

ÍNDICE GENERAL

1	INTRODUCCIÓN	1
2	HIDDEN MARKOV MODEL	2
2.1	Modelo de Markov	3
2.2	Modelo oculto de Márkov	5
3	HMM USANDO EM	10
3.1	Estimación de parámetros del HMM	10
3.2	Algoritmo <i>backward-forward</i>	13
3.2.1	Factor de escala	15
3.3	Selección de número de interlocutores usando boots- trap	17
4	HMM USANDO MCMC	18
5	SPEAKER DIARISATION	19
5.1	Aplicaciones de <i>speaker diarisation</i>	19
5.2	Procesamiento de señal	20
5.2.1	Esquema general del sistema	21
5.2.2	Pre-procesamiento de señal de audio	21
5.2.3	Obtención de características acústicas	22
5.3	Resolver usando modelo HMM usando EM	25

INTRODUCCIÓN

HIDDEN MARKOV MODEL

Cuando se tienen datos secuenciales, se pueden agrupar en dos tipos de acuerdo a las características de la distribución que los generó: distribuciones estacionarias y distribuciones no estacionarias.

En el caso de secuencias de datos estacionarias, se considera que los datos *evolucionan* a través del tiempo, pero la distribución a partir de la cuál fueron generados permanece igual; mientras que para el caso de datos no estacionarios, su distribución generativa varía también según pasa el tiempo.

Para el problema de *speaker diarisation*, consideraremos que los datos son estacionarios; es decir, supondremos que su distribución generativa no cambia. Para resolver nuestro problema entonces trataremos a partir de los datos ajustar una distribución inicial e inferir sus parámetros correspondientes.

Este tipo de abstracción, permiten modelar una gran cantidad de situaciones, en las que se tienen observaciones de acuerdo al tiempo y se quiere predecir el siguiente valor en la serie, dadas la observaciones que se tienen hasta el momento.

Comúnmente se suelen considerar únicamente las observaciones más recientes, pues son las que pudiéramos pensar son más informativas para la predicción; además de que al asumir que las predicciones de nuevos datos sólo dependen de las últimas observaciones, nos permite simplificar mucho el modelo de la distribución generativa.

Para trabajar con este tipo de datos, se puede usar entonces un modelo de Márkov que es una red bayesiana en la que se asume la independencia de las variables aleatorias que corresponden a las predicciones futuras con todas las observaciones excepto las últimas.



Figura 2.1: Observaciones independientes e idénticamente distribuidas.

2.1 MODELO DE MARKOV

El modelo generativo más sencillo que se podría pensar, es considerar que todas las observaciones son independientes, es decir, que son variables aleatorias i.i.d, cuyo modelo gráfico se muestra en la [Figura 2.1](#).

Al asumir que los datos no están relacionados, sin embargo, se puede perder gran cantidad de información relativa al orden en que se fueron dando estas observaciones.

Es decir, si se tiene un conjunto N de observaciones $\mathbf{X} = \{x_1, \dots, x_N\}$, y se asume que son i.i.d, la distribución conjunta de la secuencia de datos se puede escribir como sigue:

$$p(x_1, \dots, x_N) = \prod_{n=1}^N p(x_n) \quad (2.1)$$

En cambio, para expresar la dependencia entre un grupo de observaciones ecuenciales, se puede utilizar entonces un modelo probabilístico llamado *modelo de Markov*, en el que usando la regla del producto se expresa la distribución conjunta de las observaciones de la siguiente manera:

Dar ejemplo de regla

$$p(x_1, \dots, x_N) = \prod_{n=1}^N p(x_n | x_1, \dots, x_{n-1}). \quad (2.2)$$

Ahora, si se asume que cada x_n es independiente de las observaciones anteriores excepto de la más reciente x_{n-1} , se tiene que la distribución conjunta se puede escribir de forma más sencilla utilizando el teorema de separación D^1 , puesto que

$$p(x_n | x_1, \dots, x_{n-1}) = p(x_n | x_{n-1}) \quad (2.3)$$

¹ Ver anexo

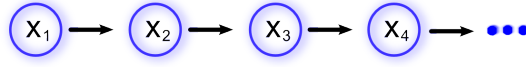


Figura 2.2: Modelo de Márkov de primer orden.

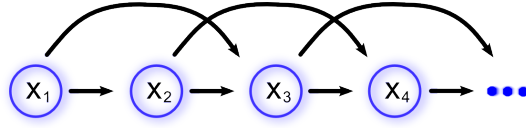


Figura 2.3: Modelo de Márkov de segundo orden.

A este modelo probabilístico se le conoce como cadena de Márkov de primer orden y la probabilidad conjunta de las observaciones está dada por:

$$p(x_1, \dots, x_N) = p(x_1) \prod_{n=2}^N p(x_n | x_{n-1}). \quad (2.4)$$

mientras que su modelo gráfico corresponde a la [Figura 2.2](#)

Aunque el modelo es mucho más general que el asumir que todas las observaciones son independientes entre sí; aún el modelo tiene ciertas restricciones.

De la misma manera en que se consideró para cuando x_n depende únicamente de la observación anterior, se puede extender esta idea para por ejemplo, que x_n dependa sólo de las dos observaciones previas. A este modelo se le conoce como cadena de Márkov de segundo orden; y siguiendo el mismo proceso, la distribución conjunta de las observaciones se puede escribir como

$$p(x_1, \dots, x_N) = p(x_1) \cdot p(x_2 | x_1) \prod_{n=3}^N p(x_n | x_{n-1}, x_{n-2}) \quad (2.5)$$

puesto que se tiene que $x_n \perp x_1, \dots, x_{n-3} | x_{n-1}, x_{n-2}$.

En este caso su modelo gráfico correspondiente es la [Figura 2.3](#), en el que se puede observar la dependencia de una x_n en específico únicamente con sus dos observaciones anteriores.

Se puede generalizar entonces para cualquier M con modelos de Márkov de M -ésimo orden, aunque al considerar demasiados estados previos se puede complicar de más el modelo; pues el número de parámetros implicados aumenta de forma exponencial al orden del modelo de Márkov.

Para evitar que el modelo se vuelva demasiado complejo, así como para no hacer alguna suposición a priori sobre de qué orden es del modelo generativo, se puede introducir una variable oculta, que permita cambiar el planteamiento del modelo.

Se considera entonces una variable latente z_n correspondiente a cada observación x_n , y entonces el conjunto de variables latentes forman una cadena de Márkov, como se muestra en la [Figura 2.4](#).

Se asumirá entonces

$$z_{n+1} \perp z_{n-1} | z_n \quad (2.6)$$

Y entonces, usando la regla de la cadena, la distribución conjunta para este modelo es la siguiente:

$$p(x_1, \dots, x_N, z_1, \dots, z_N) = p(z_1) \left[\prod_{n=2}^N p(z_n | z_{n-1}) \right] \prod_{n=1}^N p(x_n | z_n). \quad (2.7)$$

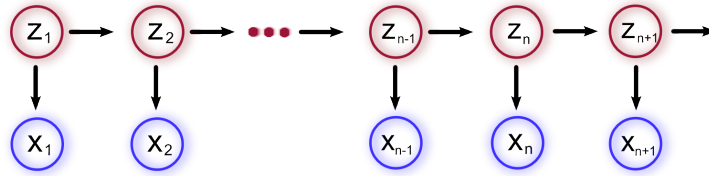


Figura 2.4: *Modelo oculto de Márkov.*

Para éste último modelo gráfico, si las variables latentes son discretas, entonces se le conoce como Modelo oculto de Márkov o HMM (*Hidden Markov Model*), y es justo el modelo que se utilizará para resolver la tarea de *speaker diarisation*.

2.2 MODELO OCULTO DE MÁRKOV

Un modelo oculto de Márkov, sigue siendo entonces un modelo para datos secuenciales, en los que además se introduce el concepto de una variable oculta de la cual dependen las observaciones que se tienen; y de forma más específica, esta variable oculta es discreta.

Usando un HMM entonces podemos modelar un proceso bi-variado discreto en el tiempo, con ciertas propiedades interesantes que se mencionarán más adelante.

Nuestro modelo HMM, se puede pensar como una mezcla de distribuciones en la que la densidad tiene una distribución dada por $p(x|z)$, es decir, la mezcla de los componentes está dada por las observaciones previas.

Se puede considerar entonces, que cada variable latente z_n tendrá una distribución multinomial discreta, que indicará cuál componente de la mezcla de distribuciones es la que ha generado la observación x_n . Para esto, se usará la notación 1-de- K , que corresponde a un conjunto de variables indicador $z_{nk} \in \{0, 1\}$, donde $k = 1, \dots, K$ señalando cuál de las K distribuciones generó a la variable x_n , i. e., si el componente generador de x_n fue la k -ésima mezcla, entonces $z_{nk} = 1$ y $z_{nj} = 0$ para todo $j \neq k$.

Ahora, se tiene que cada variable z_n depende únicamente de z_{n-1} , y puesto que las variables latente son vectores binarios de K dimensiones, entonces se tendría que la probabilidad condicional de $z_n | z_{n-1}$ se puede representar mediante una tabla o matriz, que se denotará como \mathbf{A} , y será referida como *matriz de probabilidades de transición*.

Los componentes de la matriz de transición se definen tal que $A_{jk} \equiv p(z_{nk} = 1 | z_{n-1,j} = 1)$, y puesto que son probabilidades, se cumple que $0 \leq A_{jk} \leq 1$ además de que $\sum_k A_{jk} = 1$. Considerando estas restricciones, entonces la matriz \mathbf{A} tiene $K(K-1)$ parámetros independientes.

Se puede escribir entonces la probabilidad condicional de una variable latente z_n dada la anterior variable latente z_{n-1} de la siguiente forma:

$$p(z_n | z_{n-1}, \mathbf{A}) = \prod_{k=1}^K \prod_{j=1}^K A_{jk}^{z_{n-1,j} \cdot z_{nk}} \quad (2.8)$$

Además, se tiene que considerar la variable latente inicial z_1 , puesto que ésta no tiene una variable latente anterior, la [Ecuación 2.7](#) no aplica; por lo que entonces la probabilidad marginal

de z_1 está representada por un vector de probabilidades π tal que $\pi_k \equiv p(z_{1k})$, por lo que se puede reescribir como sigue:

$$p(z_1 | \pi) = \prod_{k=1}^K \pi_k^{z_{1k}} \quad (2.9)$$

y además se cumple que $\sum_k \pi_k = 1$ por definición.

La matriz de transición también se puede llegar a representar como un grafo dirigido, si se consideran las entradas de la matriz A como los pesos de las aristas, y los nodos son cada uno de los posibles K estados. Por ejemplo, para el caso de una variable latente de $K = 3$ estados, se tendría el grafo de la [Figura 2.5](#).

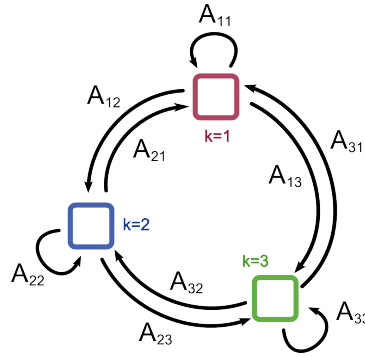


Figura 2.5: Modelo oculto de Márkov.

De la misma manera, el grafo correspondiente a la matriz de transición, se puede representar como una rejilla a través del tiempo, en la que se mantienen los vértices y aristas del grafo, pero además se introduce la noción del tiempo. Como se observa en la [Figura 2.6](#)

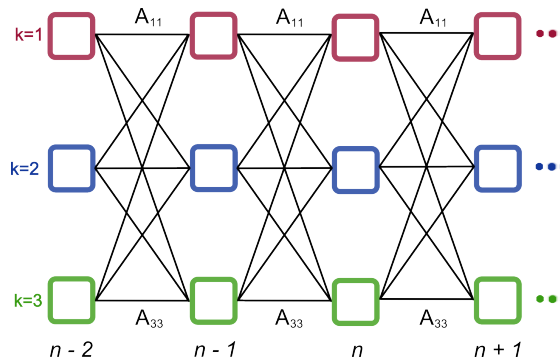


Figura 2.6: Modelo oculto de Márkov.

Por último, para completar el modelo se tiene que considerar además la distribución condicional de las variables observadas,

es decir $p(x_n | z_n, \phi)$ donde ϕ es un conjunto de parámetros específicos a la distribución de x . Se les conoce como probabilidades de emisión, y puede ser tanto una distribución discreta como una continua. Para el caso en que la probabilidad de emisión esté dada por una distribución discreta, entonces se tendrá una tabla de probabilidades.

Para calcular entonces esta probabilidad de emisión, se tiene tanto z_n como unos parámetros ϕ dados, por lo que entonces se tienen un vector de K valores para cada uno de los posibles estados del vector indicador z_n .

Se puede escribir entonces la probabilidad de emisión como sigue:

$$p(x_n | z_n, \phi) = \prod_{k=1}^K p(x_n | \phi_k)^{z_{nk}} \quad (2.10)$$

y entonces la probabilidad conjunta quedaría definida de la siguiente manera:

$$p(\mathbf{X}, \mathbf{Z} | \theta) = p(z_1 | \pi) \left[\prod_{n=2}^N p(z_n | z_{n-1}, \mathbf{A}) \right] \prod_{n=1}^N p(x_n | z_n, \phi) \quad (2.11)$$

donde $\mathbf{X} = \{x_1, \dots, x_N\}$, $\mathbf{Z} = \{z_1, \dots, z_N\}$ y $\theta = \{\pi, \mathbf{A}, \phi\}$ es el conjunto de parámetros requeridos por el modelo.

La intuición del modelo oculto de Márkov se puede entender más fácilmente si se revisa desde el punto de vista generativo. Primero, se muestrea la variable latente inicial z_1 , de acuerdo a las probabilidades π_k , así como su correspondiente x_1 . Luego, se debe escoger un z_2 . Para esto, si se supone que z_1 es igual a algún estado j , entonces usando la matriz de transición se muestrea z_2 con probabilidades A_{jk} para $k = 1, \dots, K$, y de igual manera, su correspondiente variable observada x_2 . Éste mismo proceso es el que se sigue para cada iteración en el tiempo, hasta que se forma completamente el modelo oculto de Márkov y se le conoce como *muestreo ancestral* y se suele usar para modelos con grafos dirigidos.

Si la matriz de transición es predominantemente diagonal, entonces en la secuencia de datos puede que un mismo estado i sea el que genere muchos puntos x_n , pues con poca probabilidad cambiará de un estado i a j . Este fenómeno es justo el que se espera para el caso de *speaker diarisation*, pues usualmente

una persona p_i hablará durante mucho tiempo, y luego cuando otra persona p_j toma la palabra, sucederá lo mismo.

3

HMM USANDO EM

Para resolver el modelo HMM, es decir, lograr estimar los parámetros que lo conforman, existen diferentes métodos y técnicas. En este trabajo de tesis se presentarán y platicará sobre dos de ellas. A saber, usando Expectation-Maximization, y usando simulaciones tipo Markov Chain MonteCarlo.

3.1 ESTIMACIÓN DE PARÁMETROS DEL HMM

Si se tiene un conjunto de datos observados $\mathbf{X} = \{x_1, \dots, x_N\}$, se pueden determinar los parámetros del HMM usando máxima verosimilitud. La función de verosimilitud se obtiene de la distribución conjunta (2.11) al marginalizar las variables latentes.

$$p(\mathbf{X}, \theta) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z} | \theta) \quad (3.1)$$

Sin embargo, la función obtenida presenta algunas dificultades, pues no se pueden con respecto a n , puesto que las variables latentes z_n dependen del estado anterior. Además, no es factible separar las sumas de estas N variables, pues para cada una se tendría que considerar cada uno de sus K posibles estados, y entonces el número de términos en la suma es del orden K^N , y crece exponencialmente con el largo de la cadena. Por esto y algunas otras razones, es que se descarta el estimar los parámetros del modelo por máxima verosimilitud.

Entonces, se buscará usar el algoritmo de *expectation-maximization*, en donde se hace una selección inicial de los parámetros que denotaremos como θ^{old} . Luego, en el primer paso del *expectation-maximization*, conocido como *E-step*, se toman estos parámetros para encontrar la probabilidad a posteriori de las variables latentes $p(\mathbf{Z} | \mathbf{X}, \theta^{\text{old}})$ que se usará para evaluar la esperanza de el logaritmo de una función de verosimilitud completa, que es

una función tanto de la primera estimación de θ^{old} así como de los nuevos parámetros θ .

La función de verosimilitud completa se define entonces como sigue:

$$\mathcal{Q}(\theta, \theta^{\text{old}}) = \sum_{\mathbf{Z}} p(\mathbf{Z} | \mathbf{X}, \theta^{\text{old}}) \ln p(\mathbf{X}, \mathbf{Z} | \theta). \quad (3.2)$$

Se introduce además cierta notación para simplificar las expresiones que ahora se usarán. Se usará $\gamma(z_n)$ para denotar la distribución marginal posterior de la variable latente z_n , y $\xi(z_{n-1}, z_n)$ para denotar la distribución conjunta posterior de dos variables latentes sucesivas, es decir:

$$\gamma(z_n) = p(z_n | \mathbf{X}, \theta^{\text{old}}) \quad (3.3)$$

$$\xi(z_{n-1}, z_n) = p(z_{n-1}, z_n | \mathbf{X}, \theta^{\text{old}}) \quad (3.4)$$

Considerando entonces que z_n es un vector binario, entonces se puede extender esta notación para cada uno de los componentes de la variable latente z_n , es decir, para denotar la probabilidad condicional $z_{nk} = 1$. Se tomará entonces la esperanza de tanto $\gamma(z_{nk})$ como $\xi(z_{n-1,j}, z_{nk})$ y entonces

$$\gamma(z_{nk}) = \mathbb{E}[z_{nk}] = \sum_{\mathbf{Z}} \gamma(\mathbf{z}) z_{nk} \quad (3.5)$$

$$\xi(z_{n-1,j}, z_{nk}) = \mathbb{E}[z_{n-1,j} \cdot z_{nk}] = \sum_{\mathbf{Z}} \gamma(\mathbf{z}) z_{n-1,j} \cdot z_{nk} \quad (3.6)$$

Si se sustituye $p(\mathbf{X}, \mathbf{Z} | \theta)$ de (2.11) en (3.2), así como usando las definiciones (3.5) y (3.6), y luego desarrollando el logaritmo, se obtiene:

$$\begin{aligned} \mathcal{Q}(\theta, \theta^{\text{old}}) = & \sum_{k=1}^K \gamma(z_{1k}) \ln \pi_k + \sum_{n=2}^N \sum_{j=1}^K \sum_{k=1}^K \xi(z_{n-1,j}, z_{nk}) \ln A_{jk} + \\ & \sum_{n=1}^N \sum_{k=1}^K \gamma(z_{nk}) \ln p(x_n | \phi_k) \end{aligned} \quad (3.7)$$

Por lo que entonces en el *E-step* se debe evaluar tanto $\gamma(z_{nk})$ como $\xi(z_{n-1,j}, z_{nk})$ de forma eficiente para luego continuar con la segunda etapa del algoritmo.

El segundo paso, también conocido como *M-step*, consiste en maximizar la unción $Q(\theta, \theta^{\text{old}})$ con respecto a cada uno de los parámetros $\theta = \{\pi, \mathbf{A}, \phi\}$ mientras que ahora $\gamma(z_{nk})$ y $\xi(z_{n-1,j}, z_{nk})$ se tratan como constantes.

Para maximizar entonces π , se deriva $Q(\theta, \theta^{\text{old}})$ con respecto a π_k , usando multiplicadores de Lagrange para la restricción de $\sum_k \pi_k = 1$, por lo que entonces se tiene:

$$\frac{\partial Q(\theta, \theta^{\text{old}})}{\partial \pi_k} = \frac{\partial}{\partial \pi_k} \left[\sum_{k=1}^K \gamma(z_{1k}) \ln \pi_k + \lambda \left(\sum_{k=1}^K \pi_k - 1 \right) \right]$$

y derivando y encontrando el valor de λ , para luego despejar π_k , se obtiene:

$$\pi_k = \frac{\gamma(z_{1k})}{\sum_{j=1}^K \gamma(z_{1j})} \quad (3.8)$$

mientras que para maximizar \mathbf{A} , se deriva entonces $Q(\theta, \theta^{\text{old}})$ con respecto a A_{jk} , considerando también las restricciones, es decir

$$\frac{\partial Q(\theta, \theta^{\text{old}})}{\partial \pi_k} = \frac{\partial}{\partial \pi_k} \left[\sum_{n=2}^N \sum_{j=1}^K \sum_{k=1}^K \xi(z_{n-1,j}, z_{nk}) \ln A_{jk} + \lambda \left(\sum_{k=1}^K A_{jk} - 1 \right) \right]$$

y de la misma manera, resolviendo para A_{jk} se obtiene

$$A_{jk} = \frac{\sum_{n=2}^N \xi(z_{n-1,j}, z_{nk})}{\sum_{l=1}^K \xi(z_{n-1,j}, z_{nl})} \quad (3.9)$$

Entonces, el algoritmo EM, debe ser inicializado escogiendo algunos valores para π y \mathbf{A} , considerando las restricciones que implican cada uno, pues representan ciertas probabilidades. Por tanto, se puede inicializar tanto π como \mathbf{A} con valores aleatorios, siempre y cuando cumplan con las restricciones propias de una probabilidad.

Por último, para estimar el parámetro θ , que en realidad corresponde a estimar los parámetros propios de la distribución de emisión, éstos dependen de la distribución propia de las variables observadas, aunque basta 1 observar que en (3.7) únicamente en el último término aparece θ en forma de una suma ponderada de $\ln p(x_n | \phi_k)$; y en el caso de que los parámetros ϕ_k sean independientes para los diferentes componentes de la mezcla o suma ponderada, entonces maximizar con respecto a esos parámetros se puede realizar de forma sencilla.

3.2 ALGORITMO *backward-forward*

Se presenta ahora un algoritmo que nos permite estimar de forma eficiente tanto $\gamma(z_{nk})$ como $\xi(z_{n-1,j}, z_{nk})$ que se requieren para el *E-step* del algoritmo de EM.

Para deducir el algoritmo, primero se deben tener en cuenta varias propiedades de una cadena de Márkov, que nos permitirán simplificar varios cálculos (Ver apéndice). Además, se debe tener en cuenta que éste desarrollo es general, puesto que es independiente de las probabilidades de emisión $p(x|z)$, y entonces no importa si las variables observadas son discretas o continuas, pues lo único que se requiere es poder evaluar $p(x_n|z_n)$ para cada z_n , que sabemos que son discretas.

Se puede comenzar por evaluar $\gamma(z_n)$, que es la probabilidad a posteriori $p(z_n|x_1, \dots, x_N)$ de z_n dado un conjunto de datos x_1, \dots, x_N . Entonces, usando el teorema de bayes se tiene que

$$\gamma(z_n) = p(z_n|\mathbf{X}) = \frac{p(\mathbf{X}|z_n)p(z_n)}{p(\mathbf{X})} \quad (3.10)$$

y usando la regla del producto, tenemos que

$$\gamma(z_n) = \frac{p(x_1, \dots, x_n, z_n)p(x_{n+1}, \dots, x_N|z_n)}{p(\mathbf{X})} = \frac{\alpha(z_n)\beta(z_n)}{p(\mathbf{X})} \quad (3.11)$$

donde usamos la siguiente notación:

$$\alpha(z_n) \equiv p(x_1, \dots, x_n, z_n) \quad (3.12)$$

$$\beta(z_n) \equiv p(x_{n+1}, \dots, x_N|z_n) \quad (3.13)$$

Se puede pensar $\alpha(z_n)$ como la probabilidad conjunta de observar toda una secuencia de datos hasta el momento n , además de el valor de la variable z_n ; mientras que $\beta(z_n)$ es la probabilidad condicional para una secuencia de datos desde un momento n hasta N , dado que se conoce z_n .

Primero, desarrollamos (3.12) como sigue:

$$\begin{aligned} \alpha(z_n) &= p(x_1, \dots, x_n, z_n) \\ \alpha(z_n) &= p(x_1, \dots, x_n|z_n)p(z_n) \\ \alpha(z_n) &= p(x_1, \dots, x_{n-1}|z_n)p(x_n|z_n)p(z_n) \\ \alpha(z_n) &= p(x_1, \dots, x_{n-1}, z_n)p(x_n|z_n) \end{aligned}$$

donde primero factorizamos $p(z_n)$ y luego $p(x_n | z_n)$ del resto, puesto que $x_n \perp x_1, \dots, x_{n-1} | z_n$; para luego volver a juntar $p(z_n)$ usando el Teorema de Bayes.

Luego, marginalizando sobre z_1 , podemos escribir

$$\begin{aligned}
 \alpha(z_n) &= p(x_n | z_n) \sum_{z_{n-1}} p(x_1, \dots, x_{n-1}, z_{n-1}, z_n) \\
 \alpha(z_n) &= p(x_n | z_n) \sum_{z_{n-1}} p(x_1, \dots, x_{n-1}, z_n | z_{n-1}) p(z_{n-1}) \\
 \alpha(z_n) &= p(x_n | z_n) \sum_{z_{n-1}} p(x_1, \dots, x_{n-1} | z_{n-1}) p(z_n | z_{n-1}) p(z_{n-1}) \\
 \alpha(z_n) &= p(x_n | z_n) \sum_{z_{n-1}} p(x_1, \dots, x_{n-1}, z_{n-1}) p(z_n | z_{n-1})
 \end{aligned} \tag{3.14}$$

y usando que $z_{n-1} \perp x_1, \dots, x_{n-1} | z_n$, podemos seguir el mismo procedimiento para llegar a (3.14); y por último, podemos usar la definición de (3.12) para el primer factor de la suma, llegando a que

$$\alpha(z_n) = p(x_n | z_n) \sum_{z_{n-1}} \alpha(z_{n-1}) p(z_n | z_{n-1}) \tag{3.15}$$

Con lo que se obtiene una forma recursiva para calcular $\alpha(z_n)$ a partir de $\alpha(z_{n-1})$ para cualquier n , excepto para $n = 1$; pues no se tiene definido un z_0 . Por esto mismo, podemos definir el caso inicial $\alpha(z_1)$ usando (3.12) y entonces resultaría:

$$\alpha(z_1) = p(x_1, z_1) = p(z_1) p(x_1 | z_1) \tag{3.16}$$

De la misma manera, para el caso de $\beta(z_n)$ desarrollando a partir de (3.13)

$$\begin{aligned}
 \beta(z_n) &= p(x_{n+1}, \dots, x_N | z_n) \\
 \beta(z_n) &= \sum_{z_{n+1}} p(x_{n+1}, \dots, x_N, z_{n+1} | z_n) \\
 \beta(z_n) &= \sum_{z_{n+1}} p(x_{n+1}, \dots, x_N | z_n, z_{n+1}) p(z_{n+1} | z_n) \\
 \beta(z_n) &= \sum_{z_{n+1}} p(x_{n+1}, \dots, x_N | z_{n+1}) p(z_{n+1} | z_n)
 \end{aligned}$$

donde primero agregamos la variable z_{n+1} y marginalizamos con respecto a ella, para luego factorizar $p(z_{n+1} | z_n)$. Después,

considerando que $x_{n+1}, \dots, x_N \perp z_n | z_{n+1}$, podemos simplificar la expresión.

A partir de ahí, podemos factorizar $p(x_{n+1} | z_{n+1})$ con lo que llegamos a

$$\beta(z_n) = \sum_{z_{n+1}} p(x_{n+2}, \dots, x_N | z_{n+1}) p(x_{n+1} | z_{n+1}) p(z_{n+1} | z_n) \quad (3.17)$$

donde usando (3.13) obtenemos de nuevo una forma recursiva para $\beta(z_n)$

$$\beta(z_n) = \sum_{z_{n+1}} \beta(z_{n+1}) p(x_{n+1} | z_{n+1}) p(z_{n+1} | z_n) \quad (3.18)$$

que en este caso dependería del valor de $\beta(z_{n+1})$ y aplicaría para cualquier n , excepto cuando $n = N$.

Para este caso, igual partiríamos de la definición en (3.11)

$$p(z_n | X) = \frac{p(X, z_N)_c(z_N)}{p(X)} \quad (3.19)$$

de donde se obtiene que

$$\beta(z_N) = 1 \quad (3.20)$$

3.2.1 Factor de escala

Para la implementación, se tiene que considerar un problema común del algoritmo de backward-forward: como ya se mencionó, en el proceso recursivo para calcular cada $\alpha(z_n)$ se utilizan los previamente calculados $\alpha(z_{n-1})$, además de unas multiplicaciones por un par de probabilidades. Puesto que cada probabilidad es por definición menor o igual a 1, y que esta operación se realiza iterativamente para cada estado n , se tiene entonces que los valores de $\alpha(z_n)$ decrecerán rápidamente, y llegará un momento en el que no se puedan representar en una computadora (por los límites tanto de la notación punto flotante como del doble punto flotante).

Comúnmente, cuando se tienen problemas de precisión, se suele tomar el logaritmo, y así se amplía el rango, evitando posibles underflows. Sin embargo, para el caso presentado, no es posible realizar esto, pues tanto para $\alpha(z_n)$ como $\beta(z_n)$ se manejan sumas de números pequeños, y entonces no tiene sentido

usar el logaritmo; pues el logaritmo no se aplicaría directamente sobre esos valores que pudieran ser pequeños; sino sobre la suma de ellos.

Se propone entonces manejar las probabilidades de forma reescalada. Originalmente, $\alpha(z_n)$ representa la probabilidad conjunta de todas las variables observadas x_1, \dots, x_n junto con la variable latente z_n . Se usará entonces $\hat{\alpha}(z_n)$ como la probabilidad condicional de la variable latente z_n dadas todas las observaciones antes mencionadas. Es decir:

$$\hat{\alpha}(z_n) = p(z_n | x_1, \dots, x_n) = \frac{\alpha(z_n)}{p(x_1, \dots, x_n)} \quad (3.21)$$

El orden entre las cantidades se mantendrá, puesto que se dividirá entre la probabilidad conjunta de las variables observadas $p(x_1, \dots, x_n)$.

Como en algún momento nos será útil regresar del espacio escalado al espacio original de las variables, es importante calcular estos factores de escala para cada una de las $\hat{\alpha}(z_n)$.

$$c_n = p(x_n | x_1, \dots, x_{n-1}) \quad (3.22)$$

y entonces podemos calcular el factor de escalamiento usando la regla del producto

$$p(x_1, \dots, x_n) = \prod_i^n c_i \quad (3.23)$$

y entonces, para obtener $\alpha(z_n)$ a partir de $\hat{\alpha}(z_n)$ se procede de la siguiente manera:

$$\begin{aligned} \alpha(z_n) &= p(x_1, \dots, x_n, z_n) = p(x_1, \dots, x_n) p(z_n | x_1, \dots, x_n) \\ &= \left(\prod_i^n c_i \right) \hat{\alpha}(z_n) \end{aligned} \quad (3.24)$$

De la misma manera, la forma recursiva que se había obtenido en (3.15) se puede reescribir de la siguiente manera:

$$c_n \alpha(z_n) = p(x_n | z_n) \sum_{z_{n-1}} \hat{\alpha}(z_{n-1}) p(z_n | z_{n-1}) \quad (3.25)$$

por lo que ahora, a cada paso de la recursión al calcular $\hat{\alpha}(z_{n-1})$, se debe calcular también c_n e ir almacenando los coeficientes que normalizan a $\alpha(z_n)$.

Ahora, para reescalar $\beta(z_n)$ usando los coeficientes c_n se tendría que

$$\beta(z_n) = \left(\prod_i^n c_i \right) \hat{\beta}(z_n) \quad (3.26)$$

donde

$$\hat{\beta}(z_n) = \frac{p(x_{n+1}, \dots, x_N | z_n)}{p(x_{n+1}, \dots, x_N | x_1, \dots, x_n)} \quad (3.27)$$

y por lo tanto tampoco presentaría problemas numéricos el cálculo de $\hat{\beta}(z_n)$, puesto que vuelve a ser un cociente de probabilidades. En este caso, de la condicional de las variables observadas desde x_{n+1}, \dots, x_N dada la variable latente z_n sobre la probabilidad condicional de las mismas variables observadas dadas todas las variables anteriores a x_{n+1} .

Entonces, para calcular recursivamente los valores de $\hat{\beta}(z_n)$, se deriva que

$$c_{n+1} \hat{\beta}(z_n) = \sum_{z_{n+1}} \hat{\beta}(z_{n+1}) p(x_{n+1} | z_{n+1}) p(z_{n+1} | z_n) \quad (3.28)$$

donde usamos los coeficientes de reescalaciemtno que ya habíamos calculado (y almacenado) junto con los valores de $\hat{\alpha}(z_n)$.

3.3 SELECCIÓN DE NÚMERO DE INTERLOCUTORES USANDO BOOTSTRAP

...

4

HMM USANDO MCMC

SPEAKER DIARISATION

Se conoce como *Speaker Diarisation* al problema de segmentación automática de audio a partir de la identificación de las diferentes personas que participan en una grabación. Esto se realiza generalmente identificando los segmentos que son más *homogéneos* y a partir de estos, identificar el número de personas que hablan en la grabación.

Se considera que cada persona posee ciertas características propias que la diferencian de los demás. Por ejemplo, tanto la tesitura como el timbre son rasgos que varían de persona a persona.

La idea básica cuando se busca abordar este problema es el de poder segmentar la señal de audio de acuerdo a los cambios que ocurren en ésta. Es decir, detectar por ejemplo las variaciones en la voz (que usualmente implican que alguien más está hablando) y luego, con todos los segmentos obtenidos, tratar de agruparlos según características similares; pues una misma persona tiene características propias en su voz.

5.1 APLICACIONES DE *speaker diarisation*

Por la misma naturaleza del algoritmo, cuando se realiza *speaker diarisation*, lo que se trata de inferir es cuántas personas hablan, y cuándo habla cada una de ellas; es decir, se identifica a un grupo n de personas, pero no se dice nada acerca de ellos o su identidad.

Por el mismo planteamiento del problema, las etiquetas asignadas a cada persona identificada en la grabación puede cambiar, pues no hay nada que nos permita asociar de manera no arbitraria una etiqueta a una persona en específico. Es decir, el orden en que se asignan las etiquetas puede cambiar, aunque la segmentación obtenida sea en esencia la misma.

La segmentación y agrupación de voz, es una parte importante de la transcripción de voz, así como también del reconocimiento de voz e identificación de personas que hablan.

Es de gran ayuda para la tarea de reconocimiento de voz, puesto que éste se basa en identificar palabras completas; por lo que al lograr segmentar una señal de audio de acuerdo a las personas que hablan, se tendrá entonces muy seguramente una buena segmentación tanto de palabras como oraciones completas.

En cuanto a la identificación de personas y alguna otra modelación acústica que se quiera realizar; es importante que los modelos que se entrenan usen segmentos de audio homogéneos (en este caso, que se tenga la certeza que corresponden a la misma persona), para que en realidad se esté modelando la voz de la persona, y no algo diferente a ella.

Como último ejemplo, la transcripción automática de voz, es el proceso en el que además de lograr identificar cuándo habla cada persona, se identifica de quién se trata realmente esta persona, y qué es lo que está diciendo. Ésto sirve por ejemplo, cuando se tiene una gran cantidad de grabaciones de audio; y se desea etiquetar de qué se habla en cada una de estas grabaciones.

5.2 PROCESAMIENTO DE SEÑAL

Antes de abordar de lleno el problema de *speaker diarisation*, se tiene que realizar cierto tratamiento a la señal de audio con la que se trabajará.

Es decir, a partir de la señal de entrada (que se considerará es digital) se tratarán de obtener vectores de características cada cierto tiempo, que representen de forma adecuada los rasgos que nos interesan distinguir.

Una vez que se tienen estos vectores, se les aplica un algoritmo de aglomeración para entonces obtener un conjunto de etiquetas que se podrían considerar como posibles estados o palabras de diccionario referentes a la señal de audio.

El proceso en detalle se especifica a continuación:

5.2.1 Esquema general del sistema

–Insertar dibujo y explicarlo–.

5.2.2 Pre-procesamiento de señal de audio

Se considera que se tiene una señal digital de voz, y que a partir de ésta se identificarán a las diferentes personas que hablan durante la grabación.

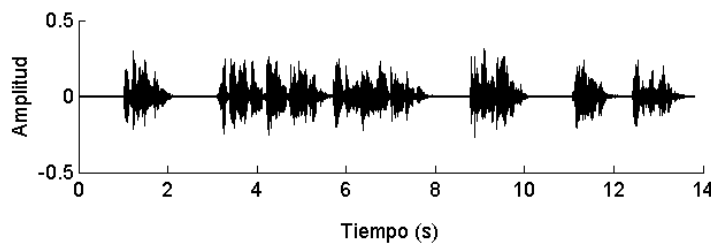


Figura 5.1: Señal original de audio.

El primer paso en el procesamiento de la señal, es tanto la detección como eliminación de silencios; pues éstos realmente no nos interesan para la modelación del sistema.

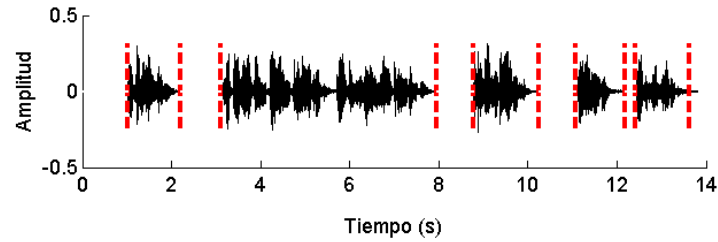
Para realizar entonces la detección de los silencios, en esta primer etapa y como un primer intento para la eliminación de silencios, se hace una detección básica de qué partes de la señal son mayormente silencios.

Para ésto, se utiliza una ventana móvil que se irá recorriendo a lo largo señal, y que irá calculando el total de energía de la señal dentro de la ventana. Se considerará entonces silencio aquellas partes de la señal cuyo total de energía esté debajo de un umbral específico.

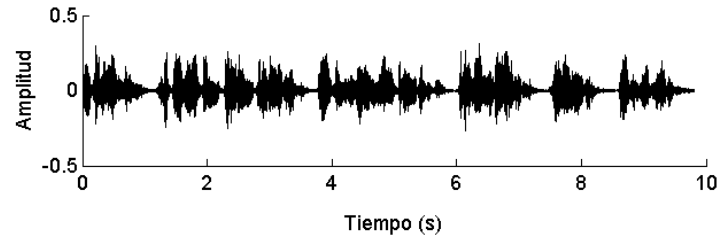
A partir de esta ventana se obtienen entonces las regiones que pertenecen tanto al silencio, como a la señal que realmente se desea modelar.

Como se observa en la [Figura 5.2](#) basta entonces con eliminar los segmentos que con baja energía y reagrupar la señal restante.

Nota: Tanto en ancho de la ventana, como el umbral para definir si será silencio se pueden ajustar dependiendo del tipo de señal.



(a) Señal original con silencio identificado.



(b) Señal procesada y recortada.

Figura 5.2: Identificación y eliminación de silencio en señal.

Con esto, se obtiene una señal en general más pequeña, y que se podría considerar sólo contiene realmente los datos que se desea modelar.

5.2.3 Obtención de características acústicas

Una vez que se tiene la señal ya sin silencios, se trata de buscar características propias de la señal de audio que nos permitan identificar de buena forma los cambios de voz a través de la señal.

Tanto la extracción como selección de la mejor representación paramétrica de la señal de audio es una importante tarea en el diseño de cualquier sistema relacionado al reconocimiento o procesamiento de señales de audio.

Para la tarea de *speaker diarisation*, se usarán los Coeficientes Cepstrales de Frecuencia Mel (MFCC), que son ampliamente utilizados por ejemplo en *speech recognition* entre otros procesos relacionados al procesamiento de voz; cuyo objetivo es comprimir la señal de audio eliminando la información que no es útil para análisis fonético.

Cabe mencionar, que originalmente los *MFCC* fueron diseñados para la tarea específica de reconocimiento de voz [referencia ICASSP 82], por lo que al momento de diseñarlos se trataba principalmente de que una misma palabra fuera parametrizada de la misma manera sin importar quién fuera quien la pronunciara.

Ésto va en contra del proceso requerido en *speaker diarisation*, puesto que se desea identificar a las diferentes personas que hablan, sin dar tanta importancia a qué es lo que están diciendo; por lo que la tarea de segmentación de señales de audio se vuelve un poco más complicada.

Para calcular los *MFCC*, se usa la Escala de Frecuencia Mel, que está espaciada de forma lineal en frecuencias bajas, mientras que aumenta su separación de forma logarítmica para frecuencias más altas. Este cambio de separación se realiza comúnmente a partir de los 1000Hz.

A partir de esta escala, se diseña un banco de filtros triangulares que después se usará (Ver [Figura 5.3](#)); y esto corresponde de forma similar a la que el oído (la cóclea) captura las características importantes del habla.

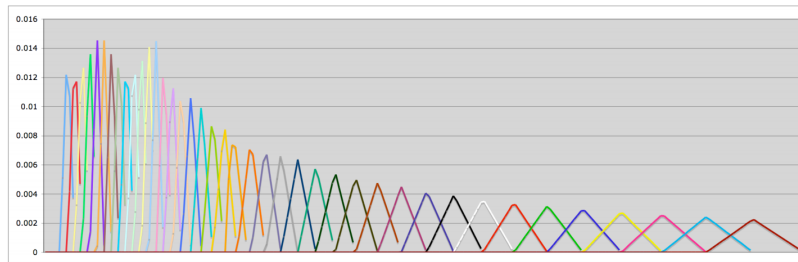


Figura 5.3: Banco de filtros triangulares en frecuencia Mel.

Puesto que el banco de filtros de Mel trabajan en la frecuencia; a la señal de audio se le aplica la transformada de Fourier, y entonces a ésta es a quien se le aplica el banco de filtros.

Sea pues [Figura 5.2](#) la señal procesada sin los silencios, la respuesta que se obtiene al aplicar la FFT y luego el banco de filtros de [Figura 5.3](#) se puede observar en la figura [Figura 5.4](#).

La dimensionalidad de la respuesta al banco de filtros dependerá de la misma contrucción del banco de filtros (tanto el número de canales que se usarán, como el tamaño de ventana que se usará para las convoluciones); pero en general se tendrá

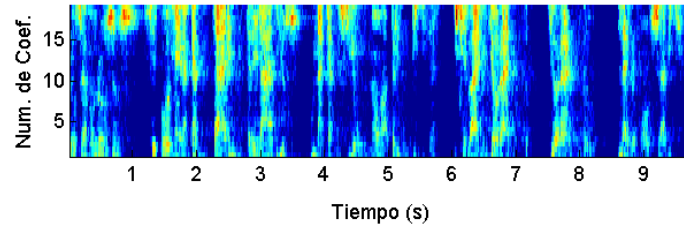


Figura 5.4: *Respuesta al banco de filtros.*

que es muy alta; por lo que es conveniente tratar de disminuir la dimensionalidad de estos datos.

Para esto, la respuesta obtenida del banco de filtros se le aplicará la DCT (Transformada Discreta del Coseno), para tratar de concentrar la energía en ciertos componentes (los primeros n coeficientes), y descartar los restantes.

Después de este proceso, nuestros datos se podrían representar de la siguiente manera (Figura 5.5) que son los MFCC que antes habíamos mencionado.

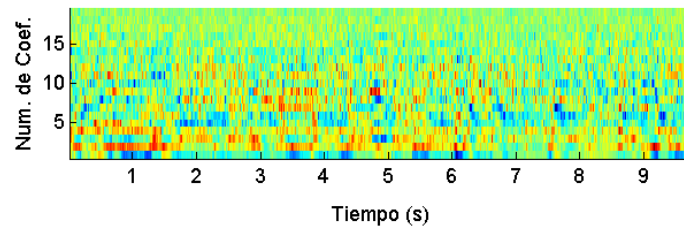


Figura 5.5: *Mel Frequency Cepstrum Coefficients.*

Por último, para los modelos que usaremos, se necesitan que las características estén de cierta forma *discretizadas*, es decir, no nos es útil el tener para cada observación en el tiempo un vector de características; sino que necesitamos una etiqueta o clase para cada observación.

Para esto, podemos utilizar diferentes métodos tanto de reducción de dimensionalidad como de agrupación/clasificación. Como primer idea, utilizaremos el método de *k-means* para agrupar los vectores de acuerdo a su cercanía en el espacio euclidiano.

Se tendría entonces el siguiente resultado para nuestra matriz de MFCC obtenida: Cabe aclarar que usamos una variante

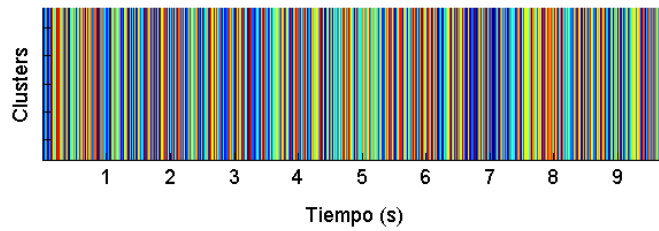


Figura 5.6: MFCC agrupados con *k-means++*.

del algoritmo original de *k-means*, que se llama *k-means++* y que propone una mejor inicialización para obtener mejores resultados.

Ahora sí, con nuestro vector de etiquetas correspondiente a la señal de audio, se podrá aplicar un modelo y tratar de inferir los parámetros que le correspondan.

5.3 RESOLVER USANDO MODELO HMM USANDO EM

Para las pruebas que se han realizado hasta ahora, se han usado grabaciones creadas sintéticamente; esto es, usando un sintetizador de voz. El hacer las grabaciones con voces sintéticas nos permite tener un mayor control sobre la calidad de la grabación (no hay ruidos aditivos o interferencias), además de poder generar muchos más casos de prueba sin tantas complicaciones.

Para las primeras pruebas, se utilizó la grabación correspondiente a un fragmento de la obra "La vida es sueño", del dramaturgo Pedro Calderón de la Barca.

Mediante bootstrap, se busca hacer la elección del número correcto de speakers identificados; por lo que para estos casos se mostrarán las segmentaciones obtenidas usando desde 2 hasta 4 speakers; además de comparar el ground truth.

En la [Figura 5.7](#) se muestran los parámetros estimados a forma de matrices en falso color: la matriz inicial o a priori, la matriz de transiciones entre speakers, la matriz de emisión

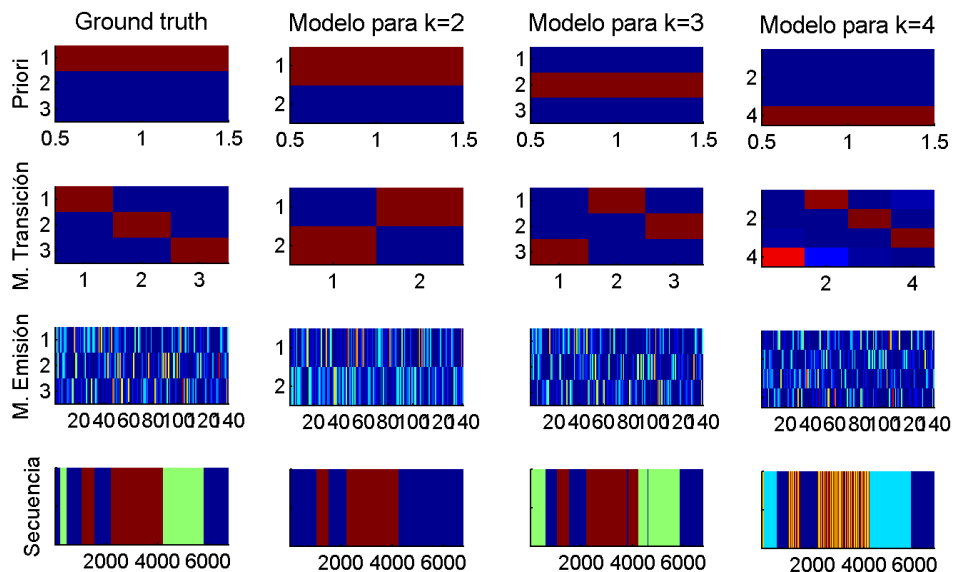


Figura 5.7: *Parámetros estimados para Prueba 1*

para cada speaker y todo el diccionario de palabras, así como la secuencia recuperada.

En la [Figura 5.8](#) se muestra la secuencia recuperada, como una serie de tiempo en donde además se marcan en rojo los errores al asignar a una persona equivocada.

Se puede observar como para el caso en que el modelo se ajusta para 3 personas, la secuencia recuperada en esencia es la misma, exceptuando algunos saltos que se presentan a veces; pero en general logra detectar de forma correcta cada que hay un cambio de interlocutor en la grabación original.

Para la segunda prueba, se utilizó un segmento del guión teatral "The King Lear", del escritor William Shakespeare.

Se muestran a continuación los resultados obtenidos de la misma manera.

En estas gráficas se observa de igual manera cómo se logró estimar de forma correcta el número de interlocutores. Cuando se aplica el modelo para un número menor que el real, se detectan las inconsistencias y aumenta la tasa de error; así como cuando se ajusta el modelo para un número mayor de personas, se sobreajusta el modelo y se marcan como errores.

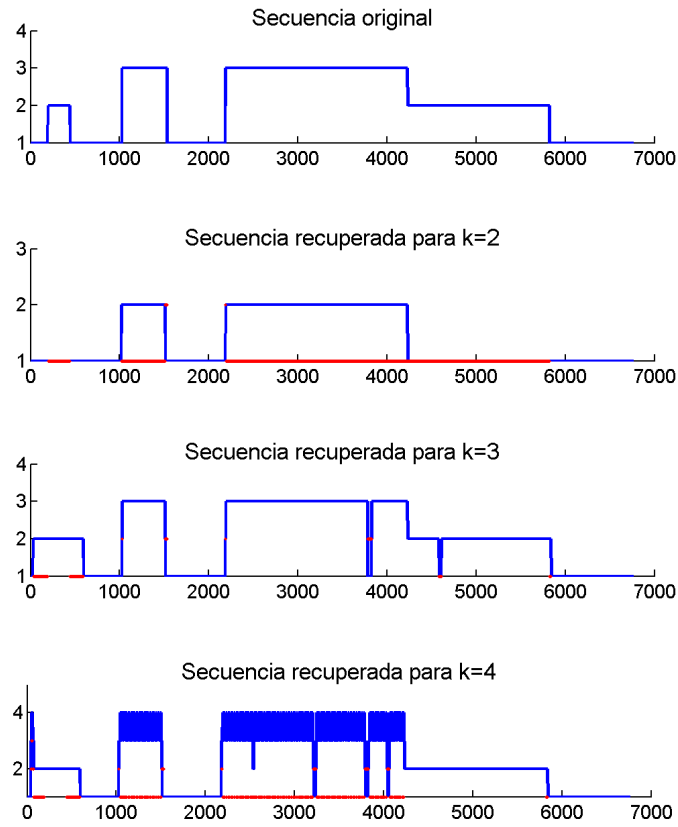


Figura 5.8: *Secuencia recuperada para Prueba 1*

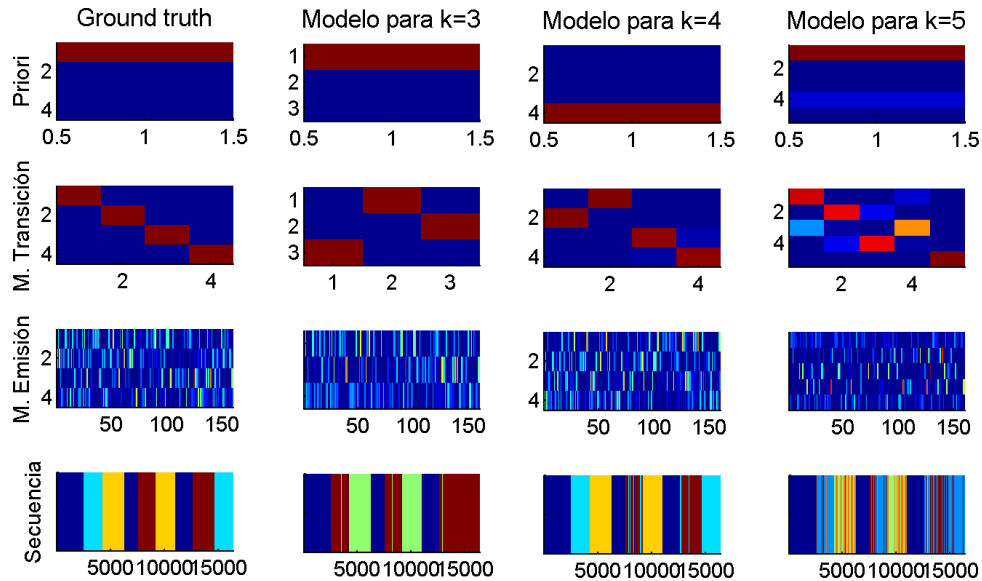


Figura 5.9: *Parámetros estimados para Prueba 2*

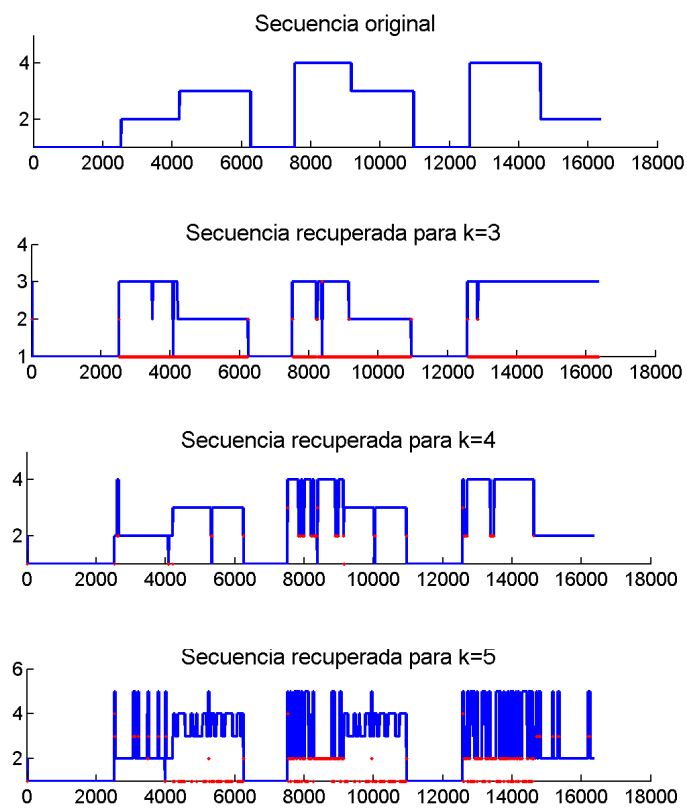


Figura 5.10: *Secuencia recuperada para Prueba 2*