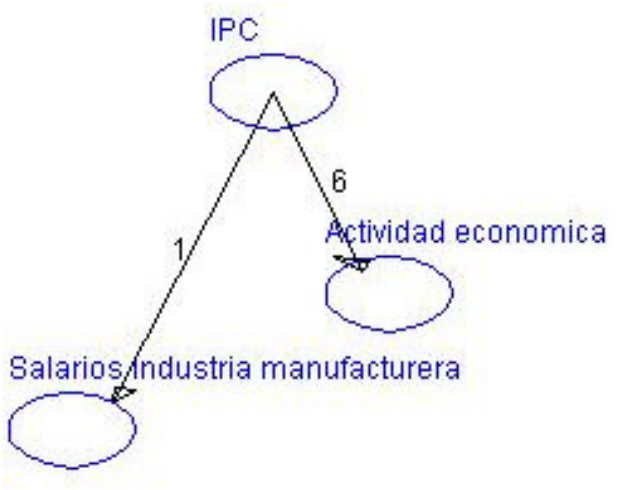


Figura 8.9. Estructura de la Red Bayesiana obtenida utilizando σ=215.55 para discretizar *Salarios industria manufacturera*

8.3 Función utilizada para la discretización

La función mostrada en la expresión 5.1 fue obtenida en base a la idea de distancia entre vectores (ver sección 5.2). Sin embargo, el método podría utilizar cualquier función que cumpla con las siguientes características:

1. El parámetro σ determina en que medida se debe tomar en cuenta la variación de cada punto de la serie de tiempo respecto al punto anterior. Conforme σ aumenta, la variación respecto al punto anterior toma mayor relevancia en la discretización.
2. Para cualesquiera dos duplas v_i y v_j, el valor de $d(v_i, v_j)$ es proporcional a la cercanía entre m_i y m_j y entre x_i y x_j.
3. $|x_i| < \infty, |x_j| < \infty, |m_i| < \infty, |m_j| < \infty, \sigma < \infty \Rightarrow |d(v_i, v_j)| < \infty$

La primera condición hace posible establecer el grado de importancia de la variación entre los puntos de la serie de tiempo. Es gracias a esta característica que el método permite relacionar variaciones de una serie de tiempo con la magnitud de los puntos de otra serie.

La segunda condición permite establecer el grado de cercanía entre dos duplas, de modo que sea posible agruparlas en una etapa posterior. Es posible que para una aplicación específica pudiera requerirse un comportamiento distinto (por ejemplo, establecer mayor cercanía entre aquellas duplas que se encontraran en cuadrantes opuestos), pero para el caso que nos ocupa es únicamente éste el comportamiento deseable.

La tercera condición se establece para fines prácticos, e impide la aparición de distancias infinitas para una serie de tiempo cuyos valores se encuentran acotados. Esta característica es importante debido a que, usualmente, este tipo de función involucra la división por

alguno de los valores, de modo que cuando la magnitud o la variación de una serie de tiempo fueran iguales a cero, la función generaría valores infinitos.

Conociendo las características que deben cumplir estas funciones, es posible dar algunos ejemplos de posibles alternativas:

1. $d(v_i, v_j) = \sigma(m_i - m_j)^2 + (x_i - x_j)^2$
2. $d(v_i, v_j) = |x_i - x_j| |m_i - m_j|^{\sigma}$

Para observar el comportamiento de estas funciones alternativas, se ha reconstruido el ejemplo mostrado en la figura 5.9. Utilizando el valor de σ empleado en ese ejemplo (σ =3254.5), se obtienen las discretizaciones mostradas en las figuras 8.10 y 8.11.

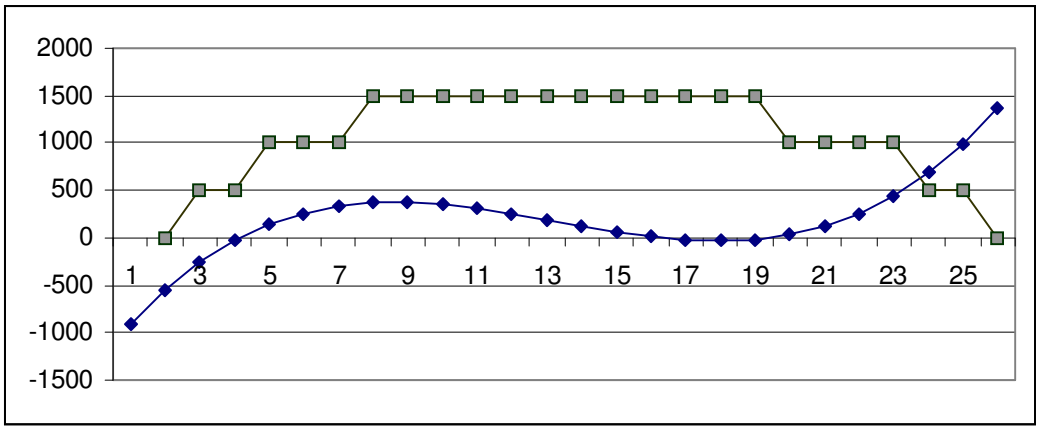


Figura 8.10. Discretización de $t^3+8t^2-44t+15$ utilizando la función alternativa 1 con σ=3254.5

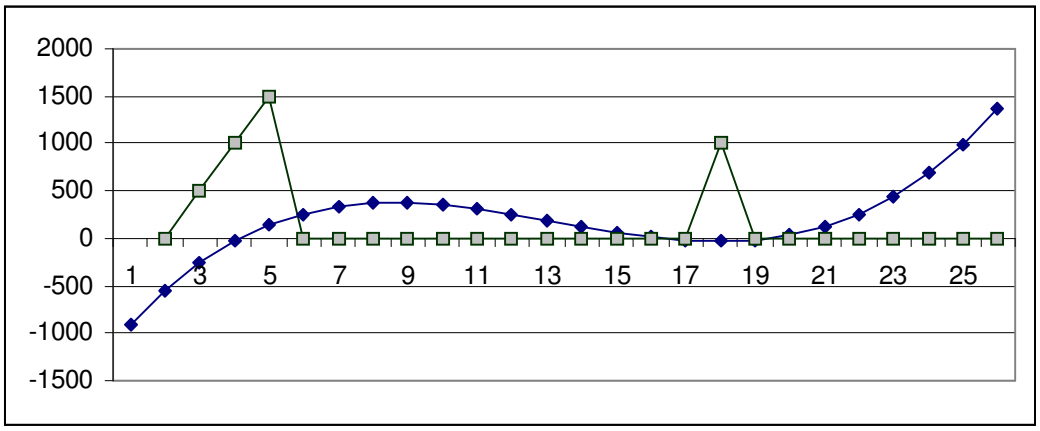


Figura 8.11. Discretización de $t^3+8t^2-44t+15$ utilizando la función alternativa 2 con σ=3254.5

La función alternativa 1 produce una discretización que depende principalmente de las variaciones de la serie de tiempo. La función alternativa 2 produce una discretización cuyo comportamiento es difícil de explicar. Esto se debe a que el valor de σ utilizado es demasiado grande cuando se aplica a estas nuevas funciones.

Utilizando la primera función alternativa con un valor de σ =135.75, se obtiene la discretización mostrada en la figura 8.12, la cual es muy similar a la obtenida con la función original.

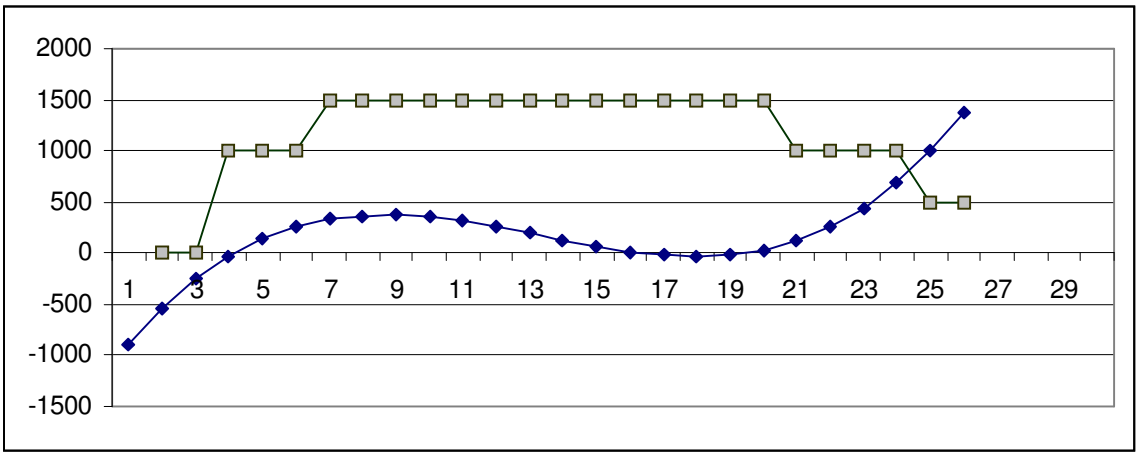


Figura 8.12. Discretización de $t^3+8t^2-44t+15$ utilizando la función alternativa 1 con σ=135.75

Utilizando la segunda función alternativa con σ =1.6 se obtiene la discretización que se muestra en la figura 8.13. Al observar detenidamente la gráfica, se observa que la discretización está basada principalmente en el valor absoluto de las variaciones de la serie de tiempo, y cuando este valor es muy similar en dos puntos diferentes, se utiliza la magnitud para colocar dichos puntos en diferentes agrupaciones.

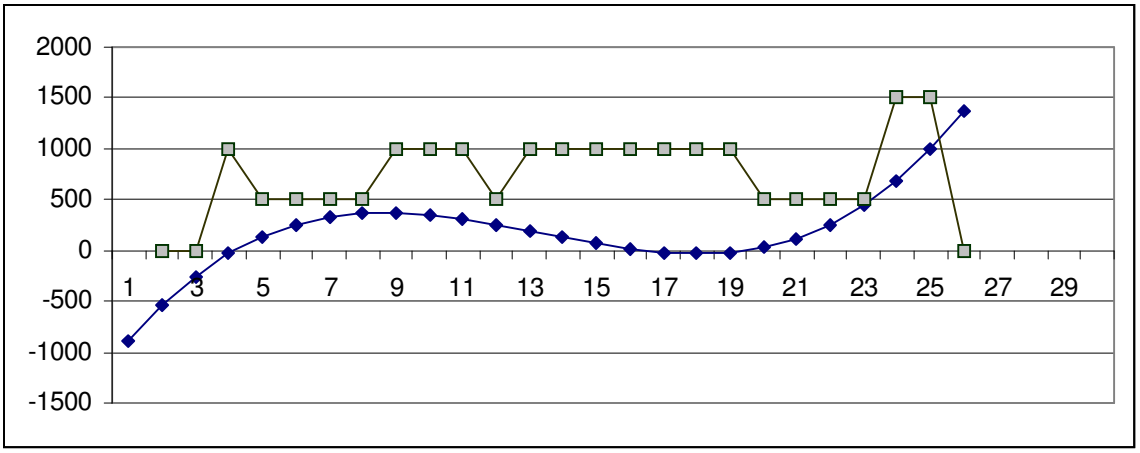


Figura 8.13. Discretización de $t^3+8t^2-44t+15$ utilizando la función alternativa 2 con σ=1.6

En lo que respecta a la calidad en la recuperación de la serie original, las funciones alternativas producen resultados menos satisfactorios. Por ejemplo, la tabla 8.2 muestra la calidad al recuperar una serie de tiempo seno (mostrada en la figura 5.10) cuando es discretizada utilizando la función alternativa 1. Realizando una búsqueda alrededor de σ =120 se encuentra que cuando σ =129.02 se obtiene una Relación Señal a Ruido de 41.7159dB, la cual es considerablemente menor a los 58.5993dB obtenidos con la función original.

σ	SNR (dB)
0.0	29.27717663003792
20.0	1.9717813332952472
40.0	30.63489810508095
60.0	12.84654717821107
80.0	13.933307882784609
100.0	38.63177940134824
120.0	39.801288178825615
140.0	34.7151601694268
160.0	38.74965346239288
180.0	24.524954889722462
200.0	32.509641236184876

Tabla 8.2. Relación Señal a Ruido obtenida con la función alternativa 1

Con la función alternativa 2 la diferencia es más notoria. La tabla 8.3 muestra la calidad de la misma serie al ser discretizada y recuperada utilizando esta función. Realizando una búsqueda se encuentra que cuando σ=11.72 la Relación Señal a Ruido es de 33.2151dB.

σ	SNR (dB)
0.0	29.27717663003792
20.0	-0.005097889714140031
40.0	11.637436106316546
60.0	13.023264469466248
80.0	11.94206841263483
100.0	5.863661815929488
120.0	-0.849759716511618
140.0	-28.20919532080866
160.0	-37.81350838025094
180.0	-36.39752287211126
200.0	-39.930012853004826

Tabla 8.3. Relación Señal a Ruido obtenida con la función alternativa 2

Al observar estos resultados se debe tener en mente que la etapa de recuperación de la serie de tiempo se pensó para la función original. Por lo tanto, es posible que exista otra forma de

recuperación con la cual se obtengan mejores resultados cuando se trabaja con las funciones alternativas.

8.4 Comparación con otros métodos

En la sección 4.2 se mencionaron algunos trabajos en los que se estudia la extracción de modelos en forma de grafos a partir de series de tiempo. En el apéndice A se muestran los resultados obtenidos por Correlation Metric Construction, el Grafo de Causalidad de Granger y el Grafo de Correlación Parcial. En esta sección se explican de manera más detallada los resultados obtenidos por estos métodos.

8.4.1. Correlation Metric Construction

Este método [Arkin et. al., 1997] permite colocar a las series de tiempo en un espacio de dos o tres dimensiones, de modo que sea posible observar la distancia relativa entre ellas. También proporciona gráficas en las que se muestra la correlación de cada serie con todas las demás, facilitando así el descubrimiento de la causalidad entre ellas.

Como ejemplo se retoman las series de tiempo utilizadas en la prueba 1, presentada en las secciones 6.3.1 y 7.3.1. Estas series de tiempo son: el Indicador Global de la Actividad Económica, la Tasa de Desempleo Abierto, el último valor para cada mes del Indice de Precios y Cotizaciones de la Bolsa Mexicana de Valores y el Indice Nacional de Precios al Consumidor. El modelo obtenido al generar el escalamiento multidimensional para dos dimensiones se muestra en la figura A.1 del apéndice A.

Correlation Metric Construction (CMC), similarmente al método propuesto en esta tesis, desplaza las series de tiempo hasta lograr la máxima coincidencia entre ellas. En la tabla 8.4 se indican los desplazamientos que producen la máxima correlación entre las series de tiempo. El desplazamiento indicado se aplica a la serie de tiempo a la que pertenece la fila. Por ejemplo, *Actividad económica* e *Inflación* presentan una correlación máxima cuando *Actividad economica* se deplaza 17 unidades de tiempo hacia adelante, es decir, cuando cualquier valor de *Inflación* coincide con el valor de *Actividad económica* que se encuentra 17 unidades de tiempo antes de él.

	Actividad económica	**Desempleo abierto**	**Inflación**	**IPC BMV**
Actividad económica	-	0	-17	-5
Desempleo abierto	0	-	-18	-12
Inflación	17	18	-	14
IPC BMV	5	12	-14	-

Tabla 8.4. Desplazamiento entre las series de tiempo

Una vez desplazadas las series de tiempo, CMC establece una medida de distancia basada en la cross-correlación entre las series de tiempo. En la tabla 8.5 se encuentran las distancias entre pares de series de tiempo, a partir de las cuales se obtuvo la figura A.1.

	Actividad económica	Desempleo abierto	Inflación	IPC BMV
Actividad económica	0	0.6682	0.1114	0.3539
Desempleo abierto	0.6682	0	0.5508	0.7550
Inflación	0.1114	0.5508	0	0.3194
IPC BMV	0.3539	0.7550	0.3194	0

Tabla 8.5. Distancias entre pares de series de tiempo

La figura 8.14 muestra la cross-correlación existente entre *Actividad económica* y las series de tiempo restantes.

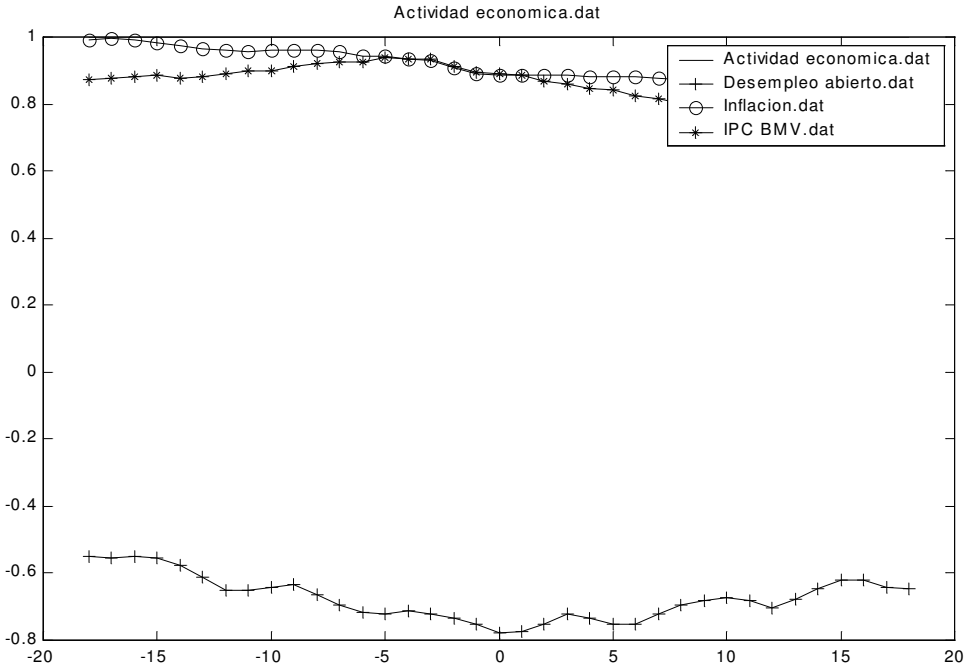


Figura 8.14. Cross-correlación entre *Actividad económica* y las demás series de tiempo

La figura 8.15 muestra la cross-correlación entre *Desempleo abierto* y las series restantes.

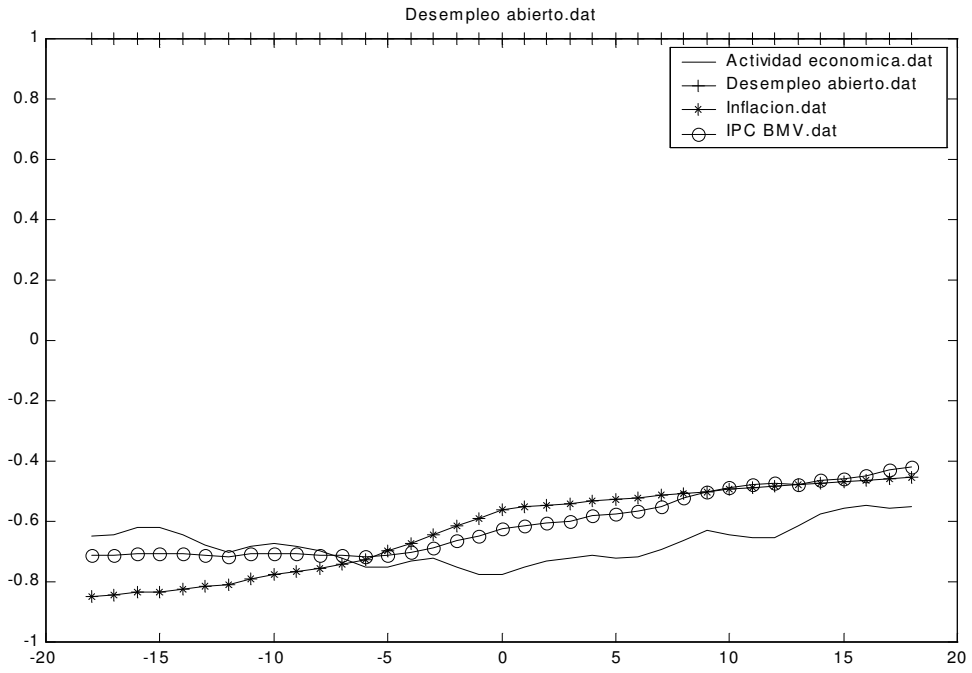


Figura 8.15. Cross-correlación entre *Desempleo abierto* y las demás series de tiempo

La figura 8.16 muestra la cross-correlación entre *IPC* y las series restantes.

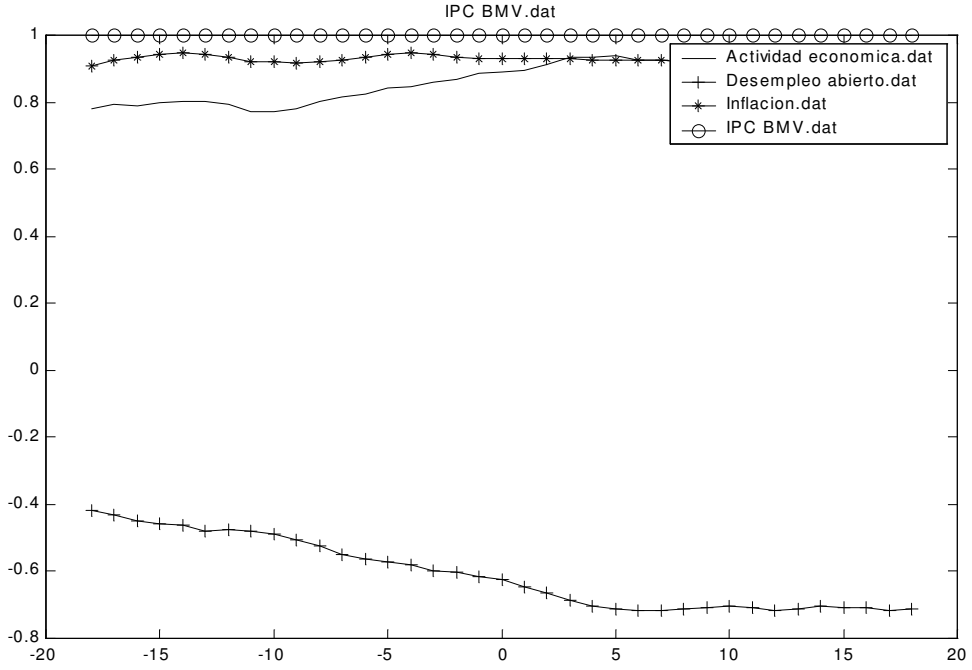


Figura 8.16. Cross-correlación entre *IPC* y las demás series de tiempo

La figura 8.17 muestra la cross-correlación entre *Inflación* y las series de tiempo restantes.

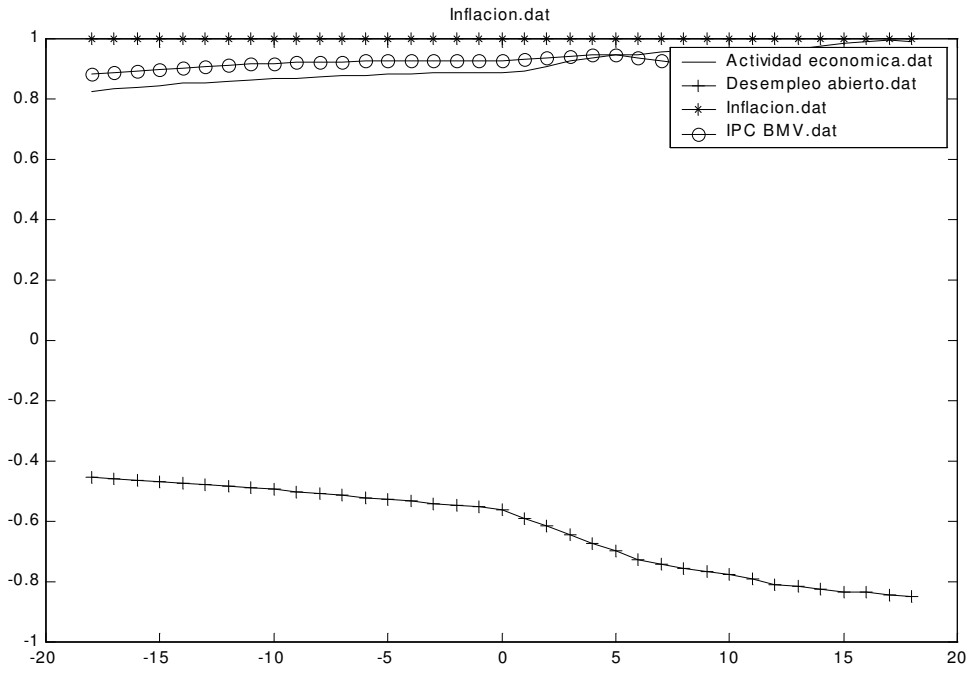


Figura 8.17. Cross-correlación entre *Inflación* y las demás series de tiempo

8.4.2. Grafos de Causalidad de Granger y de Correlación Parcial

Estos tipos de modelo [Dahlhaus & Eichler, 2000] han sido mencionados en la sección 4.2. Utilizando las cuatro series de tiempo presentadas durante la prueba 1 (sección 6.4.1), se han obtenido los grafos correspondientes, los cuales se muestran en las figuras A.2 y A.3 del apéndice A.

Durante la obtención de estos modelos, se generan diversas gráficas que reflejan el comportamiento de la relación entre las series de tiempo en el dominio de la frecuencia. Por ejemplo, la figura 8.18 muestra los espectros, cross-espectros y funciones conocidas como espectros de fase, de los cuales se obtiene el Grafo de Correlación Parcial. En esta figura, las gráficas corresponden, de izquierda a derecha y de arriba a abajo, a las series de tiempo *Actividad económica*, *Desempleo abierto*, *Inflación* e *IPC BMV*. Por ejemplo la gráfica que se encuentra en la esquina superior izquierda corresponde al espectro de la serie de tiempo *Actividad económica*, mientras que la que se encuentra justo debajo de ésta corresponde al cross-espectro entre *Actividad económica* y *Desempleo abierto*. La gráfica que se encuentra a la derecha del espectro de *Actividad económica* corresponde al espectro de fase entre *Actividad económica* y *Desempleo abierto*.

La figura 8.19 muestra las gráficas de coherencia, las cuales pueden ser útiles para discriminar arcos en el grafo de correlación parcial.

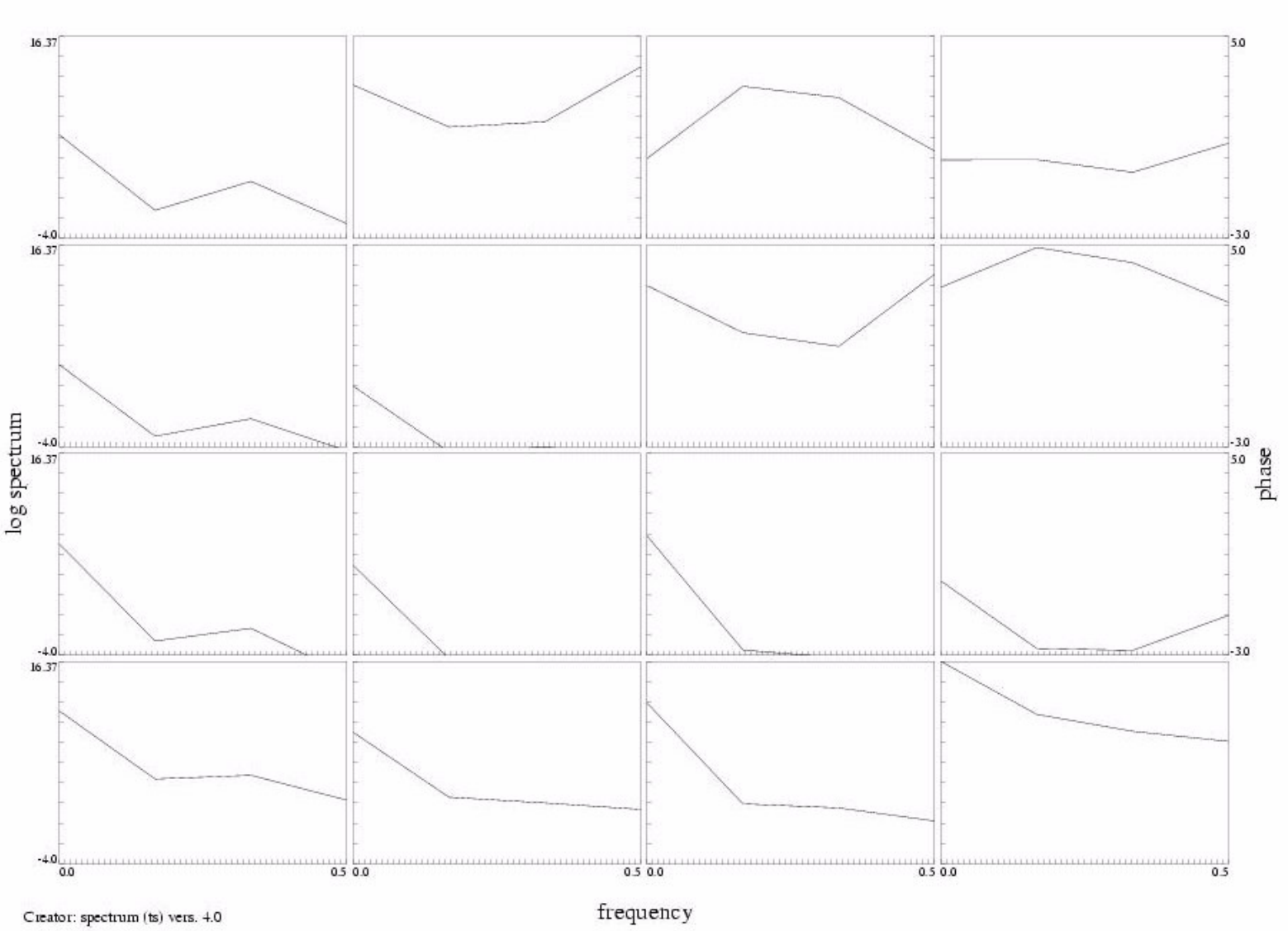


Figura 8.18. Espectros y cross-espectros para las series de tiempo de la prueba 1

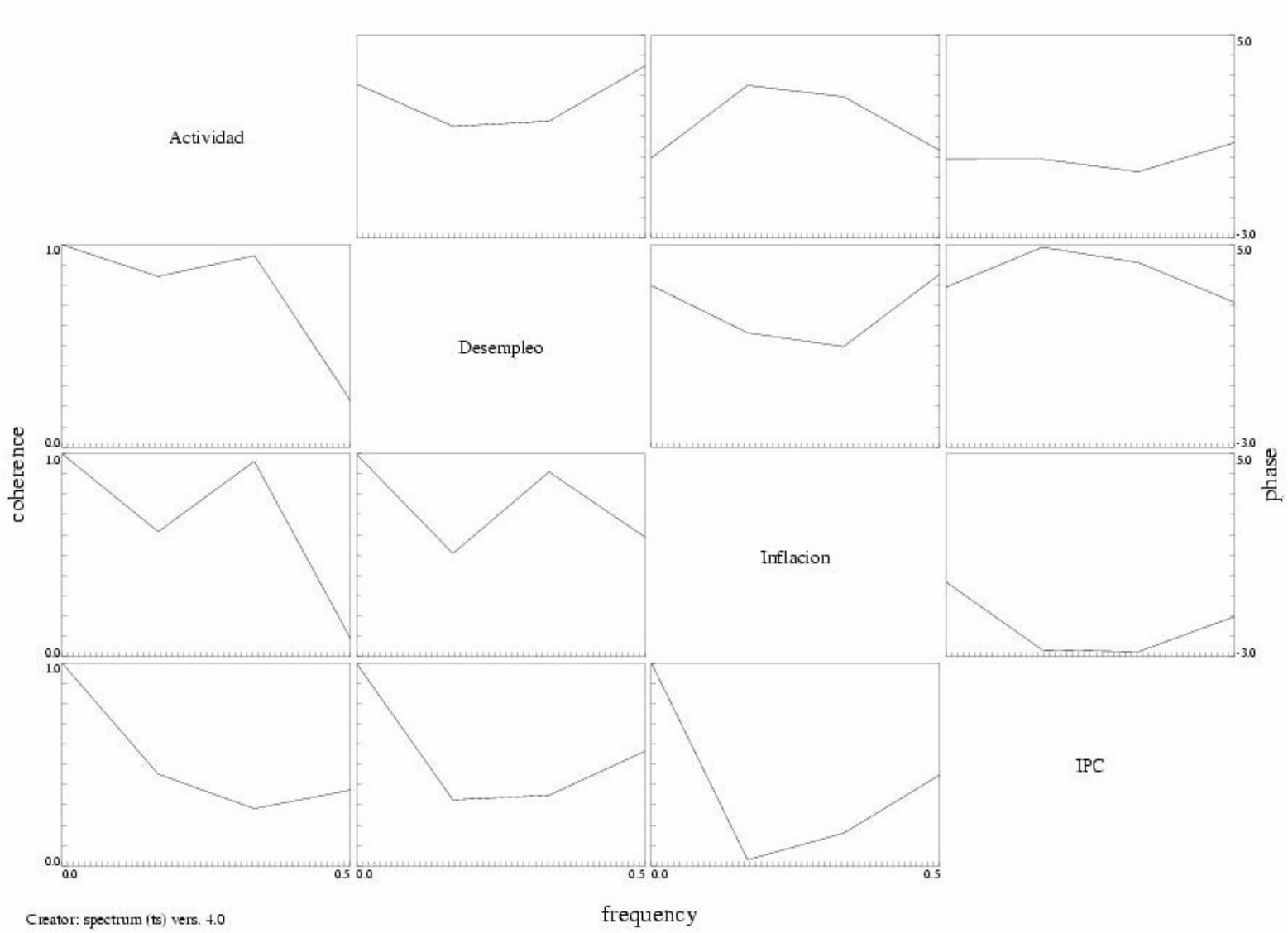


Figura 8.19. Gráficas de coherencia

9. IMPLANTACIÓN Y USO DE LA HERRAMIENTA

Las propuestas presentadas a lo largo de esta tesis han sido conjuntadas en una herramienta visual que permite la creación y/o extracción de Redes Bayesianas, así como la recuperación y predicción de las series de tiempo que les dieron origen. La herramienta permite la colaboración en la construcción y explotación de Redes Bayesianas mediante un ambiente distribuido.

En este capítulo se describe la arquitectura, los principales aspectos de diseño y la forma de uso de la herramienta.

9.1 Implantación en un ambiente distribuido

El modelo utilizado para la implantación de Redes Bayesianas Distribuidas está basado en dominios. Un dominio es un área de conocimiento delimitada que puede ser representada por un conjunto de variables semánticamente relacionadas, de modo que debe ser posible construir una Red Bayesiana consistente utilizando solamente las variables que se encuentran dentro de un dominio determinado. Sin embargo, algunas variables en diferentes dominios podrían estar relacionadas, creando así conexiones entre ellos.

Como ejemplo, considérense los dominios *Clima* y *Tránsito*. Se puede construir una Red Bayesiana para cada uno, conteniendo las variables de interés. La figura 9.1 muestra una conexión entre estos dominios. Si está lloviendo, la probabilidad de un accidente automovilístico aumenta, creándose así un arco entre la variable *Lluvia* y la variable *Accidente automovilístico*. Cuando se calcula la probabilidad de algún valor para la variable *Tráfico*, se deben considerar las variables en el dominio *Clima* [Medina & Figueroa, 2002].

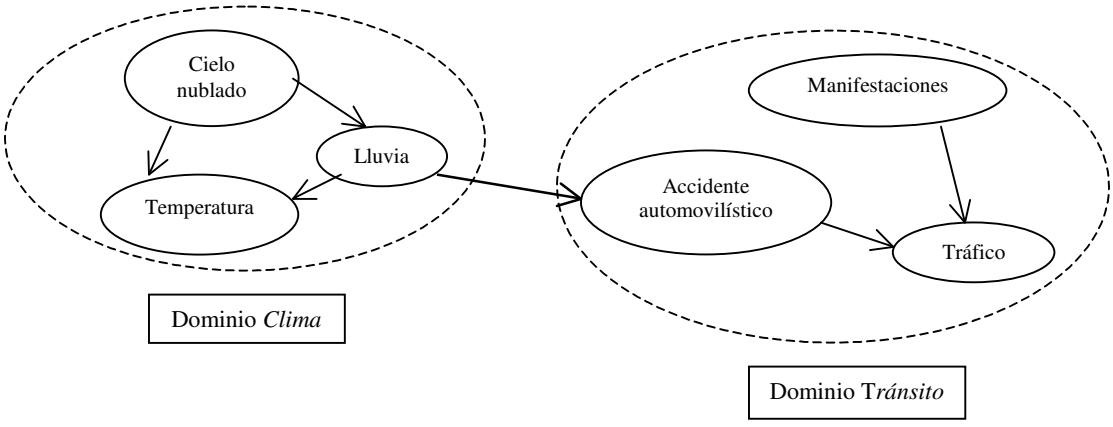


Figura 9.1. Relación entre los dominios *Clima* y *Tránsito*

La probabilidad del valor de una variable en la red se evalúa de acuerdo a un escenario dado. El escenario es el conjunto de valores que se asignan a las variables conocidas, de modo que la probabilidad resultante considere esta información. Si se omite el valor de una variable, el cálculo se realiza utilizando reglas de probabilidad.

La implantación de la herramienta en un ambiente distribuido permite la colaboración entre expertos de diferentes áreas para construir Redes Bayesianas más completas. Para cada grupo de colaboración que se encuentre construyendo redes relacionadas debe existir un repositorio de Redes Bayesianas. En este repositorio debe registrarse toda aquella red que desee compartir sus nodos, con la finalidad de que éstos puedan ser añadidos a otras redes que se encuentren en el mismo grupo, al tiempo que nodos de otras redes puedan ser utilizados en la propia red. Asimismo, es posible obtener una red del repositorio para trabajar de manera colaborativa con el usuario que la creó. Expresando la creación de una Red Bayesiana como un diagrama de casos de uso se obtiene la figura 9.2.

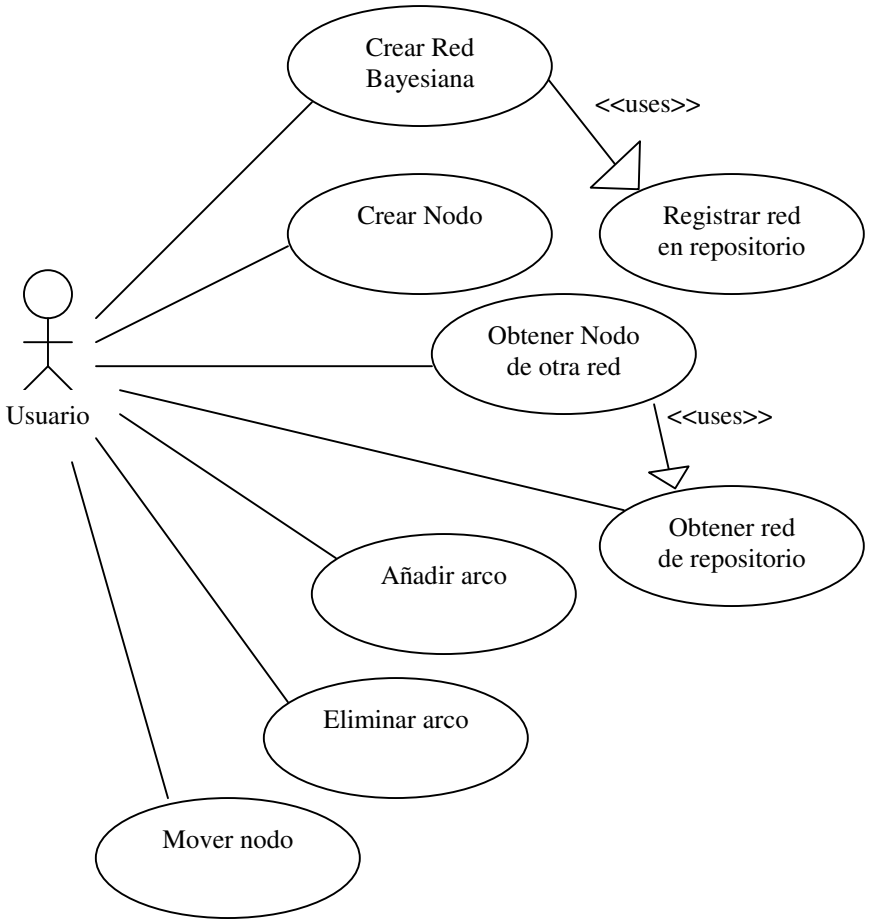


Figura 9.2. Algunos casos de uso en la creación de Redes Bayesianas

Cuando se agrega un nodo de una Red Bayesiana remota, es probable que el nodo añadido esté conectado con otros nodos en su red original. En este caso, los nodos a los que el nodo añadido esté conectado afectarán los cálculos de probabilidad en los que éste intervenga. Por consiguiente, se presentan las opciones de agregar únicamente el nodo elegido a la Red Bayesiana o agregar a todos aquellos nodos que estén conectados con el elegido. Llamaremos a aquellas redes en las que la agregación de nodos remotos se realiza como en el primer caso redes *limitadas*, mientras que a aquellas que se comportan como en el

segundo caso las llamaremos redes *completas*. El tipo de red, limitada o completa, se debe especificar al momento de su creación o de su obtención a partir del repositorio.

Toda Red Bayesiana que contenga un nodo determinado debe recibir notificaciones acerca de cualquier cambio visible relacionado con el mismo, tal como la adición o eliminación de algún arco, o el cambio de posición dentro de la red. Pensando en que podrían existir otros objetos, además de la Red Bayesiana, interesados en recibir notificaciones acerca de cambios en el nodo, se ha optado por utilizar el patrón de diseño *Observer* [Gamma et. al., 1995]. El diagrama colaboración para modificar un nodo se muestra en la figura 9.3.

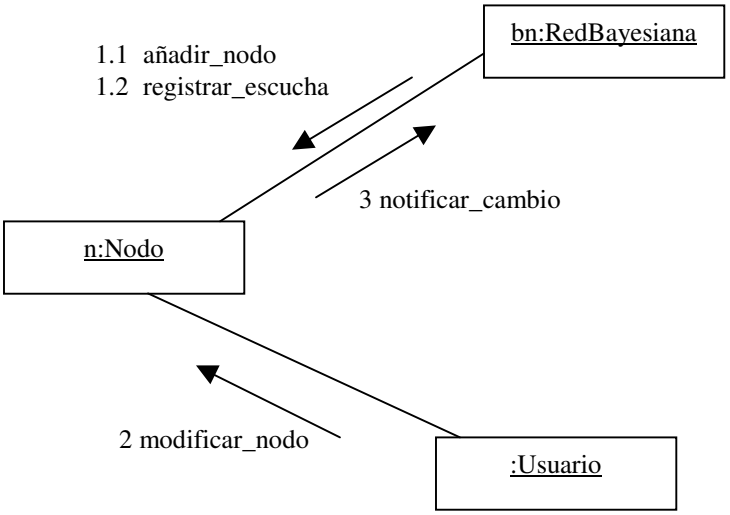


Figura 9.3. Diagrama de colaboración para la notificación de eventos de un nodo

El repositorio se crea con la primer Red Bayesiana. Todas las redes de creación posterior utilizarán ese repositorio, como se muestra en la figura 9.4.

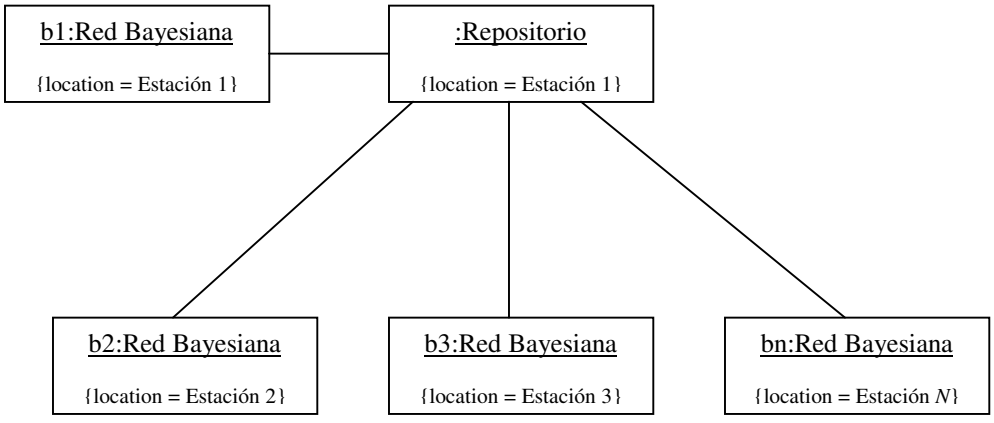


Figura 9.4. Diagrama de despliegue de las Redes Bayesianas y el repositorio

La herramienta se ha desarrollado en base a una metodología incremental iterativa, y su implantación se realizó utilizando objetos remotos bajo CORBA. Los objetos remotos que se utilizan son: *BayesNet*, *Node*, *DataNode*, *TSDataNode*, *Distribution*, *Scenario*, *BayesNetsRepository* y *BayesNetObserver*.

BayesNet. Este objeto representa una Red Bayesiana completa. Contiene los nodos del dominio, y es responsable de aquellas tareas que involucran a toda la red, tales como calcular la probabilidad de un nodo dado dentro de la red, o encontrar los nodos raíz (aquellos nodos que no tienen padres). Este objeto también es responsable de generar y recibir notificaciones acerca de eventos que sucedan en la red, tales como la adición o eliminación de un nodo, o la adición o eliminación de un arco entre dos nodos. Toda Red Bayesiana en el sistema es identificada por un nombre único proporcionado por el usuario. Además del nombre, la red mantiene una descripción de si misma, la cual puede servir como ayuda para seleccionar una red entre un conjunto.

Node. El objeto *Node* representa una variable. Cada nodo mantiene referencias a todos sus nodos padre e hijo, además de una referencia a su densidad de probabilidad, su posición dentro de la red y su nombre. Cada nodo es identificado por un nombre único definido por el usuario.

DataNode. Este objeto se deriva de *Node* para permitir el manejo de datos en el nodo. Entre sus métodos se encuentran algunos relacionados al conteo de datos, así como un método para la obtención de distribuciones de probabilidad a partir de datos.

TSDataNode. Es una extensión de *DataNode* que agrega métodos para trabajar con datos provenientes de series de tiempo. Los métodos de este objeto se refieren principalmente a la discretización de la serie de tiempo.

Distribution. El objeto *Distribution* representa la distribución probabilidad de una variable. Dado que esta distribución de probabilidad depende del valor de los padres de dicha variable, este objeto mantiene una asociación entre cada combinación de valores de los padres y la probabilidad para cada posible valor de la variable. Las operaciones se realizan en base a un escenario dado.

Scenario. El objeto *Scenario* se utiliza como una asociación entre nombres y valores. Se emplea principalmente como una interfaz para realizar operaciones sobre el objeto *Distribution*.

BayesNetsRepository. El repositorio de Redes Bayesianas es el objeto que almacena referencias y descripciones de toda Red Bayesiana que se encuentre registrada, de modo que debe existir exactamente un repositorio por cada grupo de trabajo. Cada red es identificada por su nombre.

BayesNetObserver. En ocasiones, los objetos locales tales como interfaces de usuario necesitan recibir información acerca de eventos que suceden en la Red Bayesiana para actualizar su estado. Los objetos locales no pueden registrarse con el objeto *BayesNet*, debido a que no pueden ser parámetros en una invocación remota, por lo que es necesario

definir un objeto remoto que escuche los eventos de *BayesNet*, y cuya implantación del servant pueda registrar a los objetos locales interesados en recibir notificaciones. Los eventos recibidos por *BayesNetObserver* son simplemente retransmitidos a los objetos locales.

Las clases del sistema se han organizado dentro de nueve paquetes, como se muestra en la figura 9.5.

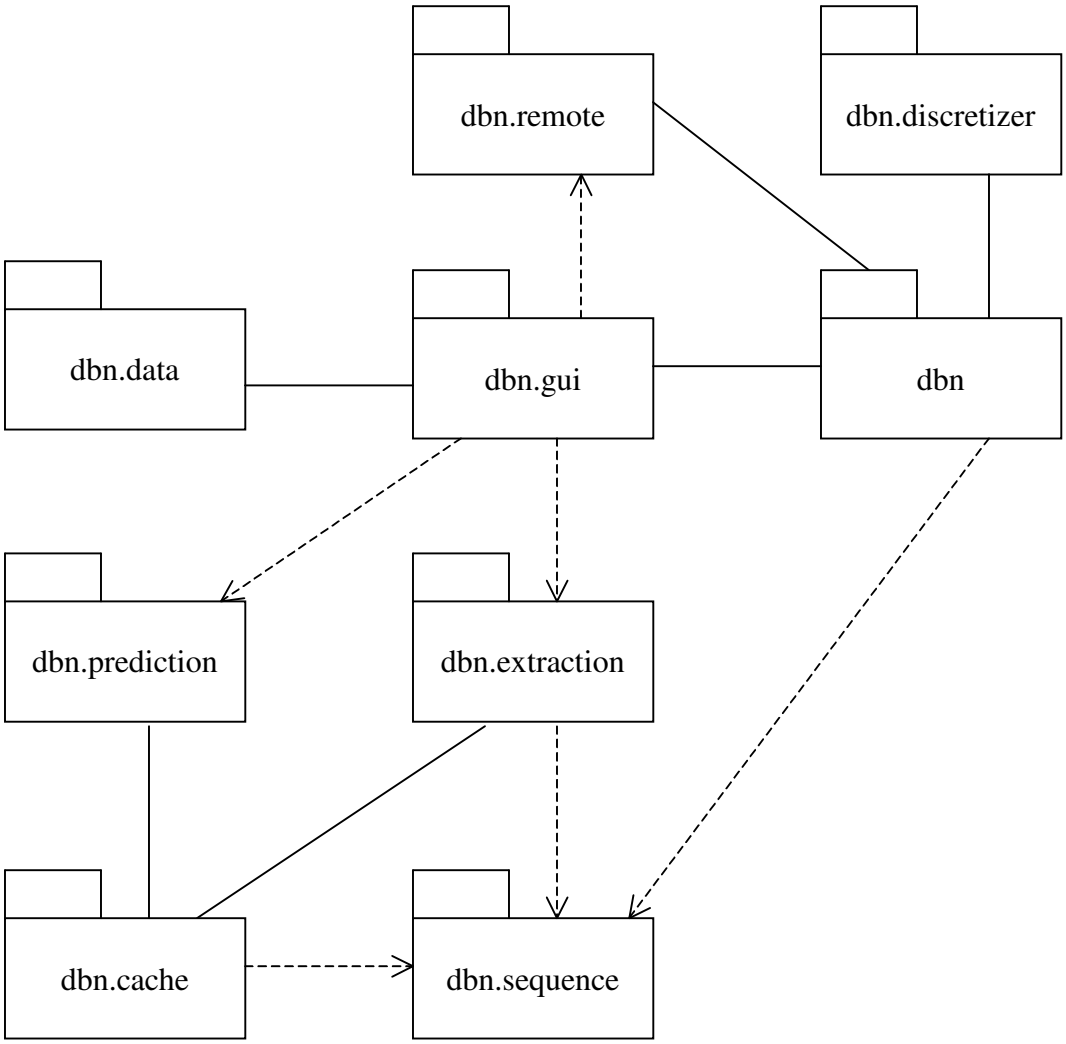


Figura 9.5. Paquetes en los que se organizan las clases

dbn. Es el paquete que contiene a todos los demás. Además, contiene todas las clases servant que implementan las interfaces remotas tales como *Node*, *Distribution*, etc.

La figura 9.6 muestra un diagrama de clases en el que se observa la relación entre las interfaces remotas y sus clases servant. Por simplicidad, se ha expresado la relación entre la interfaz remota y su servant como una implementación. Sin embargo, se sabe que en

realidad, para cada interfaz remota, se genera un adaptador del cual hereda el correspondiente servant. Como se ha explicado antes, todas las interfaces remotas se encuentran en el paquete *dbn.remote*, y los servants se encuentran en el paquete *dbn*.

La clase central en la figura 9.6 es *RemoteBayesNet*, el servant de la interfaz remota *BayesNet*. Este servant mantiene el control sobre los nodos de la red y se encarga de realizar cálculos en la misma, para lo cual hace uso de la interfaz remota *Scenario*.

Note que, aun cuando las interfaces remotas *Node*, *DataNode* y *TSDataNode* se encuentran en una jerarquía de herencia, sus respectivos servants *RemoteNode*, *RemoteDataNode* y *RemoteTSDataNode* no se encuentran en la misma situación. Esto se debe a la incapacidad del lenguaje Java para manejar herencia múltiple (recuerde que, en realidad, todos los servants heredan de un adaptador generado por CORBA). Para suplir la funcionalidad proporcionada por la herencia se utilizó el patrón de diseño *Decorator* [Gamma et. al., 1995].

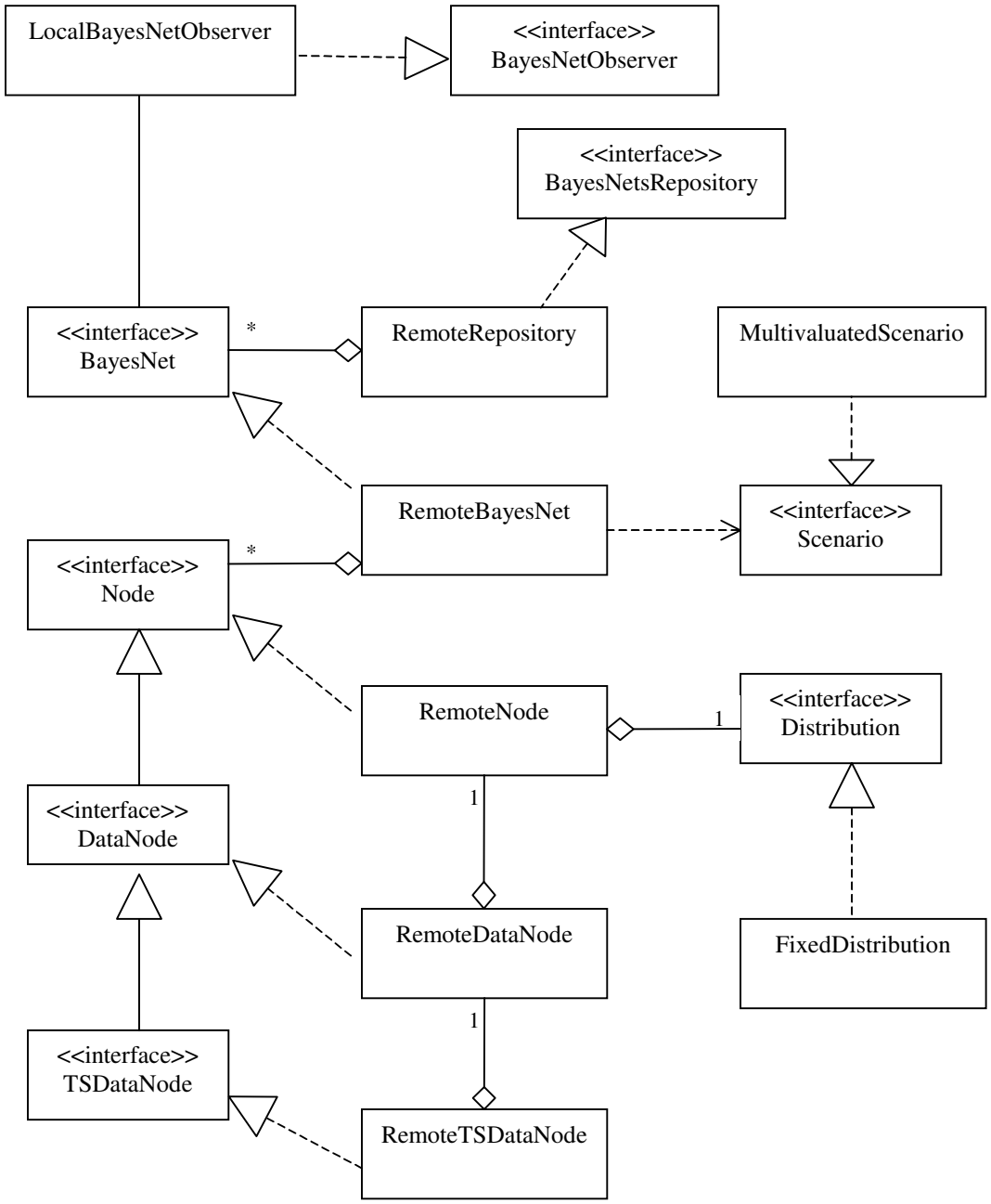


Figura 9.6. Diagrama de clases de las interfaces remotas y sus servants

dbn.cache. Contiene clases que implementan parte del funcionamiento de los objetos remotos de manera local, a fin de mejorar el rendimiento cuando se realiza un gran número de operaciones. Los objetos se crean copiando el estado de algún objeto remoto.

La figura 9.7 muestra el diagrama de clases para este paquete. En este caso, la implementación de las interfaces remotas se realiza de la manera usual, y no de manera indirecta como en el caso de los servants.

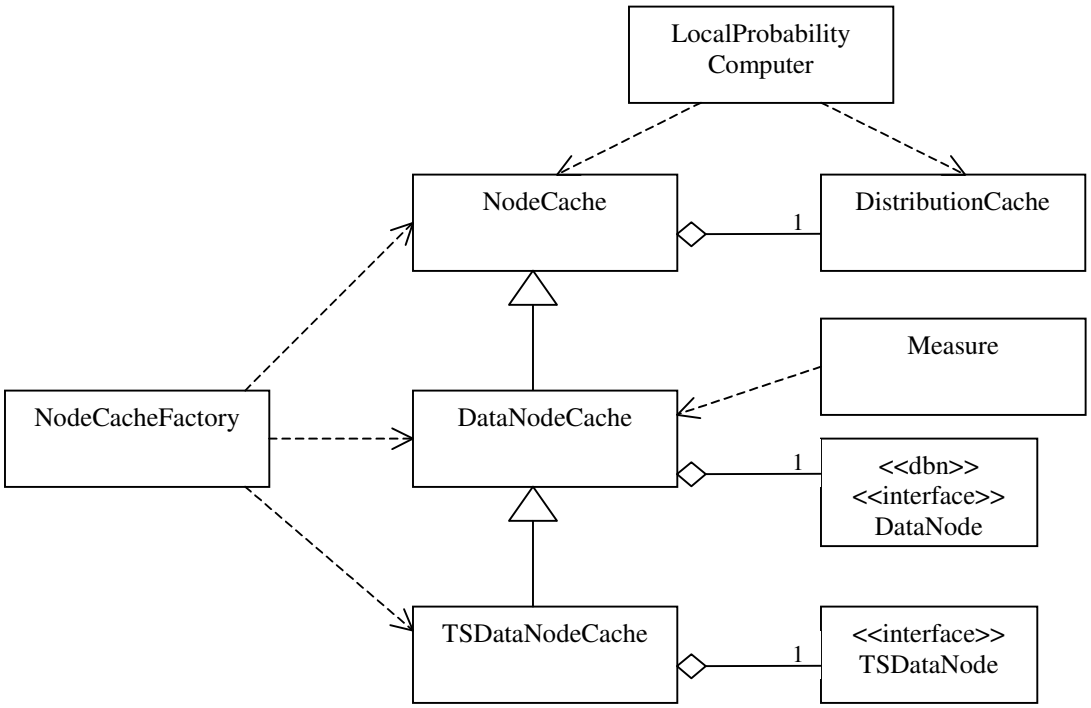


Figura 9.7 Diagrama de las clases contenidas en el paquete *dbn.cache*

dbn.data. Contiene las clases e interfaces utilizadas para la obtención de datos, ya sea a partir de archivos de texto, bases de datos, etc.

La figura 9.8 muestra las clases contenidas en este paquete. Una gran parte de éstas es utilizada por clases de otros paquetes que requieren obtener o almacenar información en un formato específico. La clase *DataDiscretizer* es un caso especial, ya que para realizar su trabajo requiere la utilización de clases del paquete *dbn.discretizer*.

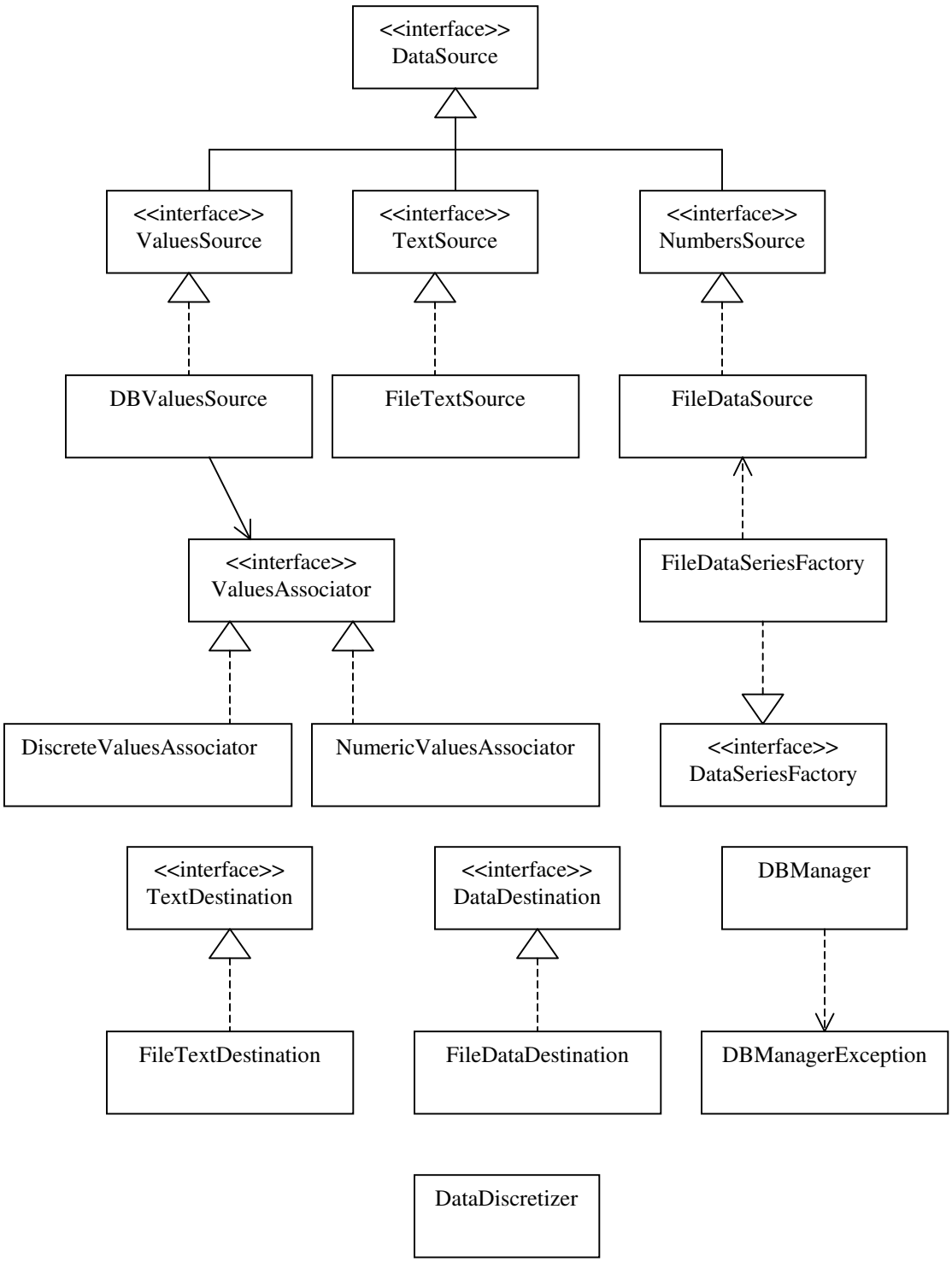


Figura 9.8 Diagrama de las clases contenidas en el paquete *dbn.data*

dbn.discretizer. Contiene las clases que implementan la discretización basada en vectores. La figura 9.9 muestra las clases contenidas en este paquete. Las clases *AbsoluteDistance*, *DefaultDistance* y *SquaredDistance* son implantaciones de distintas funciones de discretización, las cuales se explican en la sección 8.3.

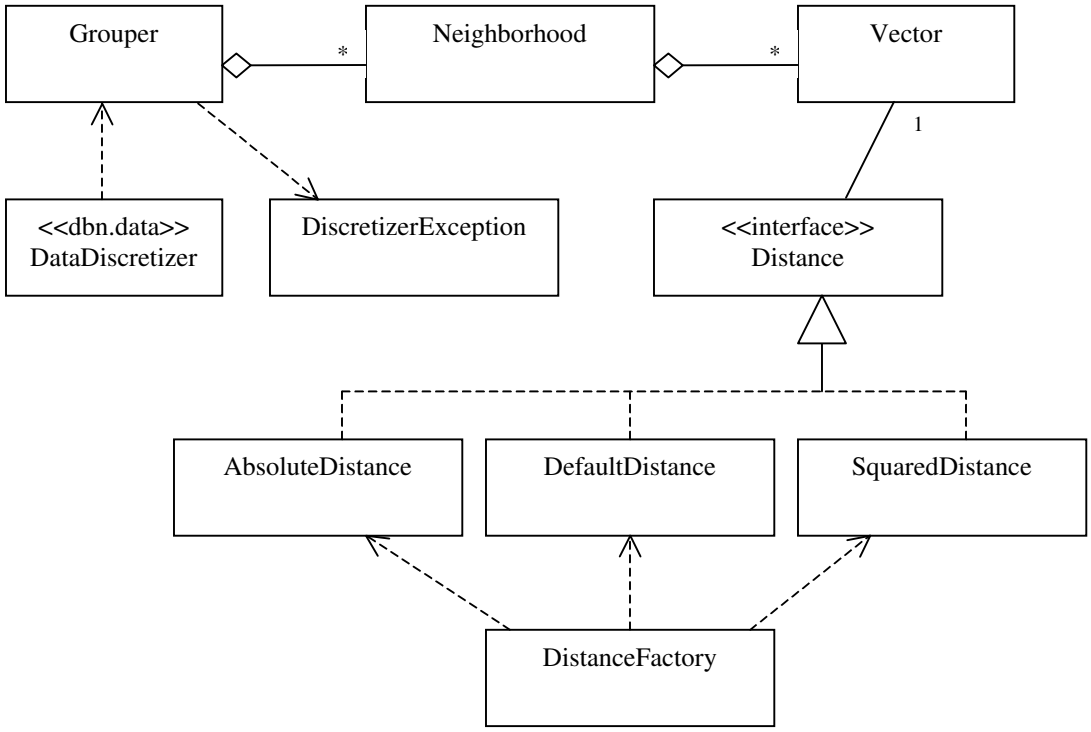


Figura 9.9 Diagrama de las clases contenidas en el paquete *dbn.discretizer*

dbn.extraction. En este paquete se encuentran las clases que se utilizan para alinear las secuencias discretas y extraer la estructura de la Red Bayesiana a partir de éstas. Debido a la gran cantidad de operaciones necesarias, las clases de este paquete utilizan clases del paquete *cache* en vez de utilizar objetos remotos. La figura 9.10 muestra las clases involucradas.

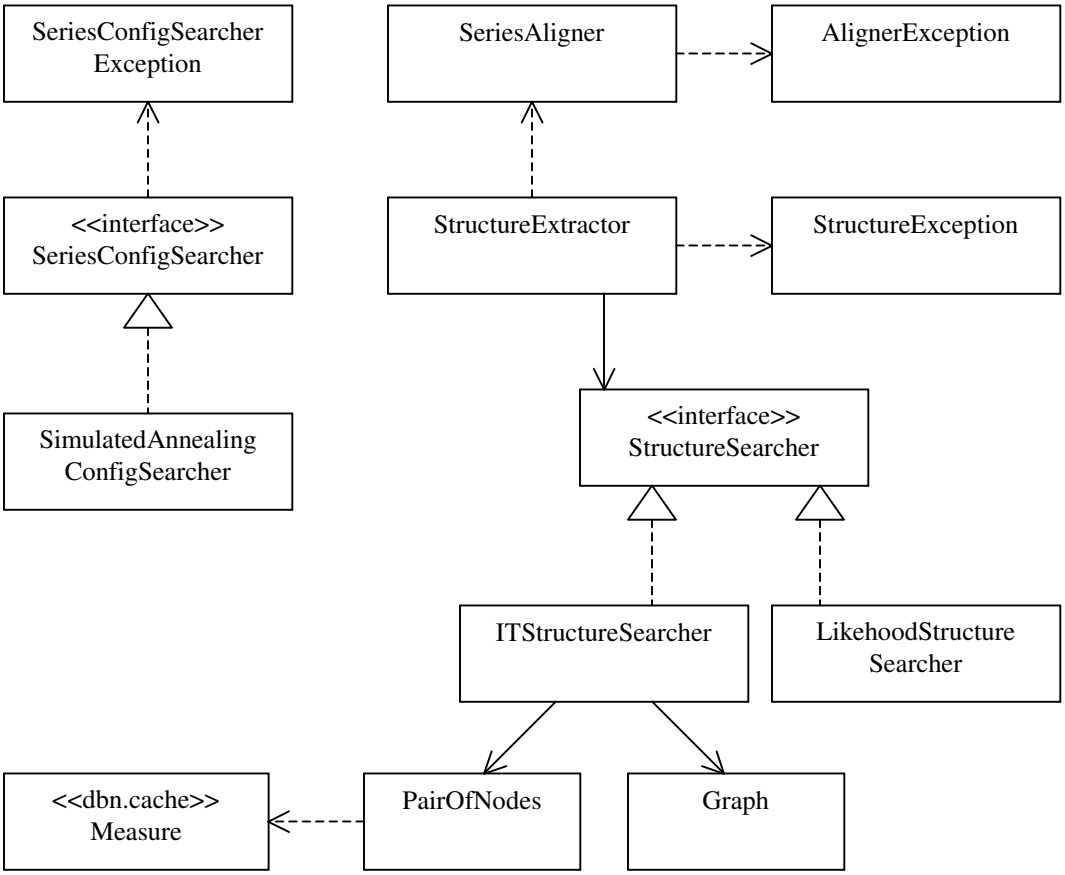


Figura 9.10 Diagrama de las clases contenidas en el paquete *dbn.extraction*

dbn.gui. Es el paquete que contiene todas las clases necesarias para generar la interfaz de usuario. Las clases más importantes de este paquete son *SystemWindow* y *SystemController*, que implementan la pantalla principal y su funcionamiento respectivamente. La figura 9.11 muestra algunas clases de este paquete.

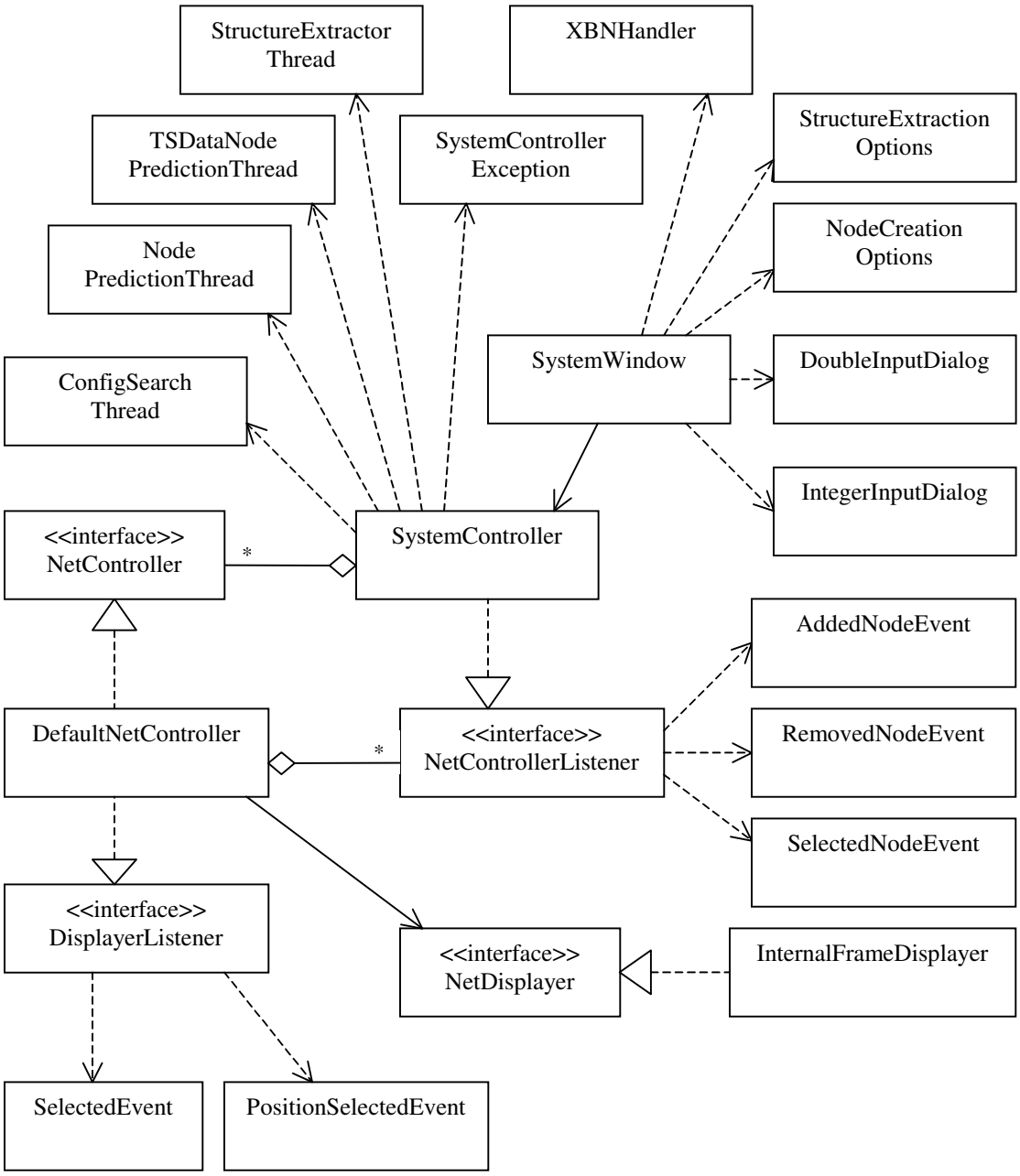


Figura 9.11 Diagrama de las clases contenidas en el paquete *dbn.gui*

dbn.prediction. Contiene las clases que se utilizan para recuperar una serie de tiempo a partir de una Red Bayesiana. Las clases de este paquete utilizan objetos de las clases que se encuentran en el paquete *cache* en vez de utilizar objetos remotos. La figura 9.12 muestra las clases involucradas.

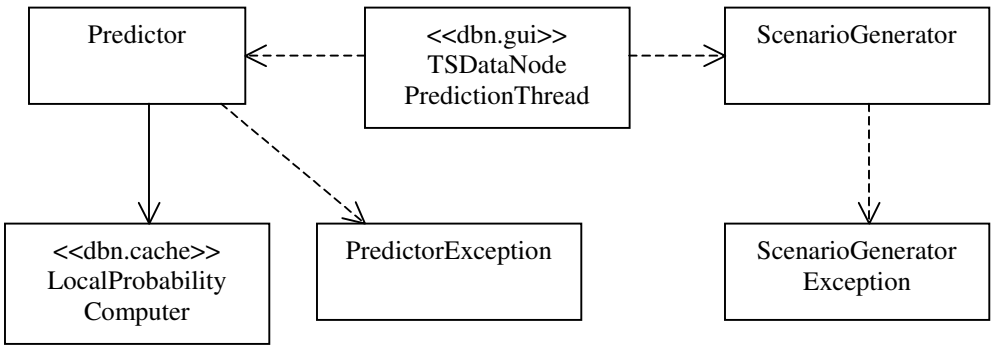


Figura 9.12 Diagrama de las clases contenidas en el paquete *dbn.prediction*

dbn.remote. Es el paquete que contiene todas las interfaces remotas, stubs, esqueletos, etc. generados a partir de la definición de las interfaces en IDL, y necesarios para generar el ambiente distribuido. Las interfaces remotas se mostraron en la figura 9.6.

dbn.sequence. Contiene clases que se utilizan para generar secuencias de valores en un arreglo. Por ejemplo, algunas clases podrían generar todos los valores del tipo 0 0 0 0, 0 0 0 1, 0 0 0 2, ..., 0 0 0 7, 0 0 1 0, 0 0 1 1, ..., 7 7 7 7. La figura 9.13 muestra las clases pertenecientes a este paquete.

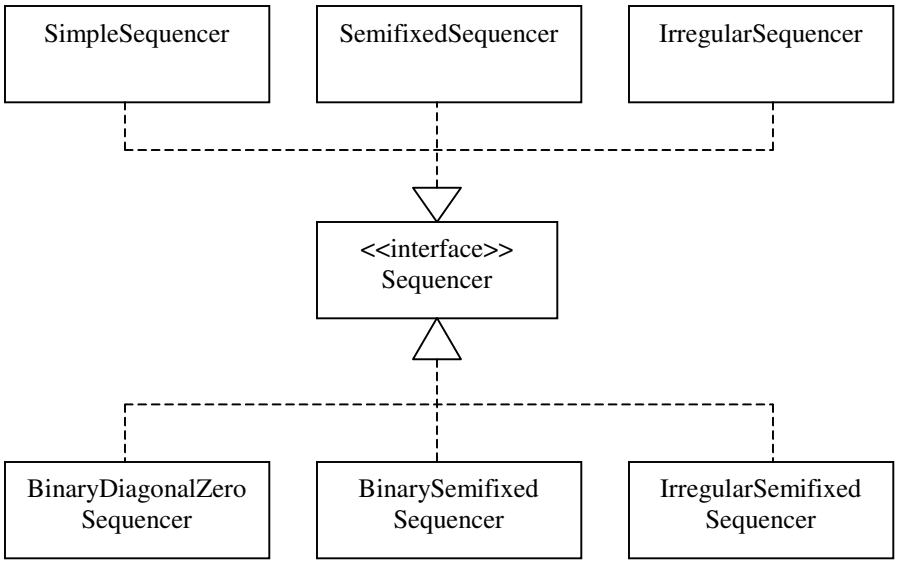


Figura 9.13 Diagrama de las clases contenidas en el paquete *dbn.sequence*

9.2 Instalación y uso de la herramienta

La herramienta ha sido completamente programada en Java, por lo que para su ejecución es necesario contar con el entorno de ejecución JRE (Java Runtime Environment) en su versión 1.4.2_05 o superior.

9.2.1 Instalación

Para instalar bajo Windows:

Descomprima el archivo *dbn.zip* en el directorio de su preferencia. Se creará un subdirectorio llamado *dbn*.

Establezca la variable de ambiente *JAVA_HOME*, asiganando el directorio en el que se encuentra instalado el entorno de ejecución de Java (por ejemplo, *j2sdk1.4.2_05*). En este directorio se debe encontrar un subdirectorio *bin*, que contiene archivos ejecutables, tales como *java.exe* e *tnameserv.exe*. La manera de establecer una variable de ambiente varía de acuerdo a la versión de Windows. En algunos casos (Windows 95, 98) se agregaría la una línea similar a:

```
SET JAVA_HOME=C:\j2sdk1.4.2_05
```

al archivo *AUTOEXEC.BAT*, mientras que en otros casos (Windows 2000, XP) es necesario acceder al panel de control y seleccionar *Sistema*, seleccionar el panel *Avanzado* y dar clic en el botón *Variables de entorno*.

Para instalar bajo Linux:

Descomprima el archivo *dbn.zip* mediante el comando

```
unzip dbn.zip –d <DIR>
```

en donde **<*DIR*>** es el directorio en el cual se desea descomprimir los archivos. Se creará el subdirectorio *dbn*.

Establezca la variable de ambiente *JAVA_HOME*, asignando el directorio en el que se encuentra instalado el entorno de ejecución de java. La manera de hacer esto depende del tipo de shell con el que se esté trabajando.

- Si está utilizando csh o tcsh, agregue una línea similar a:

  ```
  setenv JAVA_HOME /usr/java/j2sdk1.4.2_05
  ```

  al archivo *.cshrc*.
- Si está utilizando sh o ksh, agregue una línea similar a:

  ```
  export JAVA_HOME=/usr/java/j2sdk1.4.2_05
  ```

  al archivo *.profile*.

9.2.2 Inicio de la herramienta

Para utilizar una o más instancias de la herramienta, debe encontrarse en el directorio *dbn* dentro de <*DIR*>. El comando a utilizar depende de si se desea crear un Repositorio de Redes Bayesianas o utilizar uno existente. Recuerde que el repositorio solamente se debe crear con la primera red de un grupo de trabajo y las demás redes deben conectarse precisamente a ese repositorio. Además, tome en cuenta que para crear varios repositorios es necesario utilizar un puerto diferente para cada uno.

El comando más sencillo para iniciar la herramienta bajo Windows es:

```
dbn -r
```

Bajo Linux, el comando sería muy similar:

```
./dbn.sh -r
```

Este comando inicia una instancia del servidor de nombres de CORBA y Red Bayesiana, creando un repositorio que utiliza el puerto predeterminado (900). Se puede utilizar cuando se desee ejecutar la herramienta localmente, o cuando ésta sea la primera red de un grupo de trabajo y el grupo haya acordado utilizar el puerto predeterminado.

Para iniciar una red creando un repositorio que utilice otro puerto, se utiliza el comando

```
dbn -r <Puerto>
```

En caso de que no se desee crear un repositorio sino utilizar uno remoto, debe existir una instancia de Red Bayesiana con la cual se haya creado un repositorio. Esta instancia se debe encontrar en una estación cuyo nombre será referido en adelante como <*Servidor*>.

Si el repositorio en <*Servidor*> utiliza el puerto predeterminado (900), el comando para iniciar la instancia de la red que utilice dicho repositorio es:

```
dbn -j <Servidor>
```

En caso de que el repositorio remoto utilice otro puerto (al que nos referimos como <*Puerto*>), el comando para iniciar la instancia de la red es:

```
dbn -j <Servidor> <Puerto>
```

9.2.3 Uso de la herramienta

Una vez iniciada la herramienta se presentará una pantalla que en una barra presenta los siguientes menús:

- ***System.*** Comandos relacionados con el manejo de Redes Bayesianas y la herramienta en general. Dentro de este menú se encuentran:

  - ***New.*** Creación de una Red Bayesiana
    - ***Bound bayes net.*** Creación de una Red Bayesiana en la que, al agregar un nodo remoto, se añade únicamente éste y no los nodos que estén conectados a él en la red remota.
    - ***Complete bayes net.*** Creación de una Red Bayesiana en la que, al agregar un nodo remoto, se añaden también los nodos que estén conectados a él en la red remota.
  - ***Open bayes net.*** Obtiene una Red Bayesiana existente del repositorio.
  - ***Load from XBN file.*** Crea una Red Bayesiana que contiene los nodos especificados en un archivo XBN (formato XML para Redes Bayesianas).
    - ***Bound bayes net.*** Carga de una Red Bayesiana en la que, al agregar un nodo remoto, se añade únicamente éste y no los nodos que estén conectados a él en la red remota.
    - ***Complete bayes net.*** Carga de una Red Bayesiana en la que, al agregar un nodo remoto, se añaden también los nodos que estén conectados a él en la red remota.
  - ***Save as XBN.*** Guarda la red actual en formato XBN.
  - ***Close current net.*** Cierra la Red Bayesiana actual.
  - ***Exit.*** Cierra la herramienta.

- ***Node.*** Contiene los comandos utilizados para la manipulación de nodos.
  - ***Add node.*** Crea un nodo sencillo y lo añade a la red actual.
  - ***Include node.*** Añade a la red actual un nodo, de cualquier tipo, que se encuentre en una red remota.
  - ***Remove node.*** Elimina el nodo seleccionado.
  - ***Set parent.*** Agrega un nodo padre, local o remoto, al nodo seleccionado.
  - ***Set child.*** Agrega un nodo hijo, local o remoto, al nodo seleccionado.
  - ***Remove parent.*** Elimina el arco de un nodo padre del nodo seleccionado.
  - ***Remove child.*** Elimina el arco a un nodo hijo del nodo seleccionado.
  - ***Clear edges.*** Elimina todos los arcos del nodo o nodos seleccionados.
  - ***Move.*** Permite seleccionar un nodo para modificar su posición en la red.
  - ***Properties.*** Permite ver y modificar distintas propiedades del nodo seleccionado, sea éste de cualquier tipo.

- ***Distribution.*** Permite el manejo de las distribuciones de probabilidad dentro de los nodos.
  - ***View distribution.*** Muestra la distribución de probabilidad del nodo seleccionado.
  - ***Change distribution.*** Permite modificar la distribución de probabilidad del nodo seleccionado.
  - ***Validate distribution.*** Verifica que la suma de las probabilidades de los posibles valores del nodo actual sea igual a 1.0.
  - ***Adjust distribution.*** Ajusta los valores de probabilidad de la distribución del nodo actual, de modo que la suma sea 1.0, conservando las proporciones entre ellos.

- ***Compute.*** Realiza cálculos en la Red Bayesiana.

  - ***Compute probability.*** Calcula la densidad de probabilidad de un nodo, dado un escenario.
  - ***Predict node.*** Realiza la recuperación de una serie de tiempo a partir de la Red Bayesiana y algunos valores para otros nodos.

- ***Data.*** Contiene comandos para la utilización y manipulación de nodos de datos en la Red Bayesiana.
  - ***Add data node.*** Crea un nodo de datos cuya densidad de probabilidad es calculada a partir de una serie de tiempo.
  - ***Extract structure.*** Extrae la estructura de la Red Bayesiana a partir de los nodos de datos seleccionados.
    - ***Likehood method.*** Utiliza el método MLE para la extracción de la Red Bayesiana.
    - ***Information theory method.*** Utiliza el algoritmo de tres etapas para la extracción de la Red Bayesiana (ver sección 4.3.2).
  - ***Extract distribution.*** Extrae la distribución de probabilidad del nodo de datos seleccionado a partir de los datos del mismo.
  - ***Discretization.*** Comandos para la discretización de series de tiempo mediante el método basado en vectores (ver capítulo 5).
    - ***Rediscretize data.*** Discretiza de nuevo una serie de tiempo utilizando un valor diferente de σ.
    - ***Tabulate slope weights.*** Tabula la relación señal a ruido (SNR) de la discretización del nodo seleccionado variando el parámetro σ.
    - ***Search best slope weight.*** Realiza una búsqueda de aquel valor de σ que maximice la SNR para el nodo seleccionado.
    - ***Save discretized data.*** Guarda en un archivo de texto los valores discretos del nodo seleccionado.
    - ***Recover data from discretization.*** Recupera la serie de tiempo a partir de los valores discretos del nodo seleccionado.
    - ***Find slope weight and offset.*** Encuentra los valores de σ y los desplazamientos que producen el mejor acoplamiento entre las series de tiempo seleccionadas.

- ***Database.*** Comandos para la adición de nodos a partir de una base de datos.
  - ***Connect to database.*** Realiza una conexión a una base de datos para permitir la realización de consultas a la misma.
  - ***Add nodes from database.*** Permite realizar una consulta SQL y crear nodos a partir de los datos obtenidos.

Como ejemplo de uso considere la creación de una Red Bayesiana aislada en Windows. El siguiente comando se utilizaría para iniciar la red:

```
dbn -r
```

Suponga que se desea crear una red limitada (que no incluya a los nodos conectados a aquellos nodos remotos que se incluyan en la red). Para ello, en el menú se seleccionaría

System → *New* → *Bound bayes net*. Aparecerá una ventana en la que se pide el nombre de la red y otra en la que se pide la descripción. Supongamos que el nombre de la red es "Nueva red".

Si ahora se desea agregar un nodo de datos, cuyos datos provengan de una serie de tiempo *Seno*, se debe seleccionar *Data* → *Add data node*. Posteriormente se debe dar clic en el lugar en donde se desee agregar el nodo. Aparecerá un cuadro de dialogo como el que se muestra en la figura 9.14. Ahí se debe escribir el nombre del nodo, su descripción, la función de discretización a utilizar, el valor de σ (slope weight) y el número de símbolos (space length) que se utilizarán durante la discretización. También se deberá seleccionar el archivo en el que se encuentran los datos con formato de texto presionando el boton *Search*.

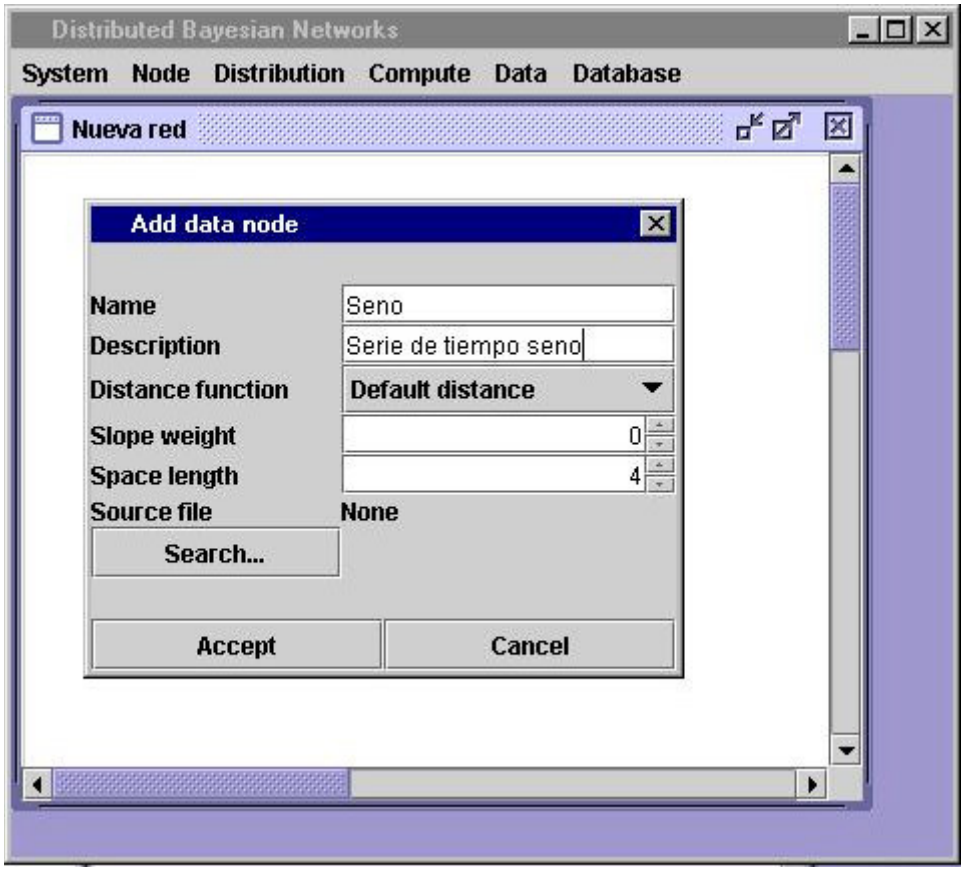


Figura 9.14. Cuadro de dialogo para agregar un nodo de datos a partir de una serie de tiempo

De la misma manera, es posible agregar un nodo *Coseno*. Si se desea extraer la estructura de una Red Bayesiana que contenga a estos dos nodos, será necesario seleccionarlos manteniendo presionada la tecla *Control* y dando clic sobre ambos nodos. Los nodos seleccionados cambiarán de color.

Posteriormente, se debe elegir del menú *Data* → *Extract structure* → *Likehood method*. Se mostrará un cuadro de diálogo en el que se permite elegir el máximo desplazamiento de las series, teniendo la opción de alinearlas o no, tal como se muestra en la figura 9.15. El máximo desplazamiento se puede entender como la mayor cantidad de mediciones que

pueden transcurrir antes de que la primera causa se refleje en la última consecuencia. En el caso del ejemplo que se presenta, se elige alinear las series con un máximo desplazamiento de 50.

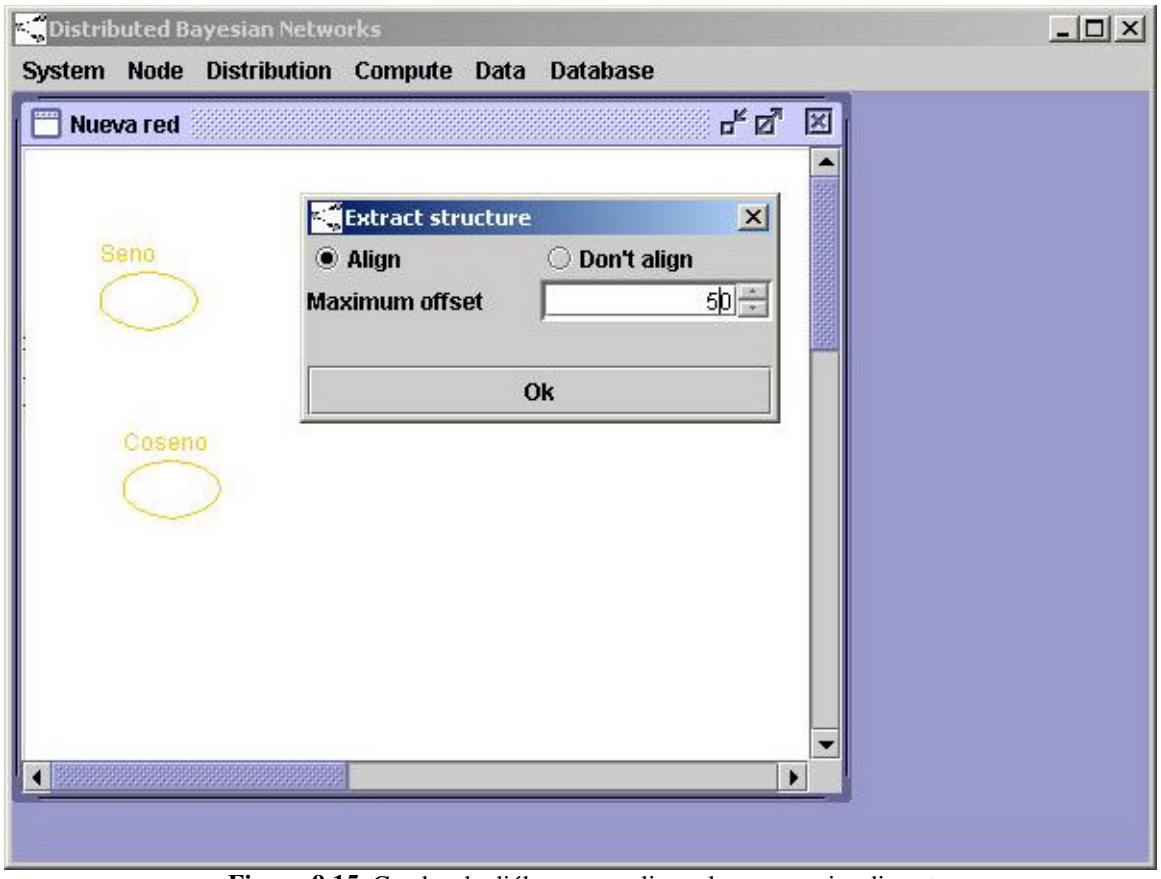


Figura 9.15. Cuadro de diálogo para alinear las secuencias discretas

El resultado de este proceso es simplemente un arco que va de coseno a seno, indicando una diferencia de tiempo de 26.

Note que en éste ejemplo se utilizó σ=0 para ambas series de tiempo. Si se desea encontrar un valor de σ más apropiado para minimizar el error por discretización, el procedimiento más recomendable es tabular la SNR sobre un amplio intervalo de σ, y posteriormente realizar una búsqueda local en aquellos valores que resulten interesantes.

Para encontrar el mejor valor de σ para el nodo *Seno* se comienza por seleccionarlo dando clic sobre él. Enseguida se debe elegir del menú *Data* → *Discretization* → *Tabulate slope weights*. Se mostrará un cuadro de diálogo en el que se solicita el valor inicial, el valor final y el tamaño del salto de σ en la tabulación. En este caso se utilizarán los valores 0, 1000 y 10 respectivamente.

El resultado de este proceso es una tabla con los valores de σ y sus respectivas SNR, como se muestra en la figura 9.16.

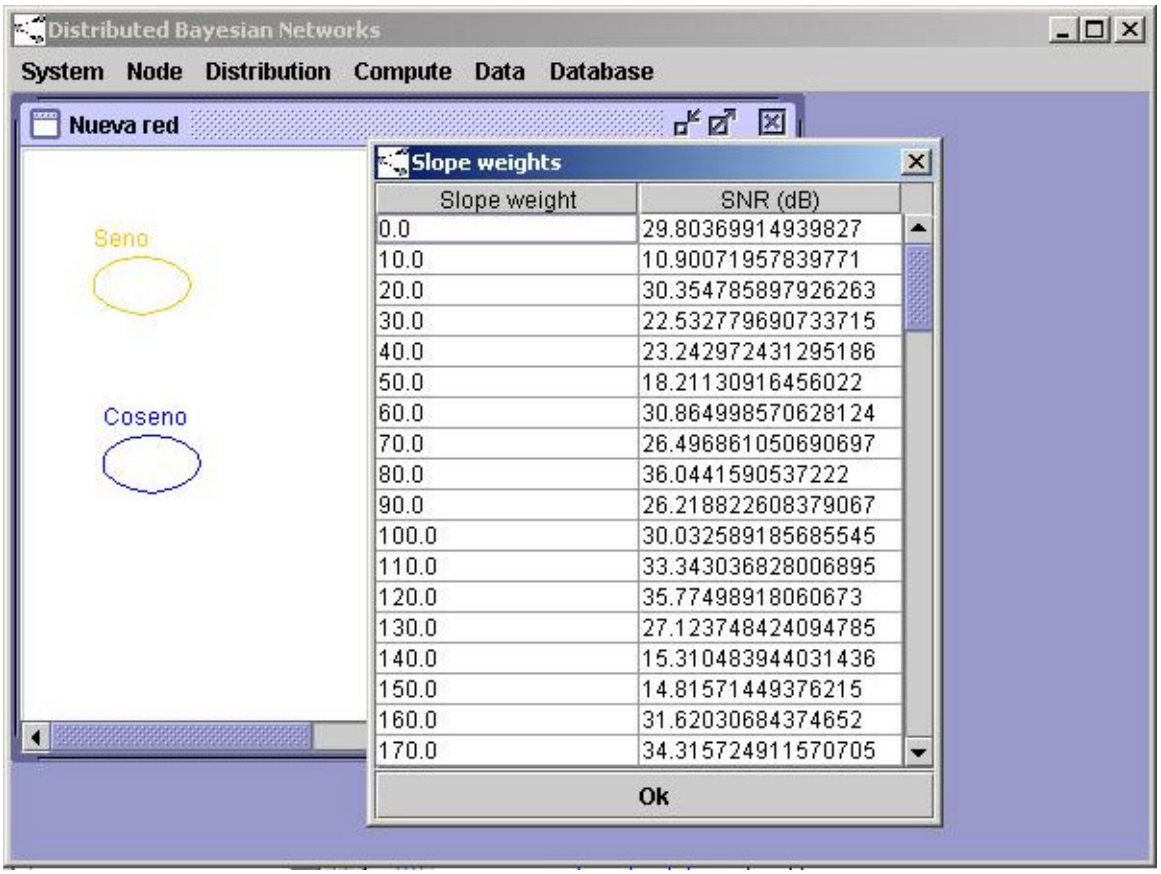


Figura 9.16. Tabulación del parámetro σ y la SNR

A simple vista se observa que el valor σ =80 presenta la mayor SNR, de modo que es conveniente realizar una búsqueda local alrededor de este valor. Para ello, teniendo seleccionado el nodo *Seno*, se debe elegir del menú *Data* → *Discretization* → *Search best slope weight*. Aparecerá el cuadro de diálogo que se muestra en la figura 9.17.

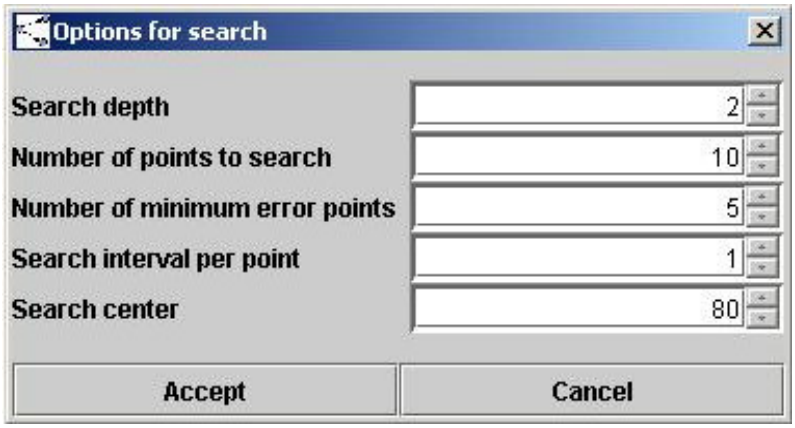


Figura 9.17. Cuadro de diálogo para búsqueda del mejor valor de σ

La búsqueda se realiza tomando como centro el valor indicado como *Search center*, y colocando la mitad de los puntos indicados en *Number of points to search* a la izquierda y

la mitad a la derecha. Cada punto tiene el tamaño indicado por *Search interval per point*, como se muestra en la figura 9.18.

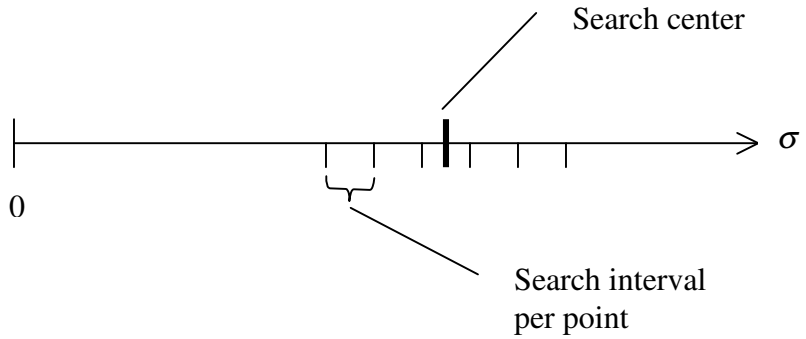


Figura 9.18. Parámetros en la búsqueda del mejor valor para σ

La serie de tiempo es discretizada y recuperada utilizando el valor de σ indicado en cada punto, y se toman los puntos cuyo error de discretización sea menor. El número de puntos tomados es indicado por *Number of minimum error points*. Hasta entonces se dice que se ha buscado en un nivel, de modo que si *Search depth* es mayor que 1, cada uno de los puntos tomados se utiliza como centro, y el intervalo es dividido entre el número de puntos a buscar (*Number of points to search*). La búsqueda se repite para cada uno de los subespacios indicados.

Una vez que se ha alcanzado el número de niveles indicado por *Search depth*, se realiza una búsqueda por descenso de gradiente para cada uno de los puntos resultantes y se elige aquel que presente el menor error de discretización.

En el caso de ejemplo, se encuentra que el mejor SNR se obtiene cuando $\sigma = 79.45$, obteniéndose un SNR prácticamente igual que con $\sigma = 80$, lo que indica que por coincidencia este valor parece ser el mejor.

Aunque la búsqueda de valores adecuados de σ para cada serie de tiempo es una opción útil, cuando se requiere extraer una red para varias series de tiempo es más conveniente utilizar el algoritmo de recocido simulado para encontrar, de forma simultánea, tanto el valor de σ como el desplazamiento de cada serie de tiempo. Para hacer esto, es necesario asignar un valor inicial de σ a cada serie de tiempo, seleccionándola y eligiendo la opción del menú *Node* → *Properties*, como se muestra en la figura 9.19. Comúnmente, un valor inicial de $\sigma = 1000$ para cada serie de tiempo produce un buen resultado.

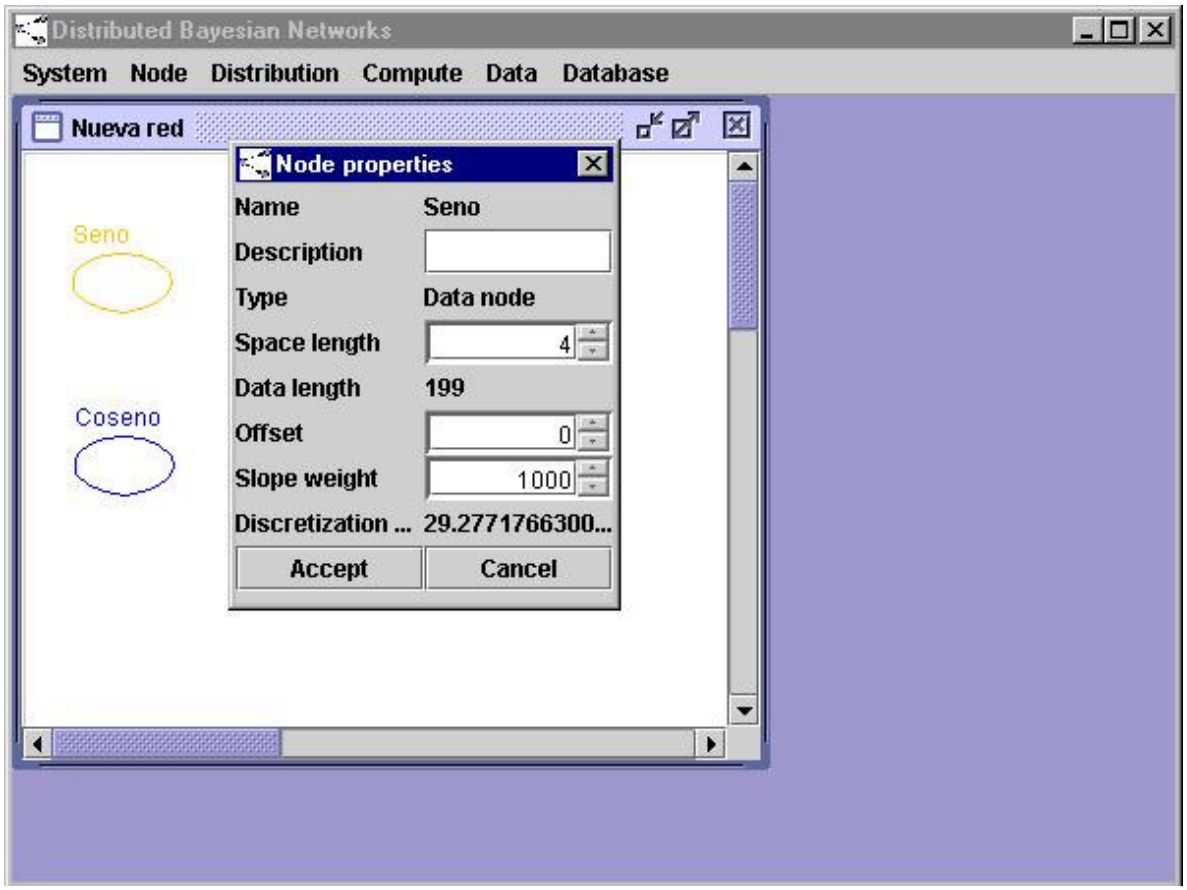


Figura 9.19. Cuadro de diálogo para modificar el valor de σ

Una vez establecido el valor de σ para todos los nodos, se deben seleccionar manteniendo presionada la tecla *Control* y dando clic sobre cada uno de ellos. A continuación se debe seleccionar la opción *Data* → *Discretization* → *Find slope weight and offset* del menú, con lo que aparecerá un cuadro de diálogo como el que se muestra en la figura 9.20.

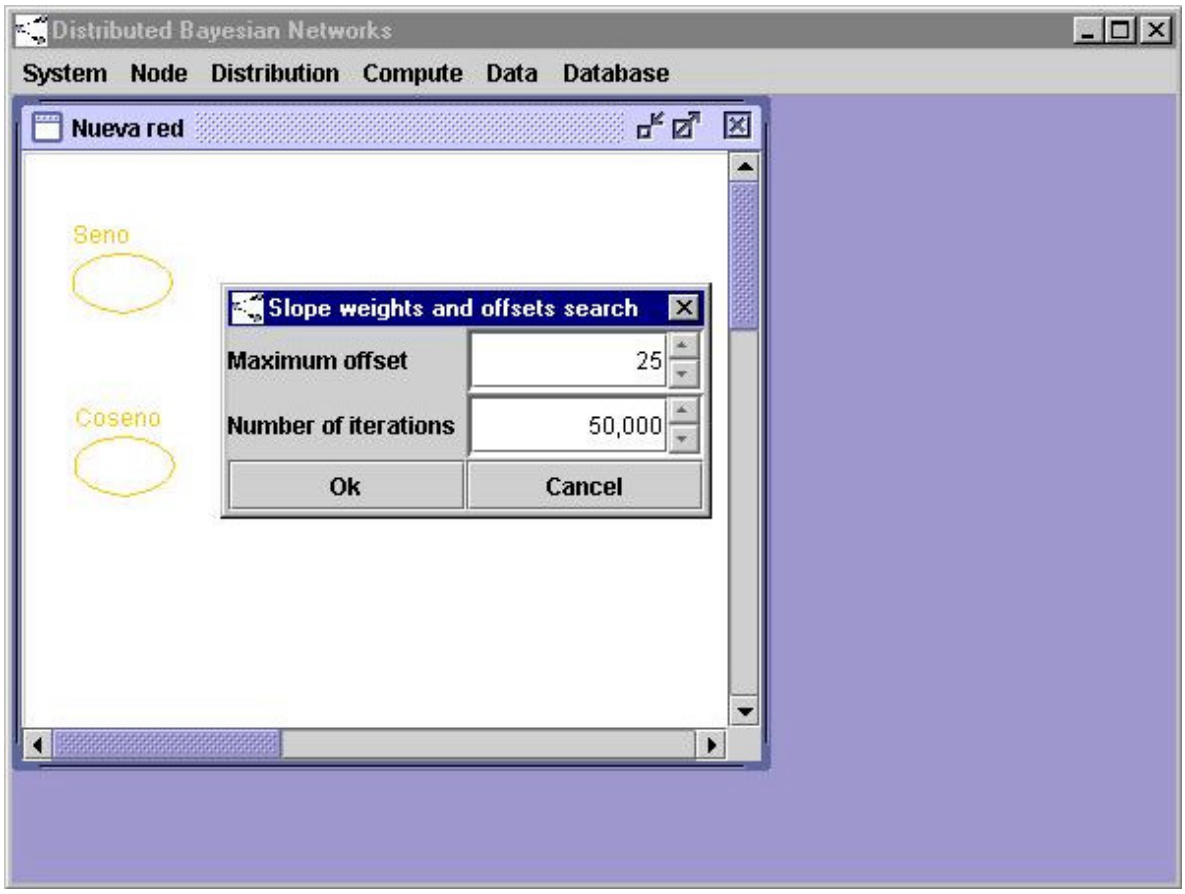


Figura 9.20. Cuadro de diálogo para buscar el valor de σ y el desplazamiento para varias series de tiempo

Seleccione el máximo desplazamiento entre las series de tiempo, así como el número de iteraciones para la búsqueda. Comúnmente se obtienen buenos resultados con un mínimo de 50,000 iteraciones. Al dar clic iniciará la búsqueda, y al terminar se tendrá el mejor acoplamiento encontrado para las series de tiempo.

10. CONCLUSIONES

El conocimiento de las relaciones entre las variables de un sistema permite su mejor entendimiento, a tal grado que en base al conocimiento que se tenga acerca de algunas de estas variables, es posible aproximar e incluso predecir aquellas que son desconocidas. En numerosas ocasiones el comportamiento de una variable se puede capturar en forma de una serie de tiempo. El modelado de relaciones entre series de tiempo ha sido estudiado en diversas disciplinas, tales como economía y biología. Es común que los modelos obtenidos se auxilien de grafos para su expresión.

Por otro lado, en el área de Machine Learning existen diversas técnicas, entre las que se encuentran las Redes Bayesianas, que permiten la expresión de relaciones entre variables. Estas redes se sirven de grafos dirigidos, manteniendo en cada nodo una función de densidad de probabilidad que expresa la relación de ese nodo con sus nodos padre, es decir, con aquellos nodos en los que se originan arcos incidentes al mismo.

Así, por un lado, los modelos de grafos para expresar relaciones entre series de tiempo se han limitado a mostrar su existencia, sin referirse a su naturaleza o comportamiento. Por otro lado, aunque las Redes Bayesianas mantienen información acerca de las relaciones por medio de una densidad de probabilidad, los algoritmos desarrollados para su extracción se han enfocado a la información contenida en bases de datos.

La propuesta central de este trabajo de tesis ha sido la utilización de Redes Bayesianas para expresar relaciones entre un conjunto de series de tiempo. Se ha observado que este conjunto, denominado base de datos temporal, contiene más información que una base de datos tradicional, haciendo posible el descubrimiento de desplazamientos temporales en las relaciones.

La mayoría de los algoritmos y técnicas empleadas soportan un conjunto de símbolos relativamente pequeño, por lo que se hace necesario discretizar las series de tiempo antes de poder aplicar sobre ellas algún algoritmo. La discretización es una etapa más importante de lo que se cree, debido a que la información que no sea capturada durante esta etapa no podrá, de ninguna manera, ser recuperada en una etapa posterior. Así, si se pierde la relación de un valor de la serie de tiempo respecto al anterior, o se pierde la posibilidad de expresar relaciones no lineales entre series de tiempo, las etapas posteriores heredarán estas limitaciones, haciendo inútil la utilización de los algoritmos más avanzados para conservar dicha información.

Por este motivo, se ha desarrollado una técnica especial de discretización capaz de conservar, de una manera definida, información tanto de la amplitud como de las variaciones de la serie de tiempo. Al estudiar esta nueva técnica, se observa que los métodos de discretización más populares son casos especiales que se presentan cuando el parámetro que define la relación entre amplitud y variaciones de la serie de tiempo (parámetroσ) es igual a cero o tiende a infinito. Los resultados obtenidos de diversas pruebas son consistentes con esta afirmación ya que, como era de esperarse, estos dos extremos no son siempre la mejor elección para el parámetro σ, por lo que una búsqueda

suficientemente exhaustiva del mismo permite generar una discretización tal que, con el mismo número de símbolos, produce una recuperación de mejor calidad que las técnicas tradicionales, incluso en las series de tiempo para las cuales éstas fueron desarrolladas (por ejemplo, señales de voz). Esto no significa que el método de discretización propuesto produzca una mejor calidad de recuperación que cualquier otro método existente, sino que, por sus características, puede representar a la serie de tiempo de una manera más conveniente para ciertos fines, sin ocasionar mayor pérdida de información que algunos de los métodos más utilizados.

Se ha observado que la función utilizada para medir la diferencia entre dos vectores produce resultados convenientes para los fines de esta tesis. Se ha establecido el conjunto de características que debe cumplir una función de este tipo, por lo cual es posible proponer alguna otra que pudiera producir mejores resultados para un contexto específico. Note que no se ha requerido que la función sea una métrica debido a que, si bien es una característica deseable, no se ha encontrado una razón de peso que impida la utilización de alguna función que no cumpla con este requisito.

Así como para la extracción de Redes Bayesianas a partir de bases de datos temporales fue necesario desarrollar la técnica de discretización basada en vectores, durante el desarrollo de este método de discretización surgió la necesidad de un método de agrupamiento (clustering) con características especiales. De esta manera, se ha presentado un nuevo método de agrupamiento que permite la adición de elementos de manera dinámica y proporciona un representante de cada agrupación formada. En este caso no sería acertado realizar una comparación con otros métodos, puesto que características que se presentaron como requisitos (por ejemplo, especificar el número de agrupamientos a formar) se han visto como obstáculos a superar en otros métodos. Del mismo modo, las agrupaciones obtenidas por este método serían vistas como incorrectas por un algoritmo que agrupe por densidad (por ejemplo, DBScan).

La nueva técnica de discretización también impone nuevos retos. Uno de ellos se presenta debido a que diferentes formas de discretización producen secuencias distintas, y dado que la extracción del modelo se basa en estas secuencias, variaciones en el parámetro de discretización producen variaciones en el modelo obtenido. Esto hace necesario contar con una forma de validar los modelos a fin de hacerlos más confiables.

La manera más directa de validar un modelo propuesto es reproducir datos conocidos, valiéndose de parámetros también conocidos. Tomando en cuenta que una Red Bayesiana es un modelo cuyos parámetros de entrada son valores en algunas variables, una manera de validarla es reproducir una serie de tiempo conocida a partir de otras series también conocidas. Si existen diferencias de tiempo positivas entre la variable que se desea recuperar y aquellas variables a las que se les asignan valores, es posible predecir algunos valores posteriores a aquellos incluidos en el conjunto de entrenamiento. Tomando en cuenta la definición de *Causalidad de Granger*, la predicción de una variable a partir de otra permite confirmar o refutar la relación de causalidad establecida entre éstas.

Por otro lado, el hecho de que sea posible validar un modelo no hace menos importante la búsqueda del mejor modelo posible. Se sabe que el modelo depende del valor de σ

asignado a cada serie para su discretización, y es claro que este parámetro debe permitir que la serie de tiempo sea recuperada con buena calidad. Sin embargo, cuando se cuenta con un conjunto de series de tiempo, el parámetro σ asignado a cada una de ellas podría revelar o esconder relaciones existentes, es decir, es posible que variaciones en una serie de tiempo afecten la amplitud de otra, y esta relación solo sería visible al discretizar cada serie de tiempo con un valor de σ específico. Asimismo, la relación entre las series de tiempo podría presentarse con algún desplazamiento en el tiempo.

De este modo, es necesario realizar la búsqueda del valor de σ y del desplazamiento adecuado para cada serie de tiempo, intentando hallar la mejor relación entre las secuencias discretas y la mejor calidad de recuperación para cada serie de tiempo. En términos prácticos, esto implica maximizar la información mutua entre las secuencias discretas y afectar el resultado con la calidad de recuperación de cada serie de tiempo. Así, ha sido utilizado un algoritmo basado en recocido simulado, tomando como medida de fortaleza de una configuración la suma de la información mutua entre pares de variables y la calidad de discretización de cada una de éstas.

En la siguiente etapa, el trabajo aborda el área de extracción de Redes Bayesianas a partir de conjuntos de casos, punto de estudio en el que diversos investigadores alrededor del mundo han contribuido con cuantiosas síntesis. Realizar una aportación más en esta área queda fuera del alcance de la presente tesis, por lo que únicamente se han utilizado algoritmos publicados.

La decisión entre utilizar el algoritmo MLE o el algoritmo de tres etapas depende de la cantidad de nodos contenidos en la red y del tipo de resultados deseado. En las pruebas se observa que el algoritmo MLE produce recuperaciones cuyo nivel de detalle es mayor que aquellas producidas con el algoritmo de tres etapas, pero este nivel de detalle también puede introducir errores considerables, como se comprueba al mirar el error MSER en la tabla 7.2. Debe tenerse en cuenta que al utilizar el algoritmo MLE se han generado todas las posibles estructuras, lo que necesariamente produce mejores resultados que si se utilizara algún algoritmo de búsqueda, pero lo hace inútil cuando el número de nodos crece (note, por ejemplo, que no ha sido posible generar una estructura para la prueba 3 con este algoritmo).

De este modo, temas tan diversos como series de tiempo, discretización, agrupamiento, maximización de funciones, Machine Learning y predicción, han quedado engarzados en una herramienta que propone una nueva forma de expresar y utilizar las relaciones existentes entre variables.

Los resultados obtenidos son aún imperfectos. El trabajo a realizar sobre el proceso de extracción de Redes Bayesianas a partir de Bases de Datos temporales es abundante. Dentro de las tareas más importantes que aún quedan por desarrollar se encuentran:

- Incorporación de mejores técnicas para extracción de Redes Bayesianas a partir de las secuencias discretas alineadas.

- Mejora en la búsqueda de los parámetros correctos de alineación y discretización de las series de tiempo.
- Formalización del método de discretización basada en vectores.

Aunque las Redes Bayesianas han mostrado ser una forma de representación sencilla y poderosa, sus limitaciones aparecen rápidamente cuando se aplican a fenómenos cuyo comportamiento no cumple plenamente con las características establecidas para los conjuntos de datos con los cuales se pensó podrían trabajar. Por lo tanto, es viable plantearse la necesidad de realizar modificaciones a su definición, o incluso la conveniencia de la creación de una nueva forma de representación, la cual debería, entre otras características, permitir reflejar de manera más clara las variaciones en los retardos entre la ocurrencia de una causa y su efecto. Por ejemplo, debería permitir reflejar casos inusuales como los ocasionados por la crisis provocada por el “Error de Diciembre”, que afectó de manera casi simultánea a un considerable número de variables que comúnmente no están sincronizadas.

APÉNDICE A. RESULTADOS OBTENIDOS CON OTROS MÉTODOS

En este apéndice se muestra tanto la estructura generada por Correlation Metric Construction [Arkin et. al., 1997] como el Grafo de Causalidad de Granger y el Grafo de Correlación Parcial [Dahlhaus & Eichler, 2000] obtenidos con las series de tiempo utilizadas para cada una de las pruebas mostradas en las secciones 6.3 y 7.3, a fin de permitir una comparación entre distintos modelos.

La figura A.1 muestra la estructura generada por Correlation Metric Construction para el conjunto de datos utilizado en la prueba 1 (secciones 6.3.1 y 7.3.1).

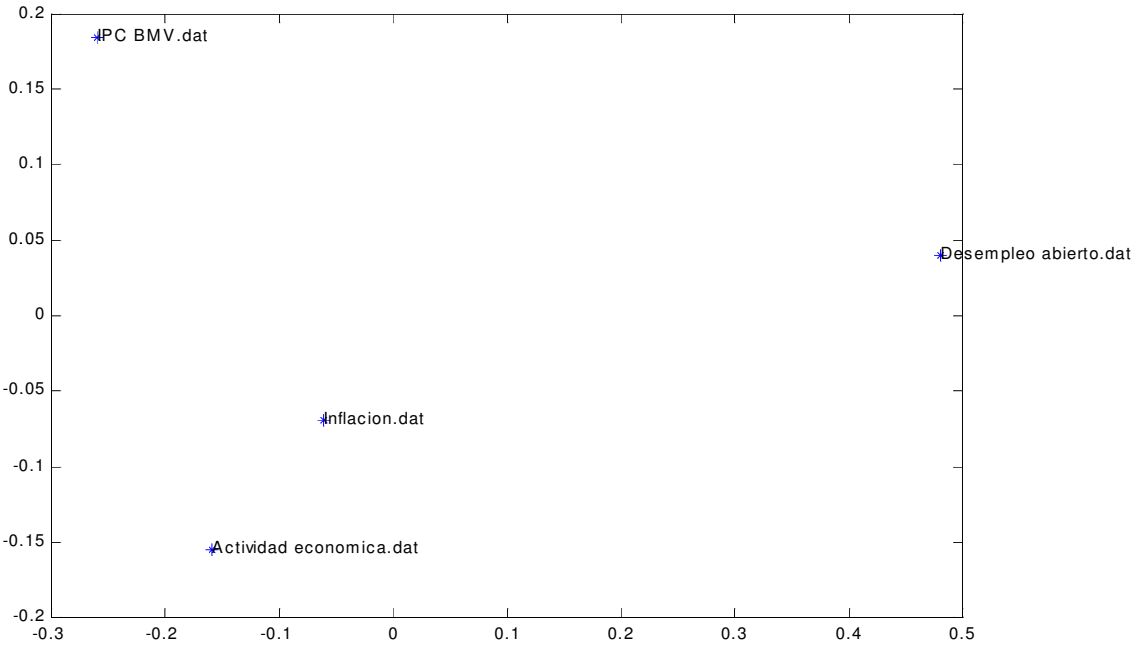


Figura A.1. Estructura obtenida con el método Correlation Metric Construction

La figura A.2 muestra el Grafo de Causalidad de Granger para las mismas series de tiempo.

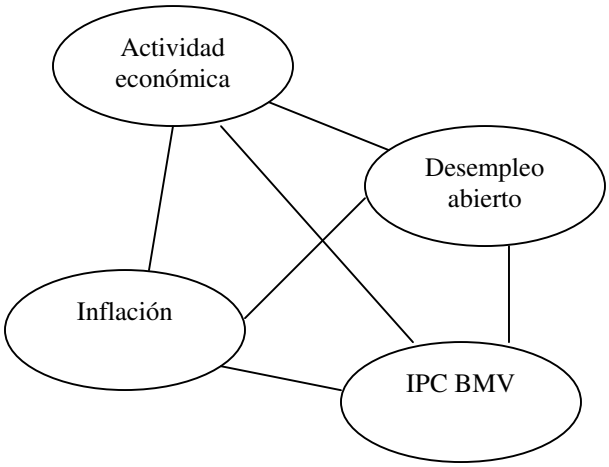


Figura A.2. Grafo de Causalidad de Granger para las series de tiempo de la prueba 1

La figura A.3 muestra el Grafo de Correlación Parcial obtenido para las series de tiempo de la prueba 1. Como se puede observar, el grafo aparece sin arcos entre sus nodos, lo que indica que para este modelo no se obtuvo relación alguna entre las series de tiempo.

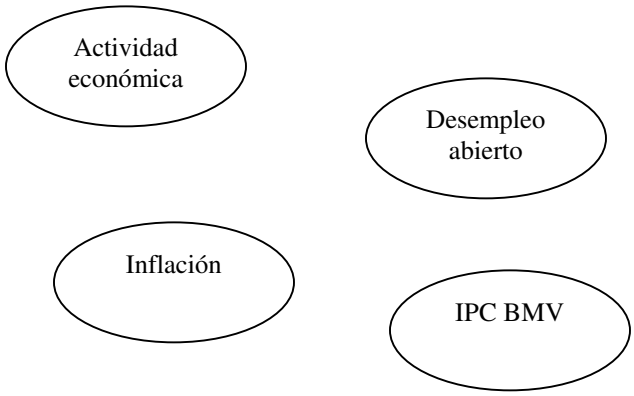


Figura A.3. Grafo de Correlación Parcial para las series de tiempo de la prueba 1

La figura A.4 muestra la estructura obtenida para las series de tiempo de la prueba 2 (secciones 6.3.2 y 7.3.2) al utilizar el método Correlation Metric Construction.

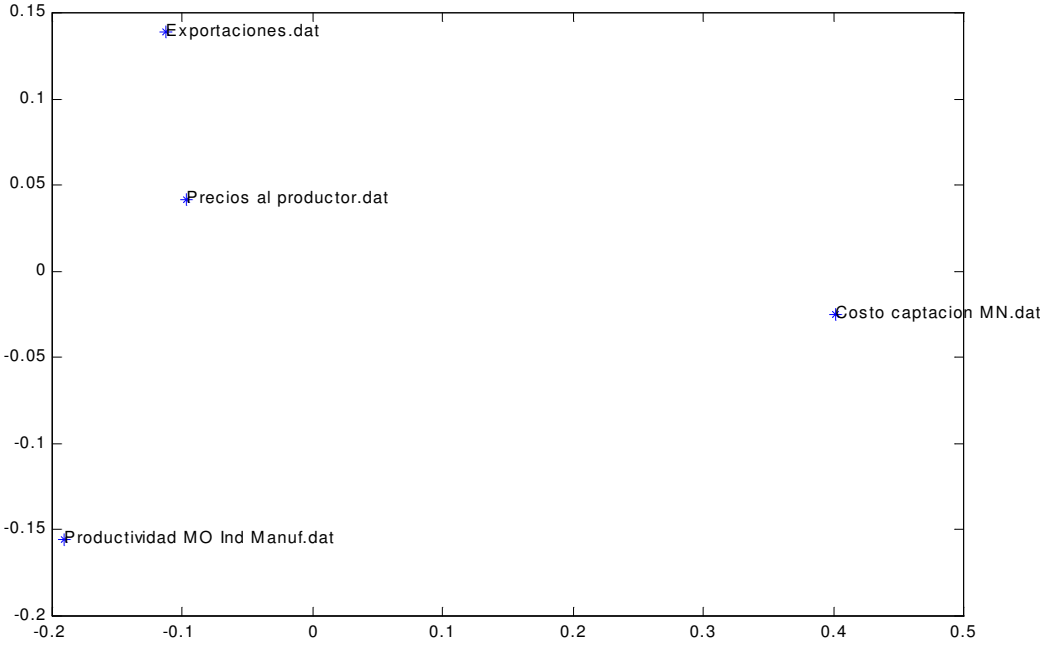


Figura A.4. Estructura obtenida utilizando Correlation Metric Construction

La figura A.5 muestra el Grafo de Causalidad de Granger obtenido para este conjunto de series de tiempo.

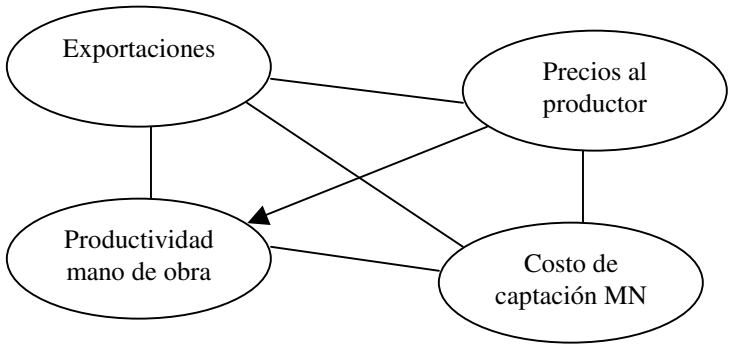


Figura A.5. Grafo de Causalidad de Granger para las series de tiempo de la prueba 2

La figura A.6 muestra el Grafo de Correlación Parcial obtenido para las series de tiempo de la prueba 2.

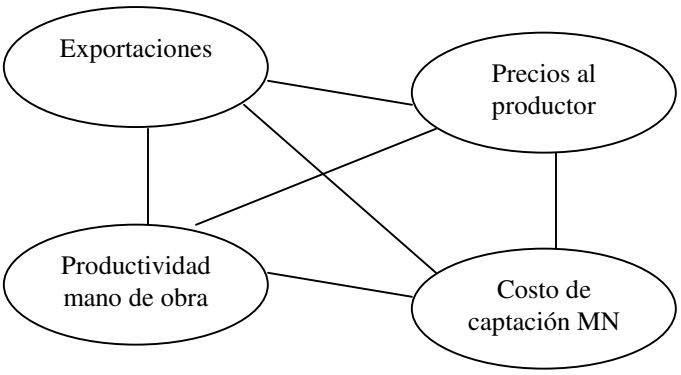


Figura A.6. Grafo de Correlación Parcial obtenido para la prueba 2

La figura A.7 muestra el resultado obtenido al aplicar Correlation Metric Construction a las series de tiempo de la prueba 3 (secciones 6.3.3 y 7.3.3).

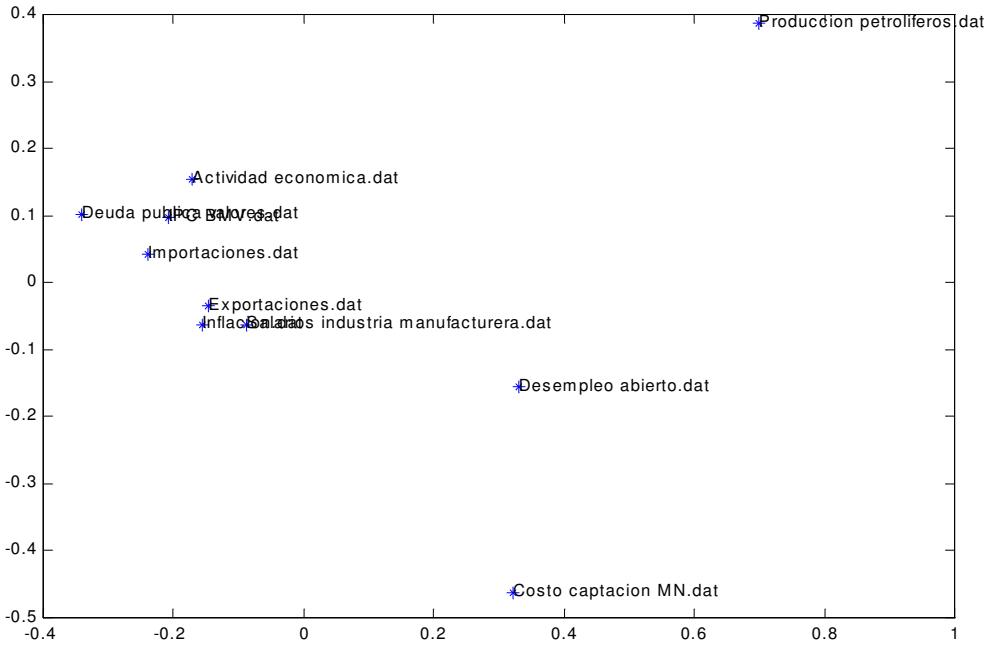


Figura A.7. Estructura obtenida utilizando Correlation Metric Construction

La figura A.8 muestra el Grafo de Causalidad de Granger obtenido para las series de tiempo de la prueba 3.

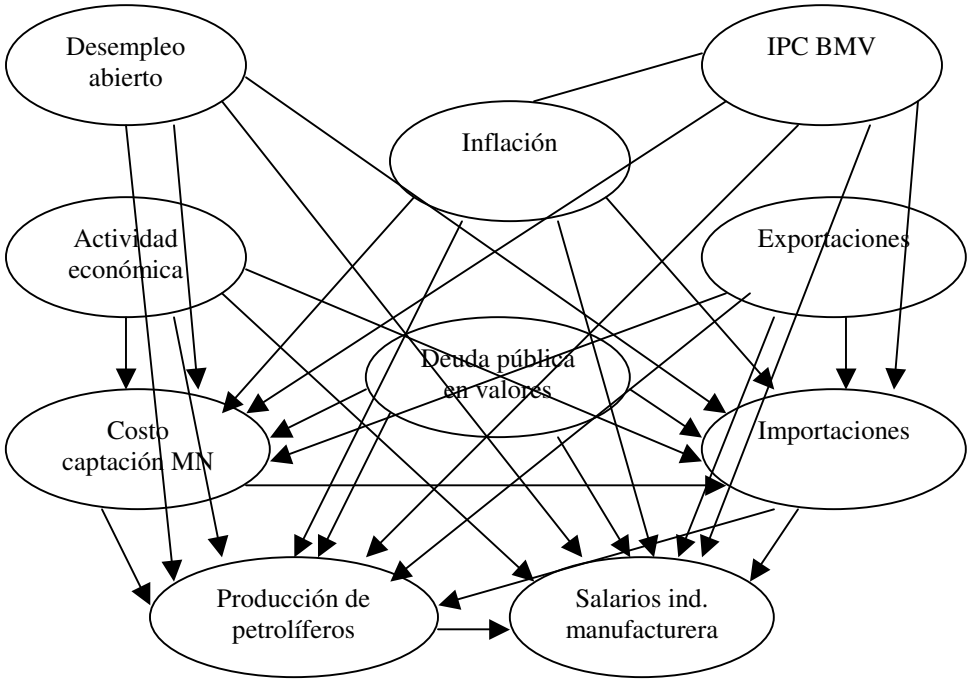


Figura A.8. Grafo de Causalidad de Granger para las series de tiempo de la prueba 3

La figura A.9 muestra el Grafo de Correlación Parcial obtenido para el conjunto de series de tiempo de la prueba 3. El grafo resultante es completo, al igual que en la prueba 2.

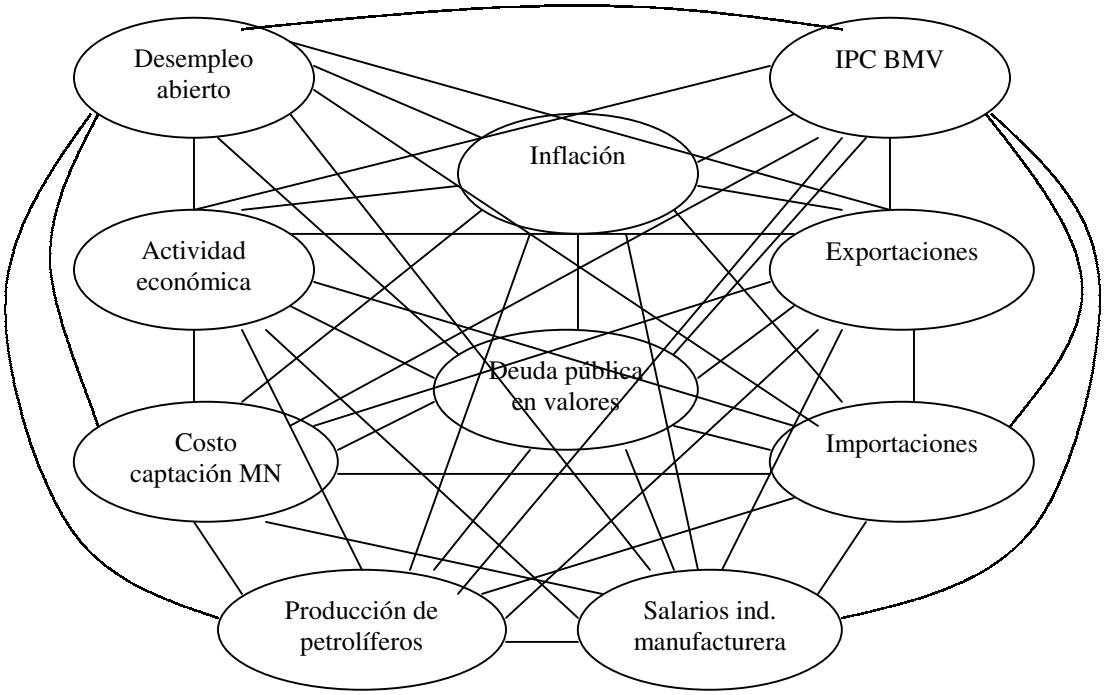


Figura A.9. Grafo de Correlación Parcial para las series de tiempo de la prueba 3

La figura A.10 muestra la estructura obtenida al aplicar Correlation Metric Construction al conjunto de series de tiempo de la prueba 4 (secciones 6.3.4 y 7.3.4).

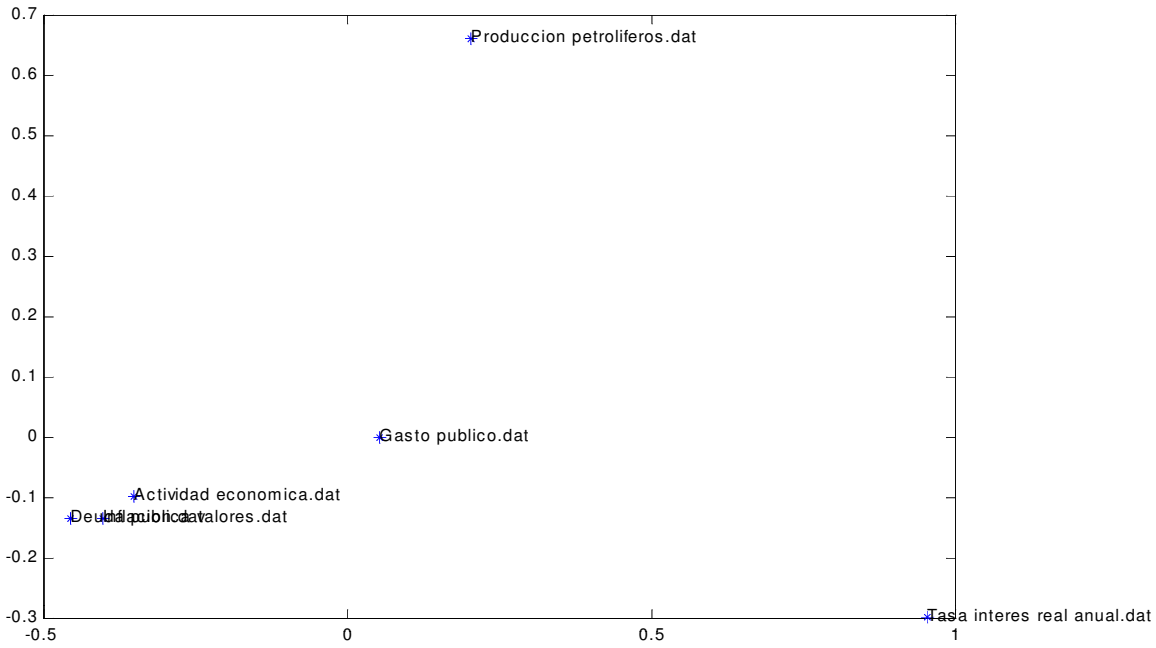


Figura A.10. Estructura obtenida al aplicar Correlation Metric Construction al conjunto de series de tiempo de la prueba 4

La figura A.11 muestra el Grafo de Causalidad de Granger obtenido para estas series de tiempo.

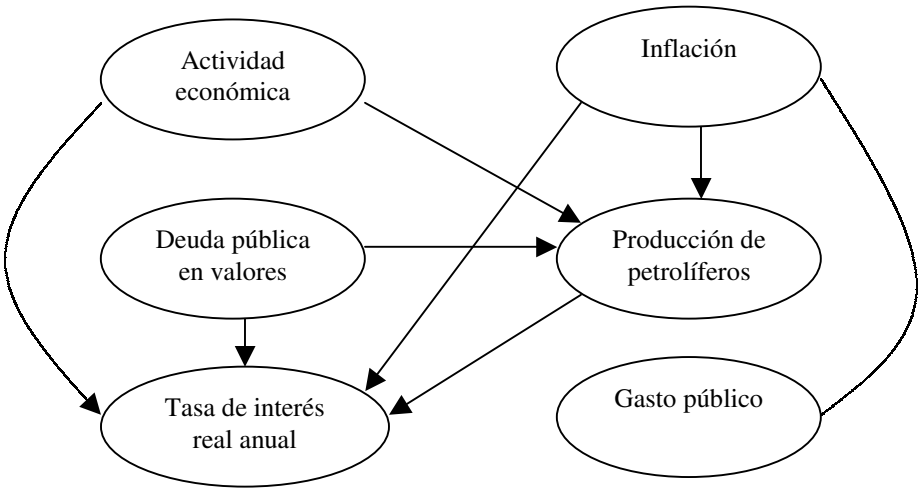


Figura A.11. Grafo de Causalidad de Granger obtenido para las series de tiempo de la prueba 4

La figura A.12 muestra el Grafo de Correlación Parcial obtenido para el mismo conjunto de series de tiempo. Nuevamente este grafo resulta ser completo.

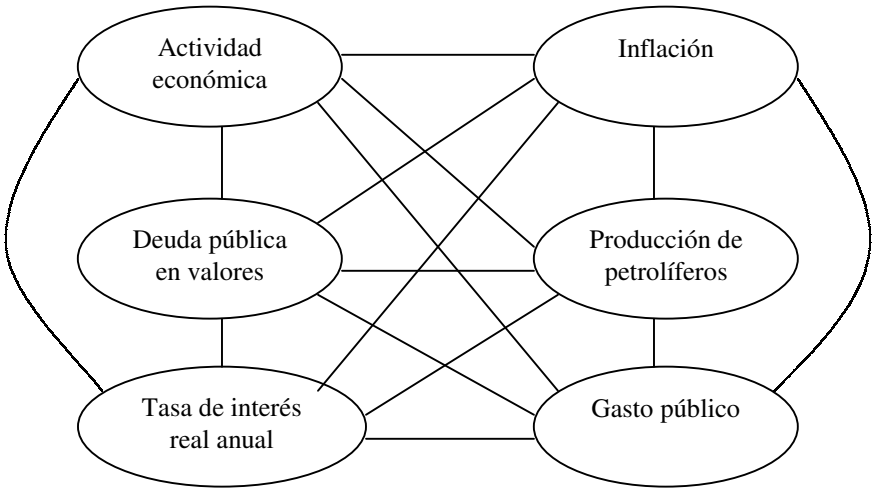


Figura A.12. Grafo de Correlación Parcial obtenido para las series de tiempo de la prueba 4

REFERENCIAS

[Arkin et. al., 1997] Adam Arkin, Peidong Shen, John Ross. A test case of correlation metric construction of a reaction pathway from measurements. Science 277. pp 1275-1279

[Avila & Figueroa, 2002] C. Avila-Sánchez, J. Figueroa-Nazuno. Dinámica discreta de N-cuerpos en interacción. XLV Congreso Nacional de Física.

[Battaglia, 1996] Glenn J. Battaglia. Mean Square Error. AMP Journal of Technology. Vol. 5. pp 31-36.

[Bautista and Figueroa, 2002] Bautista-Thompson and Figueroa-Nazuno. (2002). Matriz de Conocimiento sobre la Complejidad de Predicción en Series de Tiempo. VII Congreso Iberoamericano de Reconocimiento de Patrones (CIARP).

[Berzuini, 1990] C. Berzuini. Representing time in causal probabilistic networks. Uncertainty in Artificial Intelligence 5. pp 15-28.

[Chaitin, 1977] G. J. Chaitin. Algorithmic Information Theory. IBM Journal of Research and Development 21. pp. 350-359, 496.

[Chan, 2002] Ngai Hang Chan. Time Series Applications to Finance. Wiley-Interscience.

[Chan et. al., 1991] C. C. Chan, C. Batur, A. Srinivasan. Determination of quantization intervals in rule based model for dynamic systems. Proceedings of the IEEE Conference on Systems, Man, and Cybernetics. pp. 1719-1723.

[Cheng et. al., 1997] Jie Cheng, David A. Bell and Weiru Liu. Learning Belief Networks from Data: An Information Theory Based Approach. Proceedings of ACM CIKM'97. pp 325-331.

[Chiu et. al., 1990] D. K. Y. Chiu, B. Cheung, A. K. C. Wong. Information synthesis based on hierarchical entropy discretization. Journal of Experimental and Theoretical Artificial Intelligence. Vol. 2. pp. 117-129.

[Chmielewski & Grzymala-Busse, 1994] M. R. Chmielewski, J. W. Grzymala-Busse. Global discretization of continuous attributes as preprocessing for machine learning. Third International Workshop on Rough Sets and Soft Computing. pp. 294-301.

[Chow & Liu, 1968] C. K. Chow, C. N. Liu. Approximating discrete probability distributions with dependence trees. IEEE Transactions on Information Theory, IT-14(3). pp 462-467.

[Chu et. al, 2002] Shu-Chuan Chu, John F. Roddick, Jeng-Shyang Pan. An Incremental Multi-Centroid, Multi-Run Sampling Scheme for k-medoids-based Algorithms – Extended Report. Knowledge Discovery and Managemente Laboratory

[Cooper & Herskovits, 1992] G. F. Cooper, E. Herskovits. A Bayesian Method for the induction of probabilistic networks from data. Machine Learning, 9. pp 309-347.

[Craven & Shavlik, 1997] Mark W. Craven, Jude W. Shavlik. Understanding Time-Series Networks: A case study in rule extraction. International Journal of Neural Systems, Vol. 8, Nos. 4. pp 373-384.

[Dahlhaus et. al., 1997] R. Dahlhaus, M. Eichler, and J. Sandkühler. Identification of synaptic connections in neural ensembles by graphical models. Journal of Neuroscience Methods, 77. pp 93-107.

[Dahlhaus & Eichler, 2000] Rainer Dahlhaus and Michael Eichler. Causality and graphical models in time series analysis. P. Green, N. Hjort, and S. Richardson (eds.): Highly structured stochastic systems. University Press, Oxford.

[Dougherty et. al., 1995] James Dougherty, Ron Kohavi, Mehran Sahami. Supervised and Unsupervised Discretization of Continuous Features. Proceedings of the Twelfth International Conference.

[Eichler, 2001] M. Eichler. Markov properties for graphical time series models, Preprint, University of Heidelberg.

[Ester et. al., 1996] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise. Second International Conference on Knowledge Discovery and Data Mining pp. 226–231.

[Etxeberria et. al., 1997] R. Etxeberria, P. Larrañaga and J. M. Picaza. Analysis of the behaviour of genetic algorithms when learning Bayesian netowrk structure from data. Pattern Recognition Letters 18 (11-13) 1269-1273.

[Fayyad & Irani, 1993] U. M. Fayyad, K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. Proceedings of the 13th International Joint Conference on Artificial Intelligence. pp. 1022-1027.

[Flückiger, 1995]. Daniel Federico Flückiger. Beiträge zur Entwicklung eines vereinheitlichten Informations-Begriffs. Inauguraldissertation der Philosophisch-naturwissenschaftlichen Fakultät der Universität Bern

[Gamma et. al., 1995] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Design Patterns. Elements of Reusable Object Oriented Software. Addison Wesley.

[Gaweda et. al., 2000] E. Gaweda, R. Setiono and J. M. Zurada, Rule Extraction from Feedforward Neural Network for Function Approximation, Proceedings of the Fifth Conference Neural Networks And Soft Computing, pp. 311-316

[Gil & Badía, 2002] Reynaldo Gil-García, José Manuel Badía-Contelles. Algoritmo de agrupamiento GLC paralelo. VII Congreso Iberoamericano de Reconocimiento de Patrones (CIARP). pp 383-394.

[Granger, 1969] C. W. J. Granger. Investigating causal relations by econometric models and cross-spectral methods. Econometrica, 37. pp 424-438.

[Hassoun, 1995] Mohamad H. Hassoun. Fundamentals of Artificial Neural Networks. MIT Press.

[Heckerman, 1996] D. Heckerman. A Tutorial on Learning With Bayesian Networks. Technical report MSR-TR-95-06, Microsoft Research.

[Holte, 1993] R. C. Holte. Very simple classification rules perform well on most commonly used datasets. Machine Learning. Vol. 11. pp. 63-90.

[Hyvärinen et. al., 2001] Aapo Hyvärinen, Juha Karhunen, Erkki Oja. Independent Component Analysis. Wirley-Interscience. pp 15-52.

[Karypis et. al, 1999] George Karypis, Eui-Hong Han, Vipin Kumar. Chameleon: Hierarchical Clustering Using Dynamic Modeling. Computer. pp 68-75.

[Kerber, 1992] R. Kerber. Chimerge: Discretization of numeric attributes. Proceedings of the Tenth National Conference on Artificial Intelligence. MIT Press. pp. 123-128.

[Kirkpatrick et. al, 1983] S. Kirkpatrick, C. D. Gellat, M. P. Vecchi. Optimization by simulated annealing. Science. Vol. 220. pp 671-680.

[Kohonen, 1989] T. Kohonen. Self-Organization and Associative Memory. Springer-Verlag.

[Komorowski & Zytkow, 1997] Jan Komorowski, Jan Zytkow. Principles of Data Mining and Knowledge Discovery. Springer.

[McQueen, 1967] J. McQueen. Some methods for classification and analysis of multivariate observations. Proc. of the Fifth Berkley Symposium on Math, Stat. and Prob. Vol. 1. pp 281-296.

[Medina & Figueroa, 2002] Medina-Apodaca, Figueroa-Nazuno. Distributed Bayesian Networks for Patterns Representation. VII Congreso Iberoamericano de Reconocimiento de Patrones (CIARP). pp 371-382.

[Medina & Figueroa, 2003] Medina-Apodaca, Figueroa-Nazuno. Bayesian Networks extraction from a set of Time Series. 4to. Simposium Intertecnológico de Computación e Informática (SICI). pp 421-428.

[Mirer, 1983] Thad W. Mirer. Economic Statics and Econometrics. Macmillan Publishing. pp 113, 114.

[Mitchell, 1997] Tom M. Mitchell. Machine Learning. Mc. Graw Hill.

[Morrison, 1991] Foster Morrison. The art of Modeling Dynamic Systems. Wiley-Interscience. pp 38-41.

[Mrowczynski, 2000] Stanislaw Mrowczynski. Phi measure of azimuthal fluctuations. Acta Phys.Polon B31. pp 2065-2073.

[Ng & Han, 2002] Raymond T. Ng, Jiawey Han. CLARANS: A method for clustering objects for spatial data mining. IEEE Transactions on Knowledge and Data Engineering. Vol 14. No 5. pp. 1003-1016.

[Oates et. al., 1996] T. Oates, M. D. Schmill, P. R. Cohen. Parallel and Distributed Search for Structure in Multivariate Time Series. Technical Report 96-23, University of Massachusetts at Amherst, Computer Science Department.

[Pearl, 1988] Judea Pearl. Probabilistic Inference in Intelligent Systems. Morgan Kaufmann, San Mateo, California.

[Pearl and Russell, 2000] Judea Pearl and Stuart Russell. Bayesian Networks. M. Arbib (Ed.), Handbook of Brain Theory and Neural Networks, MIT Press.

[Pfahringer, 1995] B. Pfahringer. Compression-based discretization of continuous attributes. Proceedings of the Twelfth International Conference on Machine Learning.

[Pole et. al., 1994] Andy Pole, Mike West, Jeff Harrison. Applied Bayesian Forecasting and Time Series Analysis. Chapman and Hall.

[Ramoni & Sebastiani, 1997] M. Ramoni, P. Sebastiani. Discovering Bayesian networks in incomplete databases. Technical report KMI-TR-46, Knowledge Media Institute, The Open University.

[Rebane & Pearl, 1987] George Rebane, Judea Pearl. The recovery of causal poly-trees from statistical data. Conference on Uncertainty in Artificial Intelligence. pp 222-228.

[Richeldi & Rossotto, 1995] M. Richeldi, M. Rossotto. Class-driven statistical discretization of continuous attributes. European Conference on Machine Learning.

[Rouchka, 1997] E. Rouchka. A Brief Overview of Gibbs Sampling. IBC Statistics Study Group.

[Russell and Norvig, 1995] S. Russell, P. Norvig. Artificial Intelligence: A Modern Approach. Prentice-Hall.

[Sánchez & Figueroa, 2001] C. Sánchez-Rodríguez, J. Figueroa-Nazuno. Predicción de datos geológicos utilizando Redes Neuronales. XLIV Congreso Nacional de Física.

[Scargle, 2001] Bayesian Estimation of Time Series Lags and Structure. MAXENT2001: Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering.

[Shannon, 1948] C. E. Shannon. A Mathematical Theory of Communication. The Bell System Technical Journal. pp 379-423.

[Sprites et. al, 2000] Peter Sprites, Clark Glymour and Richard Scheines. Causation, Prediction and Search. Second edition. MIT Press. pp 5-155

[Stremler, 1993] Ferrel G. Stremler. Introducción a los sistemas de comunicación. Tercera edición. Addison-Wesley.

[Takens, 1981] F. Takens. Detecting strange attractors in turbulence. Lecture notes in mathematics. Vol 898. Dynamical Systems and Turbulence. pag. 366. Springer, Berlin.

[Tawfik & Neufeld, 1994] Ahmed Y. Tawfik, Eric Neufeld. Temporal Bayesian Networks. Proceedings of First International Workshop on Temporal Representation and Reasoning (TIME)

[Van de Merckt, 1993] T. Van de Merckt. Decision trees in numerical attribute spaces. Proceedings of the 13th International Joint Conference on Artificial Intelligence. pp. 1016-1021.

[Weiss et. al., 1990] S. M. Weiss, R. S. Galen, P. V. Tadepalli. Maximizing the predicative value of production rules. Artificial Intelligence. No. 45. pp. 47-71.