2018

MODELADO DE CONECTIVIDAD CEREBRAL EN REPOSO CON MODELOS AUTOREGRESIVOS MULTIVARIADOS Y ESTIMACIÓN EFICIENTE DE SU ORDEN

GUERRA PINTO, VICTOR DANIEL

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
Departamento de Electrónica
Valparaíso - Chile

"MODELADO DE CONECTIVIDAD CEREBRAL EN REPOSO CON MODELOS AUTOREGRESIVOS MULTIVARIADOS Y ESTIMACIÓN EFICIENTE DE SU ORDEN"

VICTOR DANIEL GUERRA PINTO

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRÓNICO CON MENCIÓN EN ELECTRÓNICA INDUSTRIAL

PROFESOR GUÍA: MATÍAS ZAÑARTU
PROFESOR CORREFERENTE: ALEJANDRO WEINSTEIN

DICIEMBRE - 2017

Página intencionalmente dejada en blanco

AGRADECIMIENTOS

AGRADEZCO A LA WACHITA RUSA QUE INVENTÓ SCIHUB Y A LOS MENS QUE HICIERON LOS TOOLBOX QUE ME SALVARON LA TESIS

Resumen

En el area de modelado de funciones cognitivas, uno de los problemas de interés es el modelado de conectividad cerebral en estado de reposo, donde se busca poder identificar patrones de actividad cerebral, cuando el sujeto experimental no se encuentra haciendo ninguna tarea cognitiva en particular, esta indentificación normalmente es realizada por un experto que pueda identificar los patrones de actividad viendo por ejemplo, un electroencefalograma (EEG), esta tarea es hecha con una muy baja resolución temporal, con un etiquetado de señales por segmentos de 30 segundos (a este proceso de etiqutado por segmentos se le dice segmentación). Una forma de facilitar el estudio de modelos cerebrales es hacer esta identificación automáticamente. Actualmente se han propuesto varios algoritmos y modelos en la literatura de series de tiempo y machine learning para resolver problemas de segmentación de señales, entre estos, los modelos escondidos de Markov Autoregresivos Multivariados (HMM-MAR en inglés), han mostrado en la literatura que reuelven de manera eficaz la segmentación. Aquí se muestran métodos para hacer la estimación de estos modelos de Markov y resolver el problema de segmentación, probandolos con distintos modelos verificando su desempeño en un entorno sintético para posteriormente proponer un experimento con mediciones fisioógicas reales.

Abstract

In the fields of functional connectivity modelling, a problem of interest is to model brain connectivity in resting state, where is sought to identify patterns of brain activity, when the experimental subject isn't doing any cognitive task in particular, this identification, is usually performed by an expert who can identify the activity patterns by seeing, for example, an electroencephalogram (EEG), this task is done with a very low temporal resolution, with a labeling of signals per 30 second segments (this process of labeling by segments is called segmentation). One way to facilitate the study of brains models is to make this identification automatically. Currently, several algorithms and models have been propose in the machine learning and time series and machine learning literature to solve problems of signal segmentation, among then, the hidden Markov multivariate Autoregressive (HMM-MAR)) has shown in the literature that it effectively solves the segmentation. Here, different methods to estimate these Markov models and solving the segmentation problem, testing them with different models, verifyng their performance in a synthetic environment to propose an experiment with real physiological measeurements.

***Palabras Clave**: Serie de tiempo, señal, segmentación, modelo escondido de markov, modelo autoregresivo*

Índice general

Índice de figuras

Índice de cuadros

Página intencionalmente dejada en blanco

Capítulo 1

Introducción

En el estudio de señales cerebrales, uno de los tópicos de interés son la redes conectadas en estado de reposo, donde se busca reconocer como están conectadas las distintas areas del cerebro cuando este no se encuentra realizando ninguna tarea cognitiva en particular, e identificar los cambios en estas conecciones a lo largo de este periodo de reposo, dicho de otra manera, es identificar estas redes conectadas en reposo (o “Resting State Networks").

Normalmente esta identificación es hecha por un experto, por ejemplo cuando se estudia una polisomnografía [1], la segmentación se hace manualmente para identificar disintas fases de sueño: fase de vigilia (totalmente despierto), sueño ligero, sueño profundo, sueño MOR (22), el problema de esta segmentación[2] es de muy pobre resolución temporal (normalmente se etiquetan las señales en segmentos de 30 segundos). Este mismo problema es recurrente cuando se requere detectar diferencias de conectividad o actividad cerebral que ocurren en ventanas muy pequeñas de tiempo (diferencia que podría no ser percibida por estudiosos del tema).

Para solucionar problemas de segmentación de señales, en la literatura se propone usar Modelos Escondidos de Markov (HMM, por su sigla en inglés)(2) que es un modelo que permite hace un modelado de una señal, modelando las características de distintos estados que presenta la señal, las probabilidades de pasar a un estado a otro y las probabilidades de tener cierta observación en algún estado en particular. Este esquema conveniente para el problema de reconocimiento de etapas de sueño, ya que, como el modelo propone, se deben considerar las relaciones (probabilidades de transición) entre estados consecutivos, como en los estudios de sueño donde se consideran relaciones de transición entre distintas etapas de sueño(23). Un factor importante a considerar de estos modelos es que a través de estos es posible hacer una identificación de estados, observación por observación, con lo que se puede hacer una segmentación automática con una resolución muy superior a que se tiene con una identificación manual de estados.

[1] tipo de estudio del sueño que incluye mediciones de de EEG, de movimiento ocular (EOG), activación muscular (EMG) y cardiaca (ECG).

[2] A partir de la lectura de una señal, separarla en distintos segmentos y etiquetar estos en clases (o estados)

Los HMM han sido propuestos para resolver distintos problemas de segmentación de señales en distintas areas: como en procesamiento de voz (2); en clasificación de señales de EEG para interfaces cerebro máquina(24) y de mediciones fNIRS[3] también para BCI(21); en bioinformática para secuenciación de estructuras de ADN (16); No limitandose a areas biológicas, también se han usado estos modelos en aplicaciones de econometría (15) y meteorología (25).

Uno de los métodos para la estimación de estos modelos es descrito en (2), donde L. Rabiner hace una descripción teórica de los distintas configuraciones de modelos HMM, pasando por modelos probabilisticos de observación discretos, continuos Gaussianos y autoregresivos, bajo un esquema de maxima verosimilitud, donde se buscan el conjunto de parámetros que maximiza una medida de confianza de los datos. Otro esquema de optimización para estos modelos es el de aprendizaje variacional, descrito por I. Rezek y S. Roberts en (7), se muestran la formulación y optimización de las distribuciones de los parámetros de los HMM, pasando también por distintos modelos de observación continuos (gaussianos y autoregresivos).

Los algoritmos de aprendizaje variacional son comparativamente superiores en términos de convergencia y cálculo computacional, ya que al buscar las distribuciones de probabilidad de los parámetros, se acota y suaviza el espacio donde se está buscando el óptimo, haciendo que la convergencia a algún óptimo sea más rápida y evite caer en óptimos locales (7).

Independientemente del modelo a utilizar, uno de los problemas con los que hay que lidiar al modelar un sistema, es la selección de orden, que es la busqueda de la cantidad adecuada de parámetros para explicar un set de datos a modelar de forma parsimoniosa, es decir, de forma de que no sea un modelo muy complejo (tenga un gran numero de parámetros), pero pueda explicar bien los datos, en terminos de verosimilitud de los datos, o alguna otra métrica de confianza. Para modelos de series de tiempo como los modelos autoregresivos existen criterios de información para discriminar y seleccionar la complejidad (29). En el contexto de los HMM autoregresivos, el esquema de optimización variacional, su formulación naturalmente da una forma de seleccionar el orden de la parte autoregresiva, y adicionalmente escoger la cantidad de estados del modelo de Markov (7).

Entre los distintos modelos de observación, se propone usar los modelos HMM autoregresivos, dado que estos son mayormente utilizados para segmentar señales y hacer analisis de conectividad de señales neuronales (4) (8) (9) (10) (11), esto se debe a que estos modelos HMM integran las carácteristicas de los modelos autoregresivos multivariados(27), que por su estructura, permiten guardar dependencias o correlaciones entre distintas variables, o en el caso de señales biológicas, entre distintas regiones del cerebro, por ejemplo, y también guarda

[3]Functional Near-Infrared Spectroscopy (Espectroscopia funcional Cercana al Infrarojo), donde se hace una medición de la actividad neuronal indirectamente a través de la observación de cambios en oxigenación de la sangre, usando rayos cuyo espectro de luz está en el rango del espectro infrarojo

caracteristicas espectrales, que son también utiles al momento de estudiar señales neuronales (28).

En este trabajo se propone el uso del aprendizaje variacional para el ajuste de los modelos, por sus ventajas con respecto al esquema de optimización de máximo verosimil, utilizando modelo de observacion continuos autoregresivos por la información espectral y de conectividad que entregan estos modelos. Y adicionalmente usar este mismo esquema de optimización para selección de modelos (7), ya que se tiene la ventaja de tener una métrica de comparación para seleccionar ordenes de modelos de la parte autoregresiva del HMM sin hacer calculos adicionales, como hay que hacerlos en el caso de utilizar los criterios de información mencionados en (29).

Para hacer selección de orden de modelos, se propone utilizar el esquema mostrado en (7), dado que este, sin cálculos adicionales de medidas de selección (más detalles en el cap. 4), tanto para la selección del orden del modelos autoregresivo a utilizar, como para escoger adecuadamente la cantidad de estados del modelo de Markov.

En este escrito se hace una revisión de los modelos escondidos de Markov, en su versión de optimización de maximo verosimil en el capítulo 3, y de aprendizaje variacional en el capítulo 4, pasando por los distintos modelos de observación, se incluye también en el capítulo 2 una revisión de los modelos autoregresivos para introducir conceptos necesarios de series de tiempo para los comprender mejor los HMM autoregresivos. Posteriormente en el capítulo 5 se muestra un esquema de simulación para poder comparar los distintos esquemas de optimización para un modelo HMM continuo gaussiano y se mostrar su desempeño para un modelo HMM continuo autoregresivo para probar cuan bien se hace la selección de modelos, y comparando la calidad de la segmentación de señales con distintas configuraciones de modelos HMM, utilizando un esquema de simulación con una señal artificial que figura ser una señal neuronal.

Desarrollo del tema

Capítulo 2

Modelos Autoregresivos

En estadística y procesamiento de señales, los modelos autoregresivos (AR), se utilizan los modelos AR para hacer predicciones a partir de observaciones pasadas (19)(20), bajo la premisa de que los procesos descritos con este tipo de modelo poseen dependencia con los valores de su pasado. En este capitulo se describen los procesos autoregresivos pasando por su versión univariada y multivariada (con sus ecuaciones para la estimación de parámetros).

2.1. Caso Univariado *AR(p)*

Los modelos autoregresivos de orden p $AR(p)$, son modelos basados en la idea de que el valor actual de una señal x_t se puede explicar como una combinación (lineal) de sus p valores anteriores $x_{t-1}, x_{t-2}, ..., x_{t-p}$ y una señal de error w_t (ecuación 2.1).

$$x_t = -a_1 x_{t-1} - a_2 x_{t-2} - ... - a_p x_{t-p} + w_t,\ w_t \sim N(0, \sigma^2) \tag{2.1}$$

los distintos a_i, representan cuanto depende la señal x_t de su valor pasado x_{t-i}.

Para futuros cálculos, será más facil tener a mano una representación del proceso AR(p) para una ventana de tiempo, en una forma matricial compacta, se puede lograr armando la matriz M, como un vector de filas, que en su *i-ésima* fila, tendrá a $\bar{x_i} = (x_{i-1}, x_{i-2}, ..., x_{i-p})$. Ilustrando la idea para el caso de una ventana de largo arbitrario N, para un modelo AR(4) , la matriz M asociada sería:

$$M = -\begin{pmatrix} x_4 & x_3 & x_2 & x_1 \\ x_5 & x_4 & x_3 & x_2 \\ \vdots & \vdots & \vdots & \vdots \\ x_{N-1} & x_{N-2} & x_{N-3} & x_{N-4} \end{pmatrix} \tag{2.2}$$

la matriz comienza desde $\bar{x_5} = (x_4, x_3, x_2, x_1)$*, para omitir las muestras anteriores a* x_1*, que no son consideradas en la formulación del modelo*

definiendo el vector de coeficientes a_i, $\mathbf{a} = (a_1\, a_2 \ldots a_p)^T$, se puede escribir el proceso AR ejemplo anterior para una ventana de señal $\mathbf{X} = -(x_5\, x_6 \ldots x_N)^T$,con su señal de error $\mathbf{W} = (w_5\, w_6 \ldots w_N)$ como en la ec. 2.3 o 2.4.

$$\begin{pmatrix} x_5 \\ x_6 \\ \vdots \\ x_N \end{pmatrix} = - \begin{pmatrix} x_4 & x_3 & x_2 & x_1 \\ x_5 & x_4 & x_3 & x_2 \\ \vdots & \vdots & \vdots & \vdots \\ x_{N-1} & x_{N-2} & x_{N-3} & x_{N-4} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix} + \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{pmatrix} \tag{2.3}$$

$$\mathbf{X} = M\mathbf{a} + \mathbf{W} \tag{2.4}$$

Visto como en la ec. 2.4, el modelo AR(p) no es más que un caso particular de la regresión lineal multivariada, donde los coeficientes de regresión corresponden al vector **a**. La estimación del vector de coeficientes **a**, se obtiene al minimizar el error cuadrático medio **ECM** (ec. 2.5) con respecto al vector **a**, cuya solución viene dada por la ec. 2.6.

$$\mathbf{ECM} = (X - M\mathbf{a})^T (X - M\mathbf{a}) \tag{2.5}$$

$$\hat{\mathbf{a}} = (M^T M)^{-1} M^T \mathbf{X} \tag{2.6}$$

Una vez obtenido $\hat{\mathbf{a}}$, se puede tener la predicción de la señal en el instante t como $\hat{x_t} = \tilde{x_t}\mathbf{a}$, se puede estimar la varianza de la señal de error de predicción $\hat{w_t} = x_t - \hat{x_t}$ como en la ec. 2.7

$$\sigma_t^2 = \frac{1}{N - n_p} (x_t - \hat{x_t})^T (x_t - \hat{x_t}) \tag{2.7}$$

n_p corresponde a la cantidad de grados de libertad que tiene el modelo (la cantidad de parámetros a estimar)

2.2. Caso Multivariado *mAR(p)*

El modelo autoregresivo multivariado *mAR(p)* consiste en modelar el valor de k múltiples señales $X_t = (x_{1,t}\ x_{2,t}\ \ldots\ x_{k,t})$ como una combinación lineal de sus valores pasados (hasta p instantes anteriores) y una señal de error $W_t = (w_{1,t}\ w_{2,t}\ \ldots\ w_{k,t})$ (ec. 2.8).

$$X_t = -A_1 X_{t-1} - A_2 X_{t-2} - \ldots - A_p X_{t-p} + W_t,\ W_t \sim N(0, \Sigma) \tag{2.8}$$

Cada una de estas matrices A_i tienen la forma

$$A_i = \begin{pmatrix} a_{11,i} & a_{12,i} & \ldots & a_{1k,i} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k1,i} & a_{k2,i} & \ldots & a_{kk,i} \end{pmatrix},$$

y guardan las interdependencias entre las distintas señales $x_{1,t}, x_{2,t}, ..., x_{k,t}$.

la ecuación 2.8 se escribe de una forma más compacta (ec. 2.9) definiendo el vector fila aumentado $\tilde{X}_t = -(X_{t-1}\ X_{t-2}\ \ldots\ X_{y-p}) \in \mathbb{R}^{1\times p*k}$ y la matriz $\mathbf{A} = (A_1\ A_2\ \ldots\ A_p)^T \in \mathbb{R}^{k\times k*p}$. Para el caso particular de un modelo *mAR(2)* de 2 señales se tiene:

$$\begin{pmatrix} x_{1,t} \\ x_{2,t} \end{pmatrix}^T = -(x_{1,t-1}\ x_{2,t-1}\ x_{1,t-2}\ x_{2,t-2}) \left(\begin{array}{cc:cc} a_{11,1} & a_{12,1} & a_{11,2} & a_{12,2} \\ a_{21,1} & a_{22,1} & a_{21,2} & a_{22,2} \end{array}\right)^T + \begin{pmatrix} w_{1,t} \\ w_{2,t} \end{pmatrix}^T \tag{2.9}$$
$$X_t = \tilde{X}_t \mathbf{A} + W_t$$

Esta ecuación es la representación del modelo *mAR* para un instante t. Para expresar el proceso *mAR* de una ventana de señal $\mathbf{X} = (X_p\ X_{p+1}\ ...\ X_N)^T$ (con su respectiva señal de error $\mathbf{W} = (W_p\ W_{p+1}\ ...\ W_N)^T$, se define la matriz $\tilde{M}$ que en su *t-ésima* fila, tendrá el vector $\tilde{X}_t$. Para un modelo *mAR(2)* de 2 señales queda como en la ec. 2.10.

$$\begin{pmatrix} X_3 \\ X_4 \\ \vdots \\ X_N \end{pmatrix} = - \begin{pmatrix} x_{1,2} & x_{2,2} & x_{1,1} & x_{2,1} \\ x_{1,3} & x_{2,3} & x_{1,2} & x_{2,2} \\ \vdots & \vdots & \vdots & \vdots \\ x_{1,N-1} & x_{2,N-1} & x_{1,N-2} & x_{2,N-2} \end{pmatrix} \left(\begin{array}{cc:cc} a_{11,1} & a_{12,1} & a_{11,2} & a_{12,2} \\ a_{21,1} & a_{22,1} & a_{21,2} & a_{22,2} \end{array}\right)^T + \begin{pmatrix} W_3 \\ W_4 \\ \vdots \\ W_N \end{pmatrix}$$
$$\mathbf{X} = \tilde{M}\mathbf{A} + \mathbf{W} \tag{2.10}$$

Con el modelo expresado como en la ec. 2.10, se puede estimar la matriz $\mathbf{A}$, con la solución de la regresión lineal multivariada (ec. 2.11).

$$\hat{\mathbf{A}} = (\tilde{M}^T\tilde{M})^{-1}\tilde{M}^T\mathbf{X} \tag{2.11}$$

Con la matriz $\hat{\mathbf{A}}$, se puede hacer la estimación de las señales en el instante t como $X_t = \tilde{X}_t\hat{\mathbf{A}}$ La matriz de covarianza del error de predicción $\hat{W}_t = X_t - \hat{X}_t$, al igual que en el caso univariado, viene dada por:

$$\Sigma_t = \frac{1}{N-np}(X_t - \hat{X}_t)^T(X_t - \hat{X}_t) \tag{2.12}$$

Capítulo 3

Hidden Markov Models (HMM)

Los HMM (llamados así por su sigla en inglés para “Modelos Escondidos de Markov"), son una forma de modelar datos secuenciales (una serie de tiempo). Un HMM asume que existe un un modelo de markov detrás de los datos observados, pero que este modelo no se conoce. Dicho de otra manera, solo se conocen los datos observados, pero no se tiene información de la secuencia de estados de la cadena de markov que condicionan a cada observación.

En este capítulo se hace un revisión completa de estos modelos, partiendo por describir un modelo de Markov, siguiendo por la formulación distintas variedades de los modelos escondido de Markov, y las ecuaciones para uno de los métodos (Expectation-Maximization(2)) para optimizar los parámetros de dichos modelos.

3.1. Modelo de Markov

Antes de comenzar con los modelos escondidos de markov, se describen los modelos de Modelos de Markov (a secas) para introducir las nociones de probabilidades utilizadas en el HMM.

Un Modelo de Markov es un tipo de modelo estadístico que describe una secuencia de estados. Suponer que durante un proceso, a cada instante, existe un cambio de estado (inclusive permanencia en un mismo estado) entre distintos N posible estados $S_1, S_2, ..., S_N$ estados según un conjunto de probabilidades asociadas a estos mismos.

Cada uno de estos estados (en cada instante de tiempo t) es denotado por q_t, y para el caso de una cadena de markov discreta, de primer orden, cumple la propiedad de Markov (eq. 3.1), que dice que la probabilidad de observar un estado en particular S_j, depende únicamente del estado anterior, ignorando la historia que tenga antes de eso).

$$P(q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, ...) = P(q_t = S_j | q_{t-1} = S_i) \tag{3.1}$$

Además, en proceso el de markov, estas probabilidades son independientes del tiempo y para los supuestos del modelo se consideran constantes durante el desarrollo del proceso, la información de las probabilidades de moverse entre estados, del estado S_i al S_j, por ejemplo, se concentra en la matriz de transción de probabilidades $\pi_{t_{t-1}} = \{\pi_{t_{t-1}ij}\}$ como:

$$\pi_{t_{t-1}ij} = P(q_t = S_j | q_{t-1} = S_i) \tag{3.2}$$

y las probabilidades de transición deben cumplir con las restricciones estocásticas de normalidad y no-negatividad (ec. 3.3).

$$\begin{aligned} \pi_{t_{t-1}ij} &\leq 1 \\ \sum_{j=1}^{N} \pi_{t_{t-1}ij} &= 1 \end{aligned} \tag{3.3}$$

En la figura 3.1 se tiene un diagrama donde se ven los pasos entre estados y sus probabilidades de transición en un modeo de 5 estados.

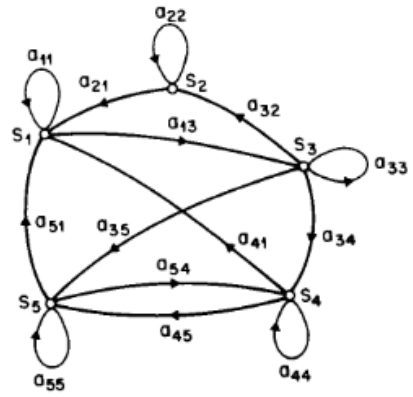

Figura 3.1: Diagrama de una cadena de markov de 5 estados (S_1 a S_5) con probabilidades de transición entre pares de estados ij denotadas por a_{ij}.

Un ejemplo típico para describir estos modelos, es el de un modelo de markov de 3 estados, para modelar el clima, en el cual se asume que una vez por día el clima observado (el estado) puede ser: lluvioso (estado S_1), nublado (estado S_2) o soleado (estado S_3).

Como se trata de un modelo de markov se dice que el clima observado en un día t, está condicionado unicamente por alguno de los estados recién mencionados con probabilidades de observación descritas en la matriz de transición $\pi_{t_{t-1}}$ junto al diagrama de la figura 3.2 que describe los saltos entre estados, con sus probabilidades.

La propiedad de markov permite escribir la probabilidad de observar una cierta secuencia de estados dado el modelo como un producto simple entre las probabilidades de transición de los estados, asumiendo que la probabilidad de encontrarse en cualquier estado j (eq. 3.4) se conoce.

$$\pi_{t_{t-1}} = \begin{pmatrix} 0{,}4 & 0{,}3 & 0{,}3 \\ 0{,}2 & 0{,}6 & 0{,}2 \\ 0{,}1 & 0{,}1 & 0{,}8 \end{pmatrix}$$

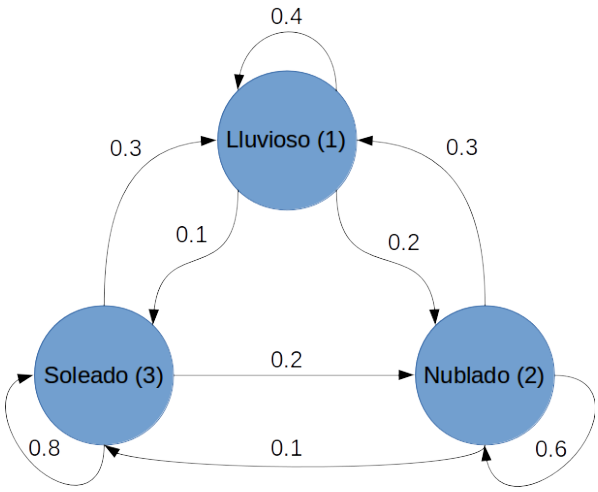

Figura 3.2: Diagrama del modelo de markov del clima.

$$\pi_0 = \{\pi_{0j}\} = P(q_1 = \{S_j\}) \tag{3.4}$$

Por ejemplo, la expresión para la probabilidad de obtener una secuencia de estados $O = \{soleado, soleado, nublado, lluvioso, soleado\}$ está dada por:

$$\begin{aligned} P(O|\pi_{t_{t-1}}, \pi_0) &= P(S_3, S_3, S_2, S_1, S_3|\pi_{t_{t-1}}, \pi_0) \\ &= P(S_3) \cdot P(S_3|S_3) \cdot P(S_2|S_3) \cdot P(S_1|S_2) \cdot P(S_3|S_1) \\ &= \pi_{03} \cdot \pi_{t_{t-1}33} \cdot \pi_{t_{t-1}32} \cdot \pi_{t_{t-1}21} \cdot \pi_{t_{t-1}13} \\ &= 1 \cdot 0{,}8 \cdot 0{,}1 \cdot 0{,}3 \cdot 0{,}3 \end{aligned}$$

3.2. HMM con observaciones discretas

Como ya se mencionó, en los HMM, las secuencias de estados son desconocidas, en su lugar se tienen observaciones (condicionadas por los estados) que tienen información oculta de los estados. En esta sección se describe el caso de HMM donde las observaciones son discretas, es decir que provienenen de un diccionario conocido (y finito) de posibles muestras observables.

3.2.1. Formulación del modelo HMM discreto

Para introducir la idea de los HMM (2), considerar la siguiente situación:

Modelo de Urnas y Pelotas: Considerar que se tiene un sistema con N urnas, donde cada una de estas contiene una gran variedad de pelotas de M posibles colores. Suponer ahora que llega una persona, y de acuerdo a cierto proceso aleatorio, esta escoge una urna inicial y saca una pelota de esta, toma nota del color que obtuvo, devuelve la pelota la urna. La persona selecciona una nueva urna (se puede repetir la urna) y repite el proceso la cantidad de veces que desee. Este proceso genera una secuencia de observaciones correspondientes a los colores que se obtuvieron de las distintas urnas (sin conocer las urnas de origen), que se desean modelar como un HMM.

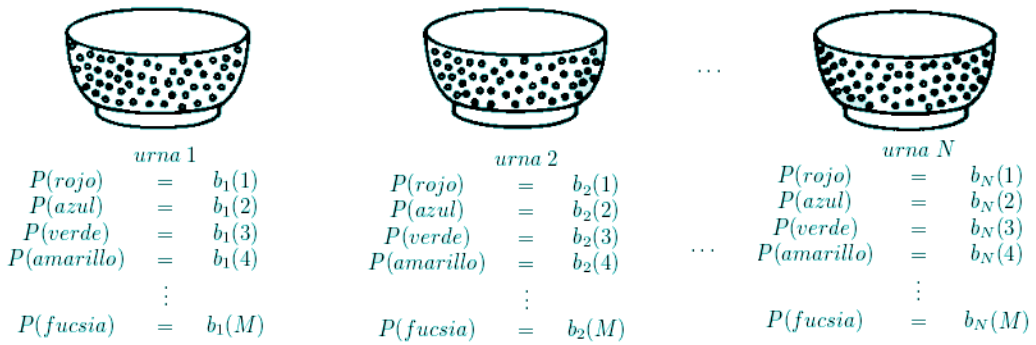

Figura 3.3: Ilustración del Modelo de Urnas y Pelotas, con N urnas, y M posibles colores de pelotas dentro de cada urna.

En esta situación, el HMM está caracterizado por:

1. N, el numero de estados del modelo. Estos estados, en el planteamiento del modelo, son desconocidos, pero tienen una significancia fśica con el proceso observado. Para el caso del modelo de urnas y pelotas, el estado está asociado a la urna de origen de las pelotas de color observadas. Cada uno de los estados se denotan por $S_1, S_2, ..., S_N$, y el estado en un tiempo t arbitrario como q_t.

2. M, el numero de posibles símbolos observables por estado. Los símbolos observables corresponden a la salida física del sistema a modelar. Para la situación planteada en el modelo de ejemplo, los simbolos observables, serían los M posibles colores de pelotas que existen en cada urna. Se denotan cada uno de estos simbolos como $X = \{x_1, x_2, ..., x_M\}$.

3. La probabilidad de distribución de transición de estados $\pi_{t_{t-1}} = \{\pi_{t_{t-1}ij}\}$. Cada π_{0ij} corresponde a la probabilidad de pasar al estado j ($q_{t+1} = S_j$), dado que en el instante actual se encuentra en el estado i ($q_t = S_i$), ecuación 3.5

$$\{\pi_{t_{t-1}ij}\} = P(q_{t+1} = S_j | q_t = S_i),\ 1 \leq i, j \leq N \tag{3.5}$$

Para el caso de las Urnas y Pelotas, esta variables corresponde a la probabilidad de cambiar de la urna i a la j entre observaciones.

4. La districución de probabilidad de los símbolos observados en el estado j, $B = b_j(k)$ (ecuación 3.6)

$$b_j(k) = P(x_k \ en \ t | q_t = S_j),\ \begin{matrix} 1 \leq j \leq N \\ 1 \leq k \leq M \end{matrix} \tag{3.6}$$

Donde en el caso de las urnas, $b_j(k)$ corresponde a la probabilidad de extraer el k-ésimo color, estando en la urna j.

5. La distribución de probabilidad del estado inicial $\pi_0 = \{\pi_{0i}\}$, esta variable modela la probabilidad de encontrarse en un estado j en el instante inicial $t = 1$ (ec. 3.7)

$$\pi_{0j} = P(q_1 = S_j),\ 1 \leq j \leq N \tag{3.7}$$

Para la situación de las urnas, π_{0j} sería la probabilidad de escoger la urna j para extraer la primera pelota.

Los elementos recién mencionados modelan un HMM y genera una secuencia de observaciones $O = O_1, O_2, \ldots, O_T$.

3.2.2. Problemas a resolver con HMM

Cuando se utiliza este modelo con observaciones de series de tiempo (por ejemplo mediciones de EEG), comúnmente se busca resolver tres problemas:

- Evaluación: Calcular la probabilidad de que un modelo ($\lambda = (\pi_{t_{t-1}}, B, \pi_0)$) haya generado una secuencia de observaciones O, es decir calcular la verosimilitud de O, dado el modelo ($P(O|\lambda)$).

- Decodificación: Encontrar la secuencia de estados $Q = q_1, q_2, .., q_t$ más probable que haya generado la secuencia de estados O, en un sentido óptimo.

- Aprendizaje: Calcular el set de parámetros $\lambda = (\pi_{t_{t-1}}, B, \pi_0)$ que definen el modelo HMM, que maximiza la probabilidad de haber generado una secuencia de datos O ($P(O|\lambda)$), visto de otra manera, es encontrar el modelo HMM que representa de mejor manera a la secuncia O.

Problema 1: Evaluación - (Método Forward-Backward)

El cálculo de $P(O|\lambda)$, se apoya principalmente en el Método Forward-Backward, el cual consiste en 2 estapas. En la primera, se calculan, para cada instante t, la probabilidad de terminar en un estado en particular S_i, dada la secuencia de observaciones parcial ($O_{1:t}$), en la segunda etapa se calculan las probabilidades, para cada t, de generar las observaciones restantes $O_{t+1,...,T}$, estando en un estado en particular $q_t = S_i$.

- Paso Forward: En este paso se define la cantidad $\alpha_t(i)$ (ec. 3.8), la probabilidad conjunta de haber observado $O_{1:t}$ y terminar en un estado $q_t = S_i$, dado el modelo λ.

$$\alpha_t(i) = P(O_1, O_2, ..., O_t, q_t = S_i|\lambda) \tag{3.8}$$

  Los distintos $\alpha_t(i)$ se van calculando de manera inductiva según:

  1. Inicialización:
$$a_1(i) = \pi_i b_i(O_1),\ 1 \leq i \leq N \tag{3.9}$$
  2. Inducción:
$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1})\ \begin{matrix} 1 \leq t \leq T-1 \\ 1 \leq j \leq N \end{matrix} \tag{3.10}$$
  3. Término:
$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{3.11}$$

  Si bien, solo basta con el último paso de la parte Forward para calcular $P(O|\lambda)$, el calculo de todas las probabilidades intermedias $\alpha_t(j)$ en conjunto con las que se mostrarán en el paso Backward, serán de gran utilidad para la estimación de parámetros en el aprendizaje del modelo.

- Paso Backward: De manera similar, se define la cantidad $\beta_t(i)$ (ec. 3.12), que deonta la probabilidad de generar las observaciones $O_{t+1:T}$, dado que se está en un estado $q_t = S_i$, para un modelo λ.

$$\beta_t(i) = P(O_{t+1}, O_{t+2}, ..., O_T|q_t = S_i, \lambda) \tag{3.12}$$

1. Inicio:

$$\beta_T(i) = 1, 1 \leq i \leq N \tag{3.13}$$

2. Inducción:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j), \quad \begin{matrix} t = T-1, T-2, ..., 1 \\ 1 \leq i \leq N \end{matrix} \tag{3.14}$$

Problema 2: Decodificación - (Algoritmo de Viterbi)

Para este problema se utiliza el algoritmo de viterbi (1), con el cual se busca la secuencia de estados $Q = q_1, q_2, ..., q_T$, que represente de mejor manera la secuencia de observaciones O (es decir que sea la secuencia de estados más probable).

Se define la cantidad $\delta_t(j)$ (ec. 3.15), que corresponde a la probabilidad de que la secuencia de estados $Q_{1:t-1}$ más probable, termine en el estado $q_t = i$, para las las observaciones $O_{1:t}$, dado un modelo λ.

$$\delta_t(j) = \max_{q_1, q_2, \cdots, q_{t-1}} P[q_1, q_2, q_t = i; O_1, O_2, ..., O_t | \lambda] \tag{3.15}$$

Resolviendo esto de manera inductiva, como se muestra en la ec. 3.19, se puede encontrar la secuencia más probable haciendo un seguimiento de los argumentos que maximizan $\delta_{t+1}(j)$ (j en le ec. 3.19), estos ultimos se guardan enuna variable auxiliar $\psi_t(j)$.

$$\delta_{t+1}(j) = [\max_i \delta_t(i) a_{ij}] b_j(t+1) \tag{3.16}$$

El procedimiento completo para encontrar la secuencia de estados es:

1. Inicialización

$$\delta_1(j) = \pi_j b_j(O_1), \quad 1 \leq j \leq N \tag{3.17}$$

$$\psi_t(j) = 0. \tag{3.18}$$

2. Inducción:

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(t), \quad \begin{matrix} 2 \leq t \leq T \\ 1 \leq j \leq N \end{matrix} \tag{3.19}$$

$$\psi_t(j) = \operatorname*{argmax}_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}], \quad \begin{matrix} 2 \leq t \leq T \\ 1 \leq j \leq N \end{matrix} \tag{3.20}$$

3. Término:

$$P^* = \max_{1 \leq j \leq N} [\delta_T(j)] \tag{3.21}$$

$$q_T^* = \underset{1 \leq j \leq N}{argmax} [\delta_T(j)] \tag{3.22}$$

4. Secuencia de estados:

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \ \ t = T-1, T-2, ..., 1. \tag{3.23}$$

Problema 3: Aprendizaje - Método de Baum-Welch

El tercer problema que a resolver es estimar el set de parámetros $\lambda = (A, B, \pi)$, que mejor represente la secuencia de observaciones generada O de manera que se maximice $P(O|\lambda)$, es decir encontrar el modelo que con mayor probabilidad, haya generado la secuencia de observaciones O. Estos parámetros se calculan de forma iterativa con el Método de Baum-Welch.

Para la reestimación de los parámetros, se definen 2 cantidades nuevas:

- $\xi_t(i, j)$: que representa la probabilidad de pasar del estado i al j en el instante t dada la secuencia de observaciones O y el modelo λ.

$$\xi_k(i, j) \triangleq P(q_t = S_i, q_{t+1} = S_j \,|\, O, \lambda) \tag{3.24}$$

  Esta probabilidad se calcula en términos de las variables forward ($\alpha_t(i)$) y backward ($\beta_t(i)$) como:

$$\begin{aligned} \xi_t(i, j) &= \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{P(O \mid \lambda)} \\ &= \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1^N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)} \end{aligned} \tag{3.25}$$

  En la fracción, el término en el numerador corresponde a la probabilidad de estar en un estado $q_t = S_i$ y pasar al estado $q_{t+1} = S_j$, para las observaciones O y el modelo λ ($P(q_t = S_i, q_{t+1} = S_j, O \mid \lambda)$).

- $\gamma_t(i, j)$: la probabilidad de estar en un estado $q_t = S_i$ dada la secuencia de observaciones O, dado el actual modelo λ.

$$\gamma_k(i) \triangleq P(q_t = S_i \,|\, O, \lambda) \tag{3.26}$$

  Esta probabilidad que calcula sumando $\xi_t(i, j)$ sobre j:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i, j) \tag{3.27}$$

Con estas cantidades definidas, se pueden calcular los valores esperados de las propiedades de transición de la cadena de Markov (ecs. 3.28 y 3.29)

$$\sum_{t=1}^{T-1} \gamma_t(i) = \textit{Número esperado de transiciones desde el estado } i \tag{3.28}$$

$$\sum_{t=1}^{T-1} \xi_t(i,j) = \textit{Número esperado de transiciones desde el estado } i \textit{ al estado } j \tag{3.29}$$

Con estos valores calculados, se calculan los parámetros que definen el HMM, según las ecs. 3.30 a la 3.32.

$$\begin{aligned} \hat{\pi}_i &= \textit{probabilidad esperada de estar en el estado } i \textit{ en el instante } t = 1 \\ &= \gamma_1(i) \end{aligned} \tag{3.30}$$

$$\begin{aligned} \hat{a_{ij}} &= \frac{\textit{Número esperado de transiciones desde el estado } i \textit{ al estado } j}{\textit{Numero esperado de transiciones desde el estado } i} \\ &= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \end{aligned} \tag{3.31}$$

$$\begin{aligned} \hat{b}_i(m) &= \frac{\textit{Número esperado de veces en el estado } i \textit{ y observando el símbolo } x_m}{\textit{Número esperado de veces en el estado } i} \\ &= \frac{\sum_{t=1, O_t = x_m}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)} \end{aligned} \tag{3.32}$$

La ec. 3.32, solo es válida cuando se trata con observaciones discretas.

El método de Baum-Welch, es una implementación del algoritmo EM (Expectation-Maximization) (3), donde el paso E se realiza en las ecs. 3.28 y 3.29 para calcular los valores esperados de las propiedades de transción de la Cadena de Markov, y el paso M se realiza en las ecs. 3.30 a la 3.32, donde se calculan los parámetros que maximizan $P(O,\lambda)$.

3.3. HMM con emisiones Continuas (Caso Gaussiano)

En el caso descrito anteriormente, la colección de observaciones $O = (O_1, O_2, ..., O_T)$ provenían de un diccionario finito de símbolos, llevandolo nuevamente al caso del modelo de Urnas y Pelotas, las observaciones venían dadas por una cantidad finita de posibles colores de las pelotas. En este caso, los símbolos observados, son una serie de tiempo proveniente de distribuciones continuas de probabilidad condicionadas por estados. En este modelo lo unico que cambia con respecto al caso discreto, son las distribuciones de probabilidad, que pasan a

ser continuas, por lo que para modelar el HMM con los datos, solo hay que modificar la ecuación de estimación 3.32 para calcular los parámetros de las distribuciones de probabilidad de las observaciones en vez de directamente calcular las probabilidades discretas de observar cada símbolo.
En esta sección se muestra la formulación y las ecuaciones para modelar el HMM cuando las distribuciones de probabilidad que condicionan las observaciones provienen de una distribución Gaussiana.

3.3.1. Formulación del modelo HMM continuo

En este modelo, como se mencionó, lo que cambia con respecto al caso discreto son las distribuciones de probabilidad de las observaciones para cada estado. Las distribuciones de las observaciones x_t son modeladas como Gaussianas (multivariadas) de media μ_j y covarianza Σ_j, para cada posible estado j (ec. 3.33).

$$b_j(x_t) = N(x_t|\mu_j, \Sigma_j), \;\; 1 \leq j \leq N \tag{3.33}$$

Todos los demás elementos del modelo, la distribución de transiciones de estado A y la distribución inicial de estar en un estado Π, se mantienen, y los problemas típicos a resolver, se resuelven de la misma manera, ya que todos estos están presentados para cualquier tipo de distribucion de observaciones b_j, lo que queda por redefinir son las ecuaciones método de Baum-Welch para estimar los parámetros de las distintas distribuciones Normales.

3.3.2. Estimación del modelo - Método de Baum-Welch

Para un modelo HMM continuo (gaussiano), de N estados, la ecuaciones de reestimación para la medias μ_j y covarianzas Σ_j vienen dadas por las ecs. 3.34 y 3.35.

$$\mu_j = \frac{\sum_{t=1}^{T} \gamma_t(j) x_t}{\sum_{t=1}^{T} \gamma_t(j)}, \;\; 1 \leq j \leq N \tag{3.34}$$

$$\Sigma_j = \frac{\sum_{t=1}^{T} \gamma_t(j)(x_t - \mu_j)(x_t - \mu_j)^T}{\sum_{t=1}^{T} \gamma_t(j)}, \;\; 1 \leq j \leq N \tag{3.35}$$

El método de Baum-Welch en este caso sigue la misma metodología que en el caso discreto, solo que en el paso M, se reemplaza la ec. 3.32 por las ecs. 3.34 y 3.35.

3.4. HMM Autoregresivo multivariado (HMM-MAR)

Otro modelo de observación para el HMM, son el modelo *MAR*, que modela las observaciones como el error (o señal de innovación) de un proceso MAR, este tipo de modelos es usado

comunmente en aplicaciones de señales donde es importante tener la información espectral (que se concentra en los parámetros del modelo *MAR*), como es el caso de señales de voz (2) o EEG (7). El modelo HMM-MAR se puede ver como una extensión a cadenas de markov del modelo *mAR(p)* (eq. 2.8), donde las matrices autoregresoras ($\mathbf{A} = (A_1, A_2, ..., A_p)$) y la varianza Σ de la señal de error W_t, son modulados por la secuencia de estados oculta, dicho en otras palabras, el modelo HMMmAR cambia entre distintos submodelos autoregresivos, cada uno con su propio set de parámetros.

3.4.1. Formulación del Modelo HMM-MAR

Antes de formular el modelo, se redefinen las variables del modelo autoregresivo (ec. 2.8), para adaptarlas a la dinámica de un HMM.

- Coeficientes autoregresores: Las matrices de coeficientes autoregresores $\mathbf{A} = (A_1, A_2, ..., A_p)$ se denotan con un superindice j ($\mathbf{A}^{(i)} = (A_1^{(i)}, A_2^{(i)}, ..., A_p^{(i)})$), para indicar que se trata del modelo autoregresivo asociado al estado j.

- Señal de error: La señal W_t se denota como $W_t(j)$ para indicar que se trata de la señal de error del modelo mAR(p) asociado al estado j.

- Varianza del error: Para cada estado, la varianza de la señal de error asociada a cada estado es Σ_j.

Con estas definiciones previas, se plantea la ecuación del nuevo modelo *mAR(p)*, para una señal multivariada $X_t = (x_{1,t}\ x_{2,t}\ \dots\ x_{k,t})^T$, donde se incluye que los modelos asociados pueden cambiar con los estados de la cadena de markov (ec. 3.36).

$$X_t = \sum_{i=1}^{p} A_i^{(j)} X_{t-i} + W_t(j), \;\; W_t(j) \sim N(0, \Sigma_j) \tag{3.36}$$

Moviendo los términos de la ecuación 3.36 se tiene una expresión para la señal de error W_t (ec. 3.37)

$$W_t(j) = X_t - \hat{X}_t, \;\; W_t(j) \sim N(0, \Sigma_j) \tag{3.37}$$

A la sumatoria $\sum_{i=1}^{p} A_i^{(j)} X_{t-i}$, se le denota por $\hat{X}_t(j)$, que corresponde a la predicción de la observación X_t con los coeficientes autoregresores del estado j

El modelo HMMmAR se modela de manera muy similar al caso Gaussiano, manteniendo todas las propiedades del HMM continuo, solo que en esta situación, la colección de observaciones O, provienen de la señal de error $W_t(j)$, distribuida de forma normal (ec. 3.38.)

$$b_j(W_t) = N(W_t \mid 0, \Sigma_j), \;\; 1 \leq j \leq N \tag{3.38}$$

3.4.2. Estimación de Parámetros

Para el HMMmAR, el cálculo de las probabilidades iniciales Π, las de transición de estados A y la covarianzas Σ_j de las distribuciónes $b_j(W_t)$ (*no es necesario calcular la media de $b_j(W_t)$, ya que está definida como cero en la ec. 3.36*), es igual que en el caso HMM continuo. Como la señal de ruido W_t está descrita en términos de las osbervaciones y los coeficientes, hay que tener las ecuaciones de reestimación de estos, para introducirlos al paso M del Método de Baum-Welch, las que se describen a continuación.

Definiendo la matriz C_i (ec. 3.39), una matriz de pesos que contiene las probabilidades parciales de estar en un estado j, $\gamma_t(j)$, y la matriz $M_t = (X_{t-1}\; X_{t-2}\; \ldots\; X_{t-p})$, el calculo de los autoregresores para cada estado j viene dado por las ecuaciones 3.40 a la 3.42.

$$C_i = diag(\sqrt{\gamma_1(j)}, \ldots, \sqrt{\gamma_T(j)}), \;\; 1 \leq j \leq N \tag{3.39}$$

$$A^{(j)} = (\tilde{M_j}^T \tilde{M_j})^{-1} \tilde{M_j} \tilde{X_j}, \;\; 1 \leq j \leq N \tag{3.40}$$

$$M = -\begin{pmatrix} M_0 \\ \vdots \\ M_t \\ \vdots \\ M_T \end{pmatrix} \quad \tilde{M_i} = C_j M \tag{3.41}$$

$$X = (X_0 \ldots X_T)^T \;\; \tilde{X_j} = C_j X \tag{3.42}$$

Es importante notar que esta ecuación tiene la misma forma que la solución del mAR(p) (ec. 2.6), pero en este caso las matrices de información (M y X), están ponderadas por las probabilidades $\gamma_t(j)$ a través de las matrices C_i.

3.5. Escalamiento

3.5.1. Escalamiento en el Algoritmo Forward-Backward

Cuando se tienen secuencias de observaciones muy grandes, los cálculos de las probabilidades a lo largo del Algoritmo Forward-Backweard, se escapan del rango de presición de los cálculos computacionales. Para entender entender mejor esta limitación, considerar el calculo

de la variable $\alpha_t(i)$ durante el paso Forward (ecs. 3.8 a la 3.11), el calculo de $\alpha_t(i)$ considera la suma de varios términos de la forma

$$\prod_{s=1}^{t-1} a_{q_s q_{s+1}} \prod_{s=1}^{t} b_{q_s}(O_s), \;\; q_t = S_j$$

Acá, los términos $a_{q_s q_{s+1}}$ y $b_{q_s}(O_s)$, son iguales o menores a 1, por lo que resultado de los próductos del término recién mostrado, se va acercando exponencialmente a cero a medida que t aumenta (con t>100 ya es dificil que la precisión de una maquina ya pueda seguir calculando estos valores). Como para $\beta_j(i)$ se tiene una situación similar, es necesario tener una forma de poder escalar estos términos (y los términos restantes que dependen de $\alpha_t(j)$ y $\beta_t(j)$) a los largo de sus cálculos, que se describe a continuación:

- Paso Forward Escalado:
  El calculo de $\hat{\alpha_t(j)}$ (la versión escalada de $\alpha_t(j)$) se hace simplemente escalando los distintos $\alpha_t(j)$, por su suma sobre los estados j (ec. 3.43).

$$\hat{\alpha}_t(i) = \alpha_t(i) c_t, \;\; c_t = \frac{1}{\sum_{i=1}^{N} \alpha_t(i)} \tag{3.43}$$

  Con $\hat{\alpha}_t(i)$, se calcula $\alpha_i(t)$ como en la ec. 3.44

$$\alpha_t(i) = \sum_{j=1}^{N} \hat{\alpha}_t(j) a_{ij} b_j(O_t) \tag{3.44}$$

- Paso Backward Escalado:
  Como en el paso Forward, el cálculo de $\beta_t(i)$ escalado ($\hat{\beta}_t(i)$), se hace multiplica por un factor de la misma forma que el de $\alpha_t(i)$ (el inverso de la suma sobre j), como se ve en la ec. 3.45.

$$\hat{\beta_t(i)} = \beta_t(i) c_t, \;\; c_t = \frac{1}{\sum_{i=1}^{N} \beta_t(i)} \tag{3.45}$$

  Se calcula $\beta_t(i)$ como en la ec. 3.46

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \hat{\beta_{t+1}(j)} \tag{3.46}$$

- El cálculo del resto de las variables dependientes de $\beta_t(j)$ y $\alpha_t(j)$, se hacen normalmente, como en sus ecuaciones originales, simplemente hay que reemplazar $\beta_t(j)$ y $\alpha_t(j)$ por $\hat{\beta_t(j)}$ y $\hat{\alpha_t(j)}$ donde corresponda.

3.5.2. Escalamiento del Algoritmo de Viterbi

En el algoritmo de Viterbi, también se trata con cantidades que se pueden escapar del rango de presición de una maquina (ec. 3.19), también hay que cambiar la escala de los datos, lo que se hace definiendo la nueva variable $\phi_t(j)$ (ec. 3.47) que es una versión logarítmica de la variable, $\delta_t(j)$. Con este cambio de escala los cálculos ya son abordables, y el algoritmo de viterbi se plantea en su versión logarítmica como se muestra en las ecs. 3.48 a la 3.54.

$$\phi_t(j) = \max_{q_1,q_2,\ldots,q_{t-1}} log(P[q_1,q_2,q_t = i; O_1,O_2,...,O_t|\lambda]) \tag{3.47}$$

1. Inicialización

$$\phi_1(j) = log(\pi_j) + log(b_j(O_1)), \quad 1 \leq j \leq N \tag{3.48}$$
$$\psi_t(j) = 0. \tag{3.49}$$

2. Inducción:

$$\phi_t(j) = \max_{1\leq i\leq N}[\phi_{t-1}(i) + log(a_{ij})] + log(b_j(t)), \quad \begin{matrix} 2 \leq t \leq T \\ 1 \leq j \leq N \end{matrix} \tag{3.50}$$

$$\psi_t(j) = \underset{1\leq i\leq N}{argmax}[\phi_{t-1}(i) + log(a_{ij})], \quad \begin{matrix} 2 \leq t \leq T \\ 1 \leq j \leq N \end{matrix} \tag{3.51}$$

3. Término:

$$P^* = \max_{1\leq j\leq N}[\phi_T(j)] \tag{3.52}$$
$$q_T^* = \underset{1\leq j\leq N}{argmax}[\phi_T(j)] \tag{3.53}$$

4. Secuencia de estados:

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \quad t = T-1, T-2, ..., 1. \tag{3.54}$$

3.6. Inicialización de parámetros

Las ecuaciones presentadas en este capítulo generalmente convergen a óptimos locales, lo que está condicionado principalmente por como se inicialicen los parámetros. Normalmente en este tipo de esquemas de optimización iterativos (como lo es el algoritmo EM(3)).
Una opción para evadir óptimos locales es iniciar los parámetros aleatoriamente y ejecutar unas cuantas iteraciones (bastan unas $5 \sim 6$) del algoritmo para optimizar el HMM, para luego escoger el set de parámetros iniciales que resulten en la mayor verosimilitud de los datos

dado el modelo (ec. 3.11).

Otra forma es iniciar los parámetros con alguna otra rutina de optimización que sea menos demandante computacionalmente, como lo son los algoritmos de clustering[1] como el Modelo de Mezcla de Gaussianas GMM (18) o el algoritmo K-means (17), lo que acelera el proceso iterativo de inicialización.

Para los modelos HMM de observaciones continuas revisados en las secciones anteriores la forma de inicializar es como sigue:

- HMM Gaussiano:
  Para los parámetros de media de las distribuciones asociadas a cada distribución se puede usar tanto el algoritmo K-means o GMM. El resto de los parámetros se escoge aleatoriamente o según el conocimiento previo que se tenga de los estados.

- HMM-MAR:
  Para encontrar los coeficientes autoregresores, se pasa por una etapa de filtro de Kalman (KF)[2] como se muestra en el apéndice B, y luego se pasan los sets de parámetros obtenidos por el modelo GMM (o el algoritmo K-means), para utilizar las medias obtenidas como inicialización de los coeficientes MAR. Es importante considerar que no todos los coeficientes AR servirán para la etapa de clustering, por lo que se hace una selección de estos, donde se escogen los que tengan una buena evidencia de predicción (ecs. B.9 y B.21). En la figura 3.4 se puede ver un ejemplo de como funciona la inicialización para un modelo HMM-AR. El resto de los parámetros se escogen aleatoriamente como el caso Gaussiano.

Es importante considerar que estas pautas de inicialización tampoco aseguran que se llegue al óptimo cuando se estima el modelo, por lo que es importante ir probando que formas de inicializar los parámetros se puede ajustar mejor a los datos que uno tenga.

[1] algoritmos que toman subconjuntos de un set de datos y los agrupan en clases (o clusters), en las que todos los datos de un mismo cluster comparten características similares entre si.

[2] También, para reducir cantidad de parámetros a calcular, se puede calcular un modelo AR por ventanas, dando resultados similares

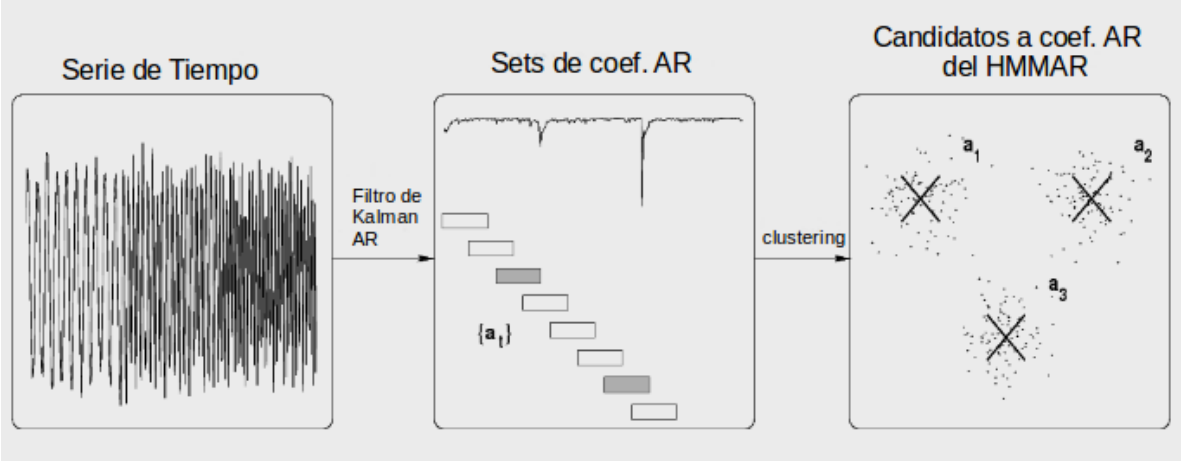

Figura 3.4: Inicialización de los coeficientes del modelo HMM-AR. Pasa el KF por la serie de tiempo original para encontrar los coeficientes AR para cada instante. Si la evidencia (gráfica superior en el cuadro de en medio) de una predición es muy baja, se remueven dichos coeficientes (cuadros coloreados en el la figura de en medio). El clustering (tercer cuadro) se encarga de escoger los set de parámetros AR de cada clase que pasarán a iniciar el modelo HMM-AR

Capítulo 4

Aprendizaje Variacional para HMM

En la sección anterior se ha descrito el esquema de máximo verosimil (algorítmo EM) para la estimación del modelo HMM, en el cual se tienen ecuaciones que llevan a un único valor de los parámetros del HMM que máximizan la verosimilitud de los datos dado el modelo. En este capitulo se muestra el esquema Bayesiano Variacional para optimizar el modelo HMM, donde, a diferencia de los estimadores de máximo verosimil, el estimador Bayesiano presentado en este capítulo, resulta en una distribución de probabilidad para los parámetros y con esto, también una medida de confiabilidad de los datos.
Tener información a priori de la probabilidad de los datos y parámetros, además acota el espacio donde se busca la solución óptima, resultando en una más rápida convergencia y una mayor robustez a caer en máximos locales (7).

Normalmente la dificultad de para optimizar modelos complejos (como los HMM), es que las expresiones de las probabilidades a posteriori (de los parámetros dados los datos) suelen ser matemáticamente muy complejos, que requieren resolver integrales incalculables, lo cual se evita a través de la optimización de una forma aproximada (y facil de calcular) de las distribuciones originales. En este capítulo se muestra la formulación del HMM Variacional con sus aproximaciones de las distribuciones de probabilidad de los parámetros, las ecuaciones para la optimización del modelo y un método de selección de modelos para escoger el numero adecuado de parámetros o estados para describir una secuencia de datos.

4.1. Principios del aprendizaje variacional

En el cálculo variacional, como en todo esquema de optimización, se busca minimizar una función de costo, en este caso, la divergencia de Kullback-Leibler (KL) (5). La divergencia KL se puede entender como una medida de distancia entre dos distribuciones de probabilidad, por ejemplo, para dos distribuciones $Q(H)$ y $P(H,V)$ (se profundizará sobre la definición de Q y P más adelante), la divergencia KL se define por la integral (ec. 4.1):

$$\mathscr{F} = D(Q(H)||P(H,V)) = \int Q(H) log \frac{Q(H)}{P(H|V)} dS + logP(V) \tag{4.1}$$

Las distribuciones $Q(H)$ y $P(H|V)$ se definen sobre todas las variables escondidas (parámetros y estados desconocidos) H, condicionados sobre los datos V. El objetivo en este esquema es aproximar la ditribución a posteriori $P(H|V)$ que parametriza el modelo real, cuya solución se puede aproximar a través de la optimización de la divergencia KL con respecto a la distribución aproximada a priori $Q(H)$, de forma iterativa.

La distribución $Q(H)$ más sencilla para aproximar $P(H|V)$, sería una que asuma todas las variables $H = \{H_1, ..., H_N\}$ como independientes, lo que resulta en una distribución $Q(H)$ factorizada como (ec. 4.2):

$$Q(H) = \prod_{i=1}^{N} Q(H_i), \quad \int Q(H_i) dH_i = 1 \tag{4.2}$$

Esta aproximación se le conoce como aproximación *mean-field* (campo medio). Con esta aproximación, y manteniendo las distintas distribuciones $Q(H_i)$ dentro de la familia exponencial, las soluciones de $Q(H_i)$ que maximizan la divergencia KL tienen la forma de la ec. 4.3(6)

$$Q(H_i) = \frac{1}{Z} exp \int Q(\bar{H}_i) logP(H_i|\bar{H}_i) d\bar{H}_i \tag{4.3}$$

donde cada $\bar{H}_i = H \setminus H_i$ es el set de todas las variables H excluyendo H_i y Z es simplemente una constante de normalización.

4.2. Aprendizaje del modelo HMM

Considerar primero una secuencia de T variables $S = S_1, .., S_t, ..., S_T$ aleatorias (cada S_t puede tomar M valores discretos $S_t = \{s_1, ..., s_M\}$) que representan la secuencia escondida de estados de una cadena de markov, donde el valor observado por S_t está condicionado por S_{t-1} y está se obtiene con una probabilidad $P(S_t|S_{t-1})$, cada S_t puede tomar M valores discretos $S_t = \{s_1, ..., s_M\}$. La secuencia de observaciones son representadas en el vector $X = \{X_1, ..., X_t, ..., X_T\}$, cada una de estas observaciones X_t están condicionadas por el estado S_t, observandose cada una con una probabilidad $P(X_t|S_t)$, la cual está parametrizada por un set de variables θ_{obs}.

La distribución de probabilidad conjunta de la secuencia de estados S y las observaciones X queda dada por (ec 4.4)

$$P(S,X) = P(S_0)\prod_{t=1}^{T} P(S_t|S_{t-1})P(X_t|S_t) \tag{4.4}$$

Las probabilidades de transición $P(S_t|S_{t-1})$ están representadas en la matriz $A \in \mathbb{R}^{M\times M}$ y la probabilidad inicial del estado S_0 está parametrizado por $\Pi \in \mathbb{R}^{1\times M}$.

Considerando los parámetros $\theta = \{A, \Pi, \theta_{obs}\}$ independientes entre si, se puede escribir la distribución a posteriori del modelo como

$$\begin{aligned} P(S,X,\theta) &= P(S,X|\theta)P(\theta) && (4.5)\\ &\triangleq P(S,X|A,\Pi,\theta_{obs})P(A,\Pi,\theta_{obs}) && (4.6)\\ &= P(S_0|\Pi)\prod_{t=1}^{T} P(S_t|S_{t-1},\Pi)P(X_t|S_t,\theta_{obs})P(A)P(\Pi)P(\theta_{obs}) && (4.7) \end{aligned}$$

Agrupando todas variables asociadas a los estados ocultos en la distribución $Q(S)$, la aproximación de campo medio para el cálculo de las variables que gobiernan el modelo queda dada por

$$Q(H) = Q(S)Q(A)Q(\Pi)Q(\theta_{obs}) \tag{4.8}$$

Bajo los supuestos de la cadena de markov, se puede representar $Q(S)$ de forma factorizada como

$$Q(S) = Q(S_0)\prod_{t=1}^{T} Q(S_t|S_{t-1}) \tag{4.9}$$

Con las consideraciones anteriores, solo basta con minimizar la nueva divergencia KL (ec 4.10) con respecto a cada una de las distribuciones $Q(S)$, $Q(A)$, $Q(\Pi)$, $Q(\theta_{obs})$ para encontrar las todas las variables del modelo.

$$\mathscr{F} = \int\cdots\int Q(S)Q(A)Q(\Pi)Q(\theta_{obs}) log \frac{Q(S)Q(A)Q(\Pi)Q(\theta_{obs})}{P(S,X,A,\Pi,\theta_{obs})} \tag{4.10}$$

4.2.1. Aprendizaje de la secuencia de estados

Se comienza por optimizar la divergencia KL (ec. 4.1) con respecto a la distribución $Q(S)$ sobre las variables escondidas asociadas a los estados S. Usando la aproximación de campo medio (ec. 4.2) se puede obtener la distribución óptima $Q(S)$ se obtiene con la ec. 4.3, reemplazando los parámetros θ por $\tilde{H}_i$, obteniendose

$$Q(S) = \frac{1}{Z} exp \int Q(\theta) log P(S, X, \theta) d\theta \tag{4.11}$$

Con la expresión $Q(S)$ se puede obtener la divergencia KL (ec 4.10), la cual, al minizarla con respecto a las distribuciones marignales $Q(S_t)$ y las conjuntas $Q(S_t, S_{t-1})$ de los estados ocultos, resulta en las recursiones del algoritmo de Baum-Welch(7) (se resumen en las ecs. 4.12 a la 4.14)

$$Q(S_t, S_{t-1}) = \frac{1}{z_t} P(S_t|S_{t-1}) \beta(t) \alpha(t-1) \tag{4.12}$$

$$\beta(t) = \frac{1}{z_t} \sum_{S_{t+1}} P(S_t|S_{t-1}) \beta(t+1), \;\; \beta(1) = \frac{1}{z_1} \tag{4.13}$$

$$\alpha(t) = z_t \sum_{S_{t-1}} P(S_t|S_{t-1}) \alpha(t-1), \;\; \alpha(1) = z_1 P(S_0) \tag{4.14}$$

Usando la misma notación del capítulo anterior, se denota la probabilidad marginal de que S_t tome algún valor $m = 1, 2, ..., M$ como $\gamma_t(m)$ (ec. 4.15) y la probabilidad conjunta de que S_t tome algún valor n y S_t haya tomado algún valor m como $\xi_t(m,n)$ (ec. 4.16).

$$\gamma_t(m) = Q(S_t = m|X) \tag{4.15}$$

$$\xi_t(m,n) = Q(S_t = n, S_{t-1} = m|X) \tag{4.16}$$

Finalmente la reconstrucción de la secuencia de estados se hace con el algoritmo de Viterbi de igual manera que se describe en el capitulo anterior en las ecuaciones 3.17 a la 3.23.

4.2.2. Aprendizaje de los parámetros del HMM

Para encontrar los encontrar los parámetros del HMM asociados a las probabilidades de transición π_0 y $\pi_{t_{t-1}}$, primero se definen las distribuciones a priori $P(\theta)$ para estos parámetros, las cuales escogen Dirichlet (de la familia de distribuciones conjugadas) M-dimensional para π_0 (ec. 4.17) y Dirichlet $M \times M$-dimensional para $\pi_{t_{t-1}}$ (ec. 4.18).

$$P(\pi_0) \triangleq Dir(\pi_0) = \frac{\Gamma(\sum_l \kappa_l)}{\prod_l \Gamma(\kappa_l)} \prod_{m=1}^{M} \pi_{0_m}^{\kappa_m - 1} \tag{4.17}$$

$$P(\pi_{t_{t-1}}) \triangleq Dir(\pi_{t_{t-1}}) = \prod_{m=1}^{M} \frac{\Gamma(\sum_l \lambda_{ml})}{\prod_l \Gamma(\lambda_{ml})} \prod_{m=1}^{M} \pi_{0_m}^{\lambda_m - 1} \tag{4.18}$$

Definiendo estas distribuciones, al minimizar la divergencia KL con respecto a las distribuciones aproximadas a posteriori $Q(\pi_0)$ y $Q(\pi_{t_{t-1}})$ resultan tambien en distribuciones Dirichlet de parámetros $\tilde{\kappa}_m$ y $\tilde{\lambda}_{m_n}$ dados por

$$\tilde{\kappa}_m = \gamma_{t=0}(m) + \kappa_m \tag{4.19}$$

$$\tilde{\lambda}_{m_n} = \sum_t \xi_t(m,n) + \lambda_{m_n} \tag{4.20}$$

donde inicialmente se fijan los hiperparámetros κ y λ como valores poco mayores que 1, lo que reflejaría que se tiene poco conocimiento de las distribuciones de π_0 y $\pi_{t_{t-1}}$ a priori (7)

4.2.3. Modelo de observación del HMM

Para completar el HMM falta especificar la distribución de probabilidad de las observaciones condicionadas por el estado $P(X_t|S_t)$, para las cuales también habrá que definir las distribuciones de probabilidad de los parámetros que las modelan. En el capitulo anterior se describieron los modelos de observacion del modelo HMM y la optimización de sus parámetros para emisiones gaussianas multivariadas y provenientes de un modelo mAR. En esta sección se hace la extensión de estos modelos para el esquema variacional.

Observaciones Gaussianas Multivariadas

Para observaciones provenientes de una distribuciones Gaussianas K-dimensionales, condicionadas por el estado S_t cuya probabilidad está dada por

$$P(X_t|S_t = m) = O(X_t|\mu_m, C_m) \tag{4.21}$$

donde $\mu = \{\mu_1, ..., \mu_M\}$ y $C = \{C_1, ..., C_M\}$ son los vectores de la media y las matrices de presición[1] para cada estado $m = 1, ..., M$. Para los vectores μ_m se definen distribuciones normales K-dimensionales (de media μ_{m_0} y precisión C_{m_0}

$$P(\mu_m) = \frac{1}{Z_{\mu_m}} exp\{-\frac{1}{2}(\mu_m - \mu_{m_0})^T C_{m_0}(\mu_m - \mu_{m_0})\}, \;\; m = 1, ..., M \tag{4.22}$$

Para las matrices de presición, se definen distribuciones Wishart K-dimensionales de parámetros de forma y escalamiento $\alpha = \{\alpha_1, ..., \alpha_M\}$ $B = \{B_1, ..., B_M\}$ respectivamente

$$P(C_m) = \frac{1}{Z_{C_m}} |C_m|^{\alpha_m - \frac{K+1}{2}} exp\{-tr(B_m C_m)\} \tag{4.23}$$

[1] La matriz de presición corresponde a la matriz reciproca a la matriz de varianza, es decir que si la varianza es Σ, la matriz de presición asociada corresponde a $C = \Sigma^{-1}$

siendo Z_{C_m} y Z_{μ_m} factores de escalamiento.

Reemplazando estas expresiones en la ecuación de energía libre (ec. 4.10) y minimizando con respecto a las densidades a posteriori $Q(\mu_m)$ y $Q(C_m)$, se obtiene una distribución normal $Q(\mu_m) \sim \mathcal{N}(\tilde{\mu}_{m_0}, \tilde{C}_{m_0})$ para las medias μ_m y una distribución Wishart() $Q(C_m) \sim \mathcal{W}(\tilde{\alpha}_m, \tilde{B}_m)$ para las matrices de presición C_m (los parámetros optimizados, se describen en las ecuaciones 4.24 a la 4.27).

$$\tilde{\mu}_{m_0} = (\bar{\gamma}_t \tilde{\alpha}_m \tilde{B}_m^{-1} + C_{m_0})^{-1} (\tilde{\alpha}_m \tilde{B}_m^{-1} \bar{x}_m + C_{m_0} \mu_{m_0}) \tag{4.24}$$

$$\tilde{C}_{m_0} = (\bar{\gamma} \tilde{\alpha_m} \tilde{B}_m^{-1} + C_{m_0}) \tag{4.25}$$

habiendo definido previamente $\bar{x}_m = \sum_{t=1}^{T} \gamma_t(m) x_t$ y $\bar{\gamma}_m = \sum_{t=1}^{T} \gamma_t(m)$.

$$\tilde{\alpha}_m = \frac{1}{2} \bar{\gamma}_m + \alpha_m \tag{4.26}$$

$$\tilde{B}_m = \frac{1}{2} \sum_{t}^{T} \gamma_t(m) (x_t - \tilde{\mu_{m_0}})^T (x_t - \tilde{\mu_{m_0}}) + \frac{1}{2} \bar{\gamma}_m \tilde{C}_{m_0}^{-1} + B_m \tag{4.27}$$

Observaciones dadas por un modelo mAR

Considerar primero un modelo multivariado lineal condicionado por estados, de observaciones $Y_t \in \mathbb{R}^d$, aproximadas por el producto $H_m X_t$ (una matriz de información X_t ponderada por una matriz de coeficientes regresores H_m) con un error gaussiano, como se ve en

$$P(Y_t - W^{(m)} X_t | S_t = m) \sim \mathcal{N}_d(0, \Sigma_m) \tag{4.28}$$

Esta distribución corresponde a cualquier proceso lineal d-variado con error gaussiano, para que la ecuación 4.28 represente un HMM-mAR se Define X_t como la concatenación de p observaciones anteriores de Y_t, $X_t = (Y_{t-1}^T, ..., Y_{t-p})$ y a $W^{(m)}$ como la matriz $W^{(m)} \in \mathbb{R}^{d \times dp}$ resultante de concatenar (en fila) de las matrices autoregresoras $W_i^{(m)} \in \mathbb{R}^{d \times d}$ (como se define en la sección 3.4.1).

La matriz de coeficientes $W^{(m)}$ se modela como una distribución a priori normal $d \times dp$-matriz variada(12) $P(W^{(m)}) \sim \mathcal{N}_{d,dp}(\Omega_m, \Sigma_m, \Phi_m)$ de media Ω_m y matrices de presición Σ_m y Φ_m. La distribución a priori de la presición del error Σ_m y la presición Φ_m se modelan como distribuciones Wishart $P(\Sigma_m) \sim \mathcal{W}_d(\alpha_{\Sigma_m}, B_{\Sigma_m})$ y $P(\Phi_m) \sim \mathcal{W}_{dp}(\alpha_{\Phi_m}, B_{\Phi_m})$ con coeficientes de forma α_{Σ_m}, α_{Φ_m} y escala B_{Σ_m}, B_{Φ_m}.

Minimizando la energía libre (ec. 4.10) con respecto a la distribución aproximada a posteriori $Q(W^{(m)})$ se obtiene una distribución normal con media $\tilde{\Omega}_m^T$ y presiciones $\tilde{\Sigma}_m$ y $\tilde{\Phi}_m$, cuyo calculo se resume en las ecs. 4.29 a la 4.31.

$$\tilde{\Omega_m}^T = \Phi_m^{-1}(\sum_n \gamma_n(m) X_n Y_n^T + \tilde{\alpha}_{\Phi_m} \tilde{B}_{\Phi_m}^{-1} \Omega^T) \tag{4.29}$$

$$\tilde{\alpha}_{\Sigma_m} \tilde{B}_{\Sigma_m}^{-1} \tag{4.30}$$

$$\tilde{\Phi}_m = \sum_n \gamma_t(m) X_n X_n^T + \tilde{\alpha}_{\Phi_m} \tilde{B}_{\Phi_m}^{-1} \tag{4.31}$$

De forma similar se obtienen distribuciones Wishart para las distribuciones posteriori $Q(\Sigma_m)$ y $Q(\Phi_m)$ de parámetros de forma/escala $\tilde{\alpha}_{\Sigma_m}/\tilde{B}_{\Sigma_m}$ para $Q(\Sigma_m) \sim \mathscr{W}(\tilde{\alpha}_{\Sigma_m}, \tilde{B}_{\Sigma_m})$ y $\tilde{\alpha_{\Phi_m}}/\tilde{B}_{\Phi_m}$ para $Q(\Phi_m) \sim \mathscr{W}(\tilde{\alpha}_{\Phi_m}, \tilde{B}_{\Phi_m})$, resumiendo sus expresiones en las ecs. 4.32 a la 4.35.

$$\tilde{\alpha}_{\Sigma_m} = \frac{1}{2}(\sum_n \gamma_n(m) + dp + 2\alpha_{\sigma_m}) \tag{4.32}$$

$$\begin{aligned} \tilde{B}_{\Sigma_m} = & \ \tfrac{1}{2}\textstyle\sum_n \gamma_n(m)[(Y_n - \tilde{\Omega}_m X_n)(Y_n - \tilde{\Omega}_m X_n)^T + tr(X_n X_n^T \Phi_m^{-1})\tilde{\Sigma_m}^{-1}] \\ & + \tfrac{1}{2}(\tilde{\Omega}_m - \Omega_m)\tilde{\alpha}_{\Phi_m}\tilde{B}_{\Phi_m}^{-1}(\tilde{\Omega}_m - \Omega_m)^T + \tfrac{1}{2}tr(\tilde{\alpha}_{\Phi_m}\tilde{B}_{\Phi_m}^{-1}\Phi_m^{-1})\tilde{\Sigma}_m^{-1} + B_{\Sigma_m} \end{aligned} \tag{4.33}$$

$$\tilde{\alpha}_{\Phi_m} = \frac{d}{2} + \alpha_{\Phi_m} \tag{4.34}$$

$$\tilde{B}_{\Phi_m} = \frac{1}{2}(\tilde{\Omega}_m - \Omega_m)^T \tilde{\alpha}_{\Sigma_m}\tilde{B}_{\Sigma_m}^{-1}(\tilde{\Omega}_m - \Omega_m) + \frac{1}{2}tr(\tilde{\alpha}_{\Sigma_m}\tilde{B}_{\Sigma_m}^{-1}\tilde{\Sigma}_m^{-1})\tilde{\Phi}_m^{-1} + B_{\Phi_m} \tag{4.35}$$

4.2.4. Estimación

Ya teniendo todas las ecuaciones para las variables del modelo, la estimación se hace iterativamente en dos pasos:

- Calcular las estadísticas asociadas a la secuencia oculta de estados (las probabilidades γ_t y y ξ_t)
- Calcular las distribuciones aproximadas a posteriori $Q(H)$ (las estadísticas del HMM y su modelo de observación asociado) que optimizan la divergencia KL

Estos calculos se repiten cuantas veces se desee, o se cumpla con algún criterio de convergencia. La convergencia del modelo se puede discriminar observando como cambia la divergencia KL 4.10 en cada iteración del algoritmo de estimación, cuando el cambio de la divergencia KL ya deja de ser significativo, se considerá que los parámetros ya convergieron a un óptimo local.

4.3. Selección de modelos

Las rutinas de optimización del modelo HMM-MAR presentadas están hechas para una estructura fija de numero estados (N) y orden de modelo MAR (p). El aprendizaje variacional, añadiendo otro supuesto, que los modelos de diferentes estructuras (valores de N u p) independiente entre si.

Con este supuesto, si se considera una colección de modelos $a \in A$, uno puede escribir la probabilidad a posteriori de todos los modelos pertenecientes a A como el producto de la probabilidad a posteriori de cada uno de los modelos individuales a, como se muestra en la ecuación 4.36, donde cada uno de los factores $P(S|a)P(\theta|a)P(a)$ corresponde a la probabilidad de cada submodelo.

$$P(S,\theta,A) = \prod_{a \in A} P(S|a)P(\theta|a)P(a) \tag{4.36}$$

Lógicamente este supuesto también es válido para la probabilidad aproximada a posteriori para un modelo a, $Q(a) = Q(S|a)Q(\theta|a)$. Inspeccionando la dievergencia KL, para las distribuciones $Q(a)$, se ve que difieren en un solo factor $\mathscr{F}_a$, dado por

$$\mathscr{F}_a = \int\int Q(S|a)Q(\theta|a) log \frac{Q(S|a)Q(\theta|a)}{P(S,V,\theta,a)} dS d\theta \tag{4.37}$$

Este factor corresponde a la divergencia KL para un solo modelo a. Considerando que la probabilidad a posteriori de un modelo a se puede calcular como $Q(a) \propto exp - \mathscr{F}_a$ ((7)), se puede considerar que el modelo a más adecuado para representar los datos, será el modelo más probable, es decir el que tenga el menor valor de $\mathscr{F}_a$.

Capítulo 5

Simulaciones

En este capitulo se comparan los esquemas de aprendizaje vistos en los capitulos 3 y 4, probandolos con una misma señal sintética para observar las ventajas ya mencionadas que tiene el esquema variacional con respecto al EM. Además se muestra el desempeño del modelo HMM-MAR en su versión variacional para hacer segmentación de otras señales sintéticas que figuran ser señales neuronales medidas con EEG, que son generadas a partir de distintos modelos HMM-MAR (con distintos ordenes p y cantidad de estados K), y también para mostrar como el esquema de aprendizaje variacional permite hacer una correcta selección de modelos.

5.1. Generación de Señales Sintéticas

Para probar el algoritmo de estimación del modelo HMM-MAR (incluida la selección de orden), fabrica una serie de tiempo proviniente de un sistema HMM-MAR, cuya secuencia de estados, numero de estados (K) y orden de modelo MAR (p) sea conocida. Estas series de tiempo se diseñan como sigue.

En las simulaciones de procesos HMM-MAR, se diseña la matriz de transición de $\pi_{t_{t-1}}$ para que el modelo HMM tenga alta permanencia en un solo estado (valores altos concentrados en la diagonal de $\pi_{t_{t-1}}$), las probabilidades π_0 se diseñan arbitrariamente (recordando que la suma de las probabilidades de pasar a un estado a los demás, no debe superar 1). Las matrices $W_i^{(m)}$ de los modelos MAR asocidados a los estados del HMM se diseñan de forma que localmente en cada estado, el sistema sea estable, como se indica en el anexo A. La serie de tiempo de observaciones se genera resolviendo recursivamente la expresión de la ecuación 3.36, usando un ruido gaussiano distribuido *i.i.d* como señal de error E_t.

En el diagrama de la figura 5.1 se muestra el esquema de simulación, que se muestra como la salida corresponde a una a un señal de error que pasa por un sistema MAR (ambos

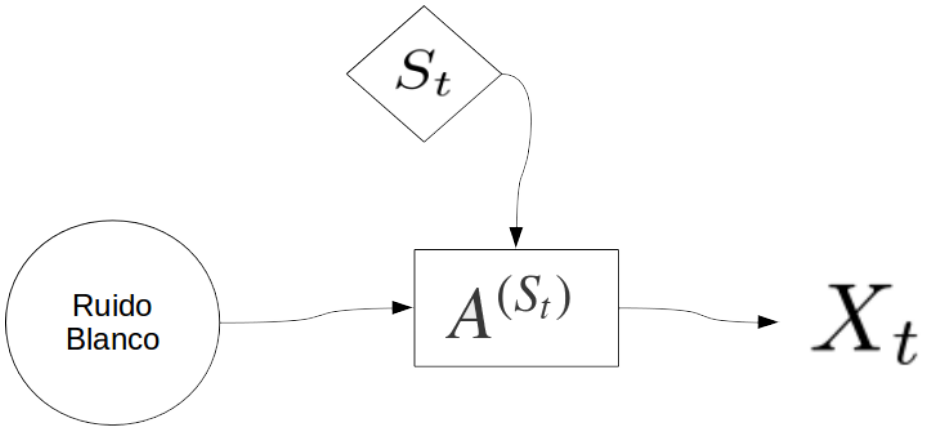

Figura 5.1: Diagrama de la simulación del proceso HMM-MAR, el cuadro del medio representa el modelo MAR que va cambiando condicionado por los N posibles valores (estados) que puede tomar S_t

dependientes del estado)).

El diseño de las matrices $W_i^{(m)}$ está implementado en la función *generate_stable_MAR.m* (apéndice C.2), y la función *simhmmmar*(13) se encarga de generar la cadena de markov y simular el proceso completo.

5.2. Comparación de algoritmo EM y aprendizaje Variacional

A partir de una señal proviniente de un modelo HMM continuo gaussiano, se hace la estimación del modelo asociado con los 2 algoritmos.

En el código C.3 se generan muestras de un proceso HMM gaussiano (se escoge este por ser un modelo de rápido calculo en comparación con HMM-MAR) con *simhmmmar.m*; se hace la estimación para el algoritmo EM y la reconstruccion de la secuencia con el algoritmo de Viterbi usando las funciones *hmmFitEm.m* y *hmmMap* implementadas en el *toolkit* (14); En la parte de aprendizaje variacional, tanto el ajuste del modelo como la reconstrucción con el algoritmo de viterbi se hace con la función *hmmmar.m* de (13).

En ambos casos, usando inicializaciones aleatorias para los parámetros, y repitiendo la simulación para distintas señales aleatorias gaussianas, se tiene una segmentación exitosa, con tasas de error menores al 0,5 % para ambas formas de estimación. En ese sentido no habría diferencia entre la optimización de máximo verosimil con la optimización variacional, pero, monitoreando la convergencia, se ve que con aprendizaje variacional, se llega mucho más rápido a a los parámetros óptimos. En la figura 5.2 se ve como en el algoritmo EM la conver-

gencia a la verosimilitud óptima ocurre en muchas más iteraciones que la Energía Libre $\mathscr{F}$ del aprendizaje variacional. Esta diferencia de tiempo (o iteraciones) de convergencia se va haciendo más evidente para modelos más complejos (de mayor cantidad de parámetros).

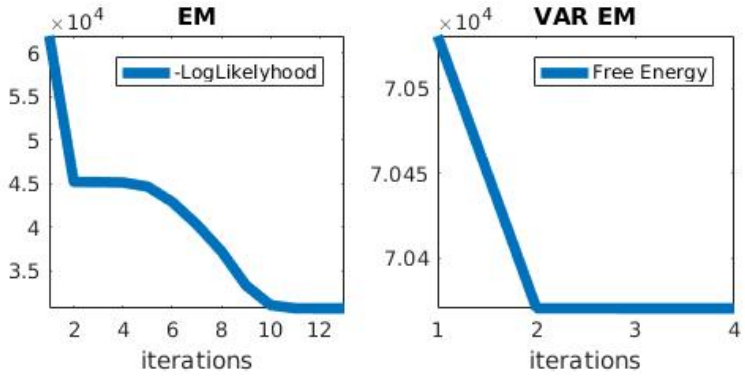

Figura 5.2: (izq.) valor de la Verosimilitud durante la optimización; (der.) valor de la Energía Libre durante la optimización

5.3. Estimación y selección de modelos HMM-MAR

Antes de calcular los parámetros del modelo, hay que considerar la inicialización de estos, para evitar dirigir la optimización a óptimos locales.
La forma de inicializar las matrices $W^{(m)}$ de los modelos MAR se hace calculando las matrices con un filtro de Kalman MAR, y posteriormente modelarlas con un GMM (de tantas distribuciones gaussianas como estados tenga el HMM), para usar los centroides encontrados como una predicción inicial de W^m. Los hiperparámetros λ_{mm} (en la diagonal) de la distribución a priori de la matriz de transición de estados $\pi_{t_{t-1}}$ son escogidos $\lambda_{mm} >> 1$, para reflejar que se conoce de antemano que los estados de la señal son persistentes (no ocurren muchas transiciones entre estados). Este supuesto se hace frecuentemente cuando se está trabajando con señales fisiológicas.

El resto de los parámetros se escogen de forma heurística, probando optimizar el modelo de forma rápida (con pocas iteraciones) para escoger como parámetros iniciales los que entreguen una menor energía libre $\mathscr{F}$ (ec 4.10).

La función *hmmmar.m*(13) implementa la optimización del modelo HMM-MAR (Aprendizaje Variacional en el capítulo 4), y la reconstrucción de la secuencia de estados con el algoritmo de Viterbi (como se muestra en la sección 3.2.2).

Se generan distintas secuencias de datos que simulan ser señales de EEG/MEG/fMRI/etc... como series de tiempo bivariadas generadas con modelos HMM-MAR de ordenes $P = \{2,3\}$

Tabla 5.1: Energía Libre obtenida para disintos modelos ajustados a un HMM-MAR con $p = 2$ y $K = 2$

	p=1	p=2	p=3	p=4	p=5
$K = 1$	25021,44	24929,80	24931,75	24932,14	24935,19
$K = 2$	18547,06	18381,14	18402,02	18416,33	18429,23
$K = 3$	18554,36	18391,38	18414,62	18432,67	18447,52
$K = 4$	18564,75	18404,92	18427,79	18446,33	18463,12
$K = 5$	18573,43	18412,94	18440,89	18461,49	18480,24

Tabla 5.2: Energía Libre obtenida para disintos modelos ajustados a un HMM-MAR con $p = 3$ y $K = 2$

	p=1	p=2	p=3	p=4	p=5
$K = 1$	32842,09	32590,82	31382,23	30956,59	30945,57
$K = 2$	28576,56	26253,81	25399,90	25418,15	25435,52
$K = 3$	28585,63	26245,42	25412,50	25432,73	25452,17
$K = 4$	28594,17	26274,84	25425,65	25450,82	25469,37
$K = 5$	28586,22	26268,90	25438,74	25463,17	25489,74

y cantidad de estados $K = \{2,3\}$, todos estos modelos sintéticos tienen matrices de transición $\pi_{t_{t-1}}$ con valores cercanos a uno en la diagonal para asegurar que haya alta permanencia en cada estado.

Para cada serie de tiempo se optimiza una colección de modelos HMM-MAR de distintos ordenes ($\hat{p} = \{1,2,3,4,5\}$) y cantidades de estados ($\hat{K} = \{1,2,3,4,5\}$) y se escogen los modelo con el que se obtiene la menor energía libre. En las tablas 5.3 a la 5.3 se puede ver que para cada serie de tiempo generada, el orden y cantidad de estados óptima corresponde a las de los modelos originales.

Para cada modelo óptimo se reconstruye la secuencia de estados con el algoritmo de viterbi, en las figuras 5.3 a las 5.6 se muestra como se recuperan las secuencias de estados con alta presición (la tabla 5.3 muestra la cantidad de aciertos para cada caso).

Los (pequeños) errores en la segmentación, se atribuyen a problemas que tiene el modelo HMM-MAR para detectar inmediatamente los cambios de estados, esto puede ser por la estructura autoregresiva del modelo, que provoca que en los intantes donde cambian los estados, se modelen las observaciones con muestras de dos estados simultaneamente (el estado al que se cambió, y el estado anterior). Además de esto, el modelo HMM-MAR tiende a fallar en

Tabla 5.3: Energía Libre obtenida para disintos modelos ajustados a un HMM-MAR con $p = 2$ y $K = 3$

	p=1	p=2	p=3	p=4	p=5
$K = 1$	34532,51	34352,90	34345,77	34351,01	34354,29
$K = 2$	30847,82	29982,71	29994,45	30010,97	30024,80
$K = 3$	29459,44	28438,03	28461,71	28485,17	28506,06
$K = 4$	29468,97	28450,49	28470,56	28503,52	28518,97
$K = 5$	29473,61	28458,03	28489,00	28514,05	28545,54

Tabla 5.4: Energía Libre obtenida para disintos modelos ajustados a un HMM-MAR con $p = 3$ y $K = 3$

	p=1	p=2	p=3	p=4	p=5
$K = 1$	31524,91	30333,25	30108,73	30099,67	30100,33
$K = 2$	28520,90	25703,93	24735,73	24673,62	26000,74
$K = 3$	26852,75	22234,01	21240,95	21268,24	21286,01
$K = 4$	26862,11	22217,99	21256,04	21284,68	21304,74
$K = 5$	26753,04	22232,42	21264,51	21295,24	21323,46

reconocer estados, cuando estos se mantienen por periodos muy cortos de tiempo, esta situacion se observa en el gráfico de la figura 5.6. Si bien esta situación no es problemática para este entorno con señales sintéticas, hay que tener cautela con estos factores si se está tratando con señales fisiológicas reales.

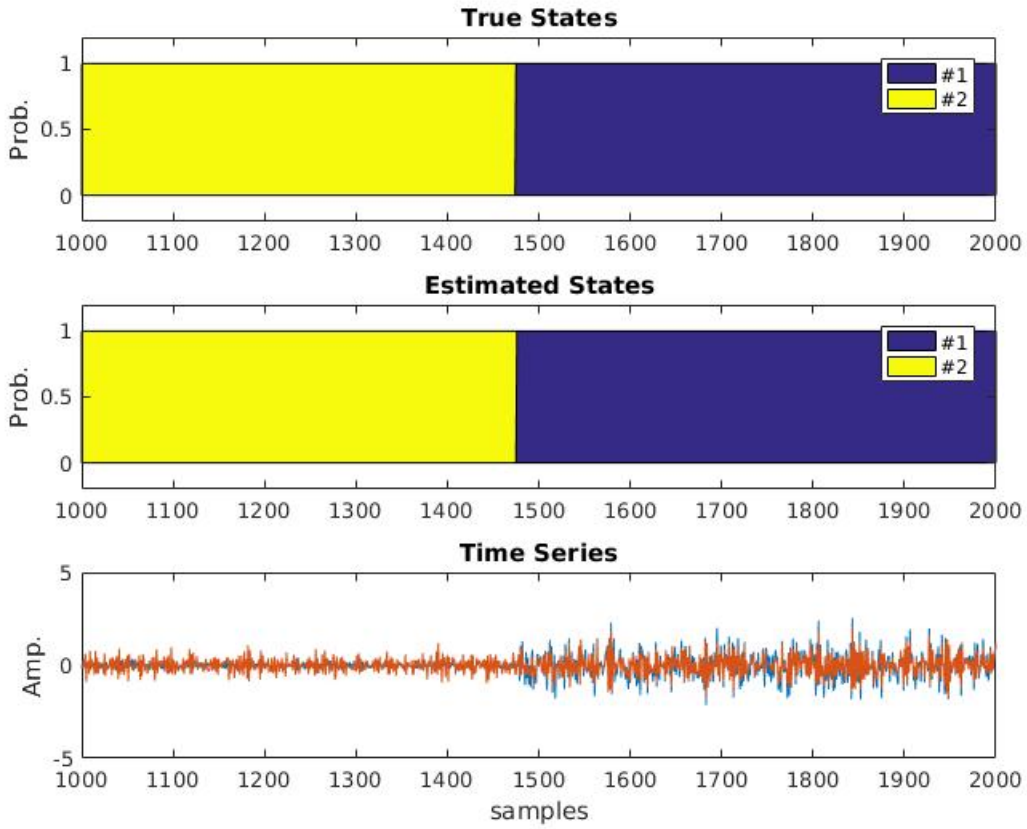

Figura 5.3: (de arriba a abajo) Secuencia original de estados; secuencia predecida de estados, señal sintética (HMM-MAR con K=2 y p=2)

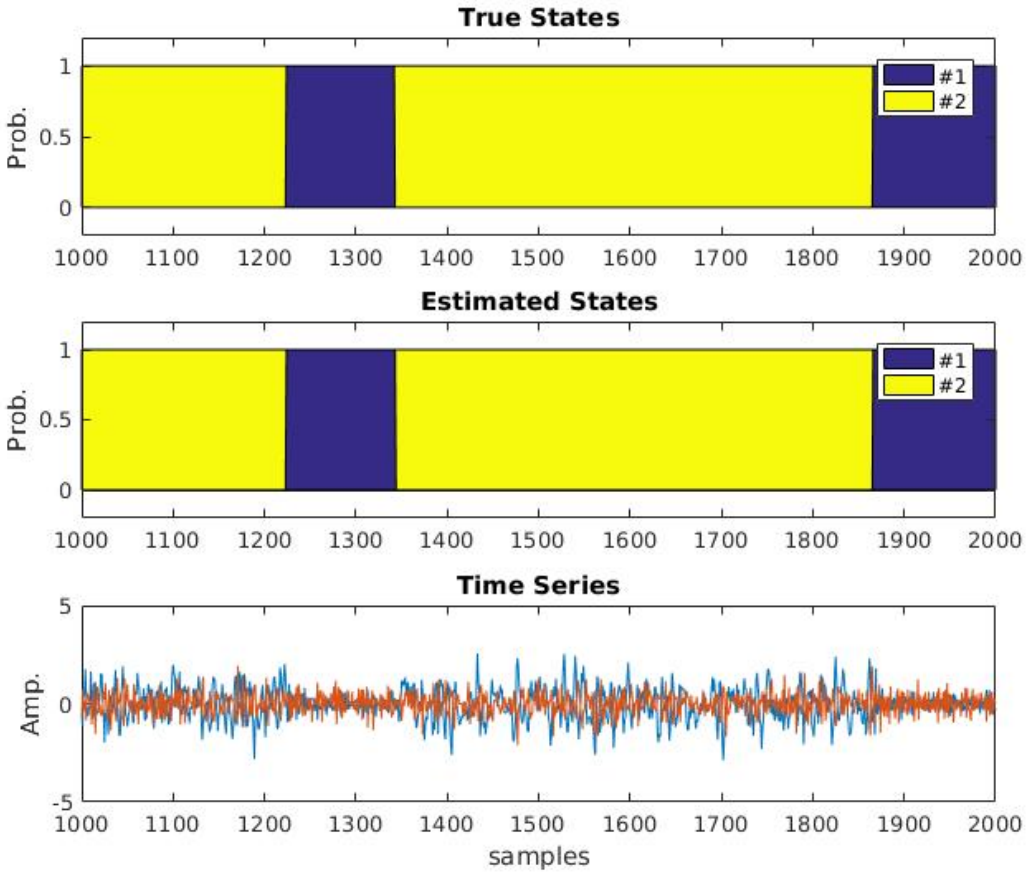

Figura 5.4: (de arriba a abajo) Secuencia original de estados; secuencia predecida de estados, señal sintética (HMM-MAR con K=2 y p=3)

Tabla 5.5: Porcentaje de aciertos de la predicción de secuencia de estados con el algoritmo de viterbi

(K,p)	(%) aciertos
$(2,2)$	99.30%
$(2,3)$	99.40%
$(3,2)$	99.92%
$(3,3)$	99,96%

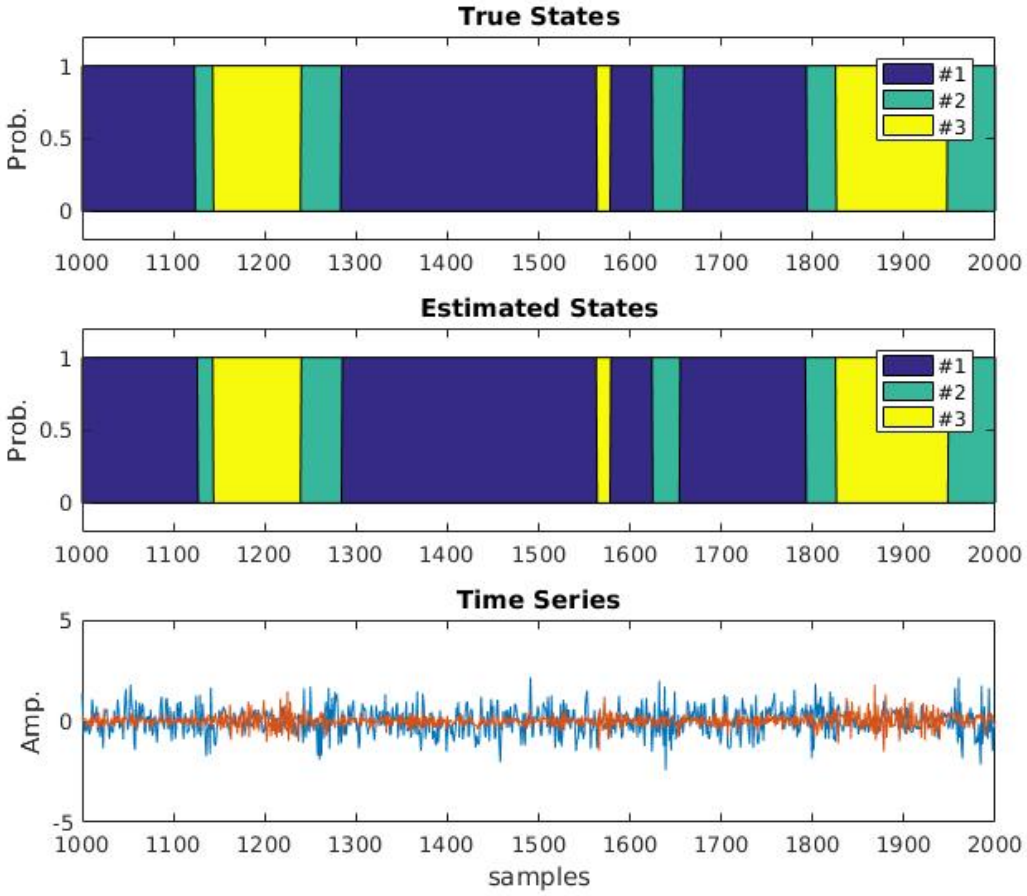

Figura 5.5: (de arriba a abajo) Secuencia original de estados; secuencia predecida de estados, señal sintética (HMM-MAR con K=3 y p=2)

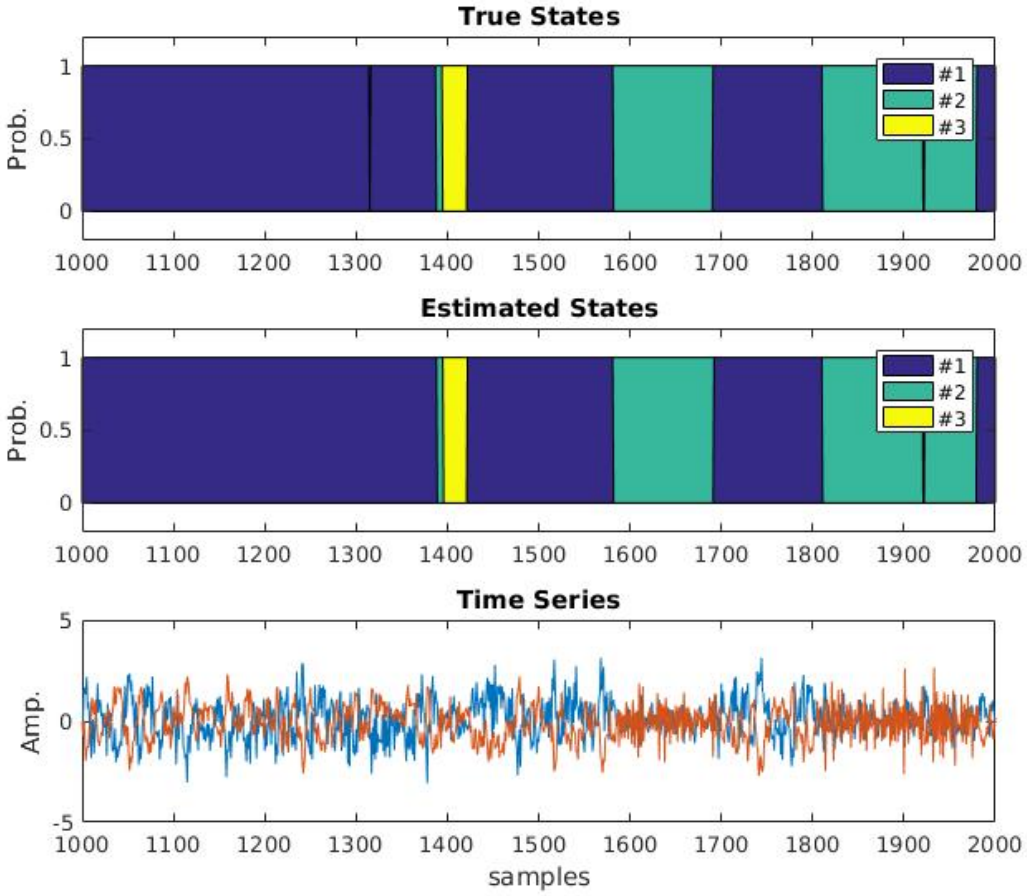

Figura 5.6: (de arriba a abajo) Secuencia original de estados; secuencia predecida de estados, señal sintética (HMM-MAR con K=3 y p=3)

Conclusiones

En este trabajo, se ha abordado el problema de segmentación de mediciones de señales cerebrales en reposo (sin tareas cognitivas especificas), haciendo una revisión de los modelos escondidos de Markov autoregresivos (HMM-MAR), que han sido frecuentemente utilizados para resolver problemas de segmentación en el area de señales fisiológicos.

Para estos modelos se han revisado dos perspectivas de optimización, de máximo verosimil y de aprendizaje variacional. Respaldado por las simulaciones (sección 5.2), donde se comparó el desempeño de la estimación de un modelo escondido de Markov sencillo, se han confirmado las ventajas del esquema variacional (mencionadas en (7)), para utilizarlo en el ajuste de modelos HMM-MAR.

Si bien este trabajo pretendía presentar resultados del ajuste del HMM-MAR en su versión variacional y la segmentación para mediciones reales de señales neuronales, en consideración del tiempo, en su lugar se ha propuesto un esquema de simulacion (sección 5.3) donde se puede probar la estimación de distintos HMM-MAR pueden ser generados aleatoriamente con cantidad de estados del HMM y orden del MAR asociado escogidos libremente.

En un entorno de datos sintéticos, el modelo HMM-MAR ajustado con aprendizaje variacional, las señales generadas son segmentadas correctamente, con tasas de acierto mayores al 99,0 %, generalmente fallando en segmentos donde las transiciones son muy rápidas (hay segmentos muy cortos de un solo estado). Además, se ha mostrado como el aprendizaje variacional ayuda a escoger correctamente la cantidad de parámetros para un modelo HMM-MAR, viendo como acierta al orden y cantidad de estados para distintas configuraciones de modelos HMM-MAR (sección 5.3).

Como trabajo futuro se puede evaluar la posiblidad de modificar la parte del modelo MAR para poder detectar transiciones más rápidas, como se propone en (8), donde se muestra como alternativa escoger los retardos p de la autoregresión de forma exponencial (por ejemplo, con retardos de potencias de 2 $\{t-2^1, t-2^2, t-2^3, ...\}$ en véz de $\{t-1, t-2, t-3, ...\}$). Además queda pendiente la tarea de proponer un expermiento donde se pueda probar el desempeño del modelo HMM-MAR para mediciones reales de señales cerebrales usando el esquema de simulación ya propuesto para hacer estimación, segmentación y selección de orden.

Bibliografía

[1] G.D. Forney; *"The Viterbi Algorithm"*; *Proc. IEEE, vol. 61*, pp 268-278, Mar. 1973

[2] Lawrence R. Rabiner; *"A tutorial on Hidden Markov Models and Selected Applications in Speech Recognition"*; *Proc. IEEE, vol. 77*, No.2, pp 257-286, Feb. 1989

[3] A.P. Dempster, N.M. Laird, D.B Rubin; *"Maximum Likelihood from Incomplete Data via the EM Algorithm"*; *Journal of the Royal Statistical Society, B Vol 39*, pp 1-38, 1977

[4] William D. Penny, Stephen J. Roberts; *"Dynamic Models for Nonstationary Signal Segmentation"*; *Computers and Biomedical Resarch, vol. 32* pp 483-502, 2000.

[5] T.M. Cover, J.A. Thomas; *"Elements of Information Theory"*; *John Wiley & Sons, New York*, 1991.

[6] M. Haft, R. Hofmann, V. Tresp; *"Model-independent mean field theory as a local method for approximate propagation of information"*; *Computation in Neural Systems*, chap. 10, pp 93-105, 1999.

[7] Rezek I., Roberts S.,2005; *"Ensemble Hidden Markov Models with extender observations densities for biosignal analysis"*; *Probabilistic Modeling in Bioinformatics and Medical Informatics. Advanced Information and Knowledge Processing*, pp. 419-450, 2005.

[8] Diego Vidaurre, Andrew J. Quinn, Adam P. Baker, David Dupret, Alvaro Tejero-Cantero, Mark W. Woolrich; *"Spectrally resolved fast transient brain states in electrophysiological data"*; *NeuroImage. Vol. 126*, pp. 81-95, 2016.

[9] Diego Vidaurre, Romesh Abeysuriya, Robert Becker, Andrew J. Quinn, F. Alfaro-Almagro, S.M Smith, Mark W. Woolrich; *"Discovering dynamic brain networks from big data in rest and task"*; *NeuroImage*, 2017.

[10] Diego Vidaurre, S.M. Smith, Mark W. Woolrich; *"Brain network dynamics are hierarchically organized in time"*; *Proceedings of the National Academy of Sciences of the USA*, 2017.

[11] Diego Vidaurre, Lawrence T. Hunt, Andrew J. Quinn, Benjamin A.E. Hunt, Matthew J. Brookes, Anna C. Nobre and Mark W. Woolrich; *"spontaneous cortical activity transiently organises into frequency specific phase-coupling networks"*; *BiorXiv*, 2017.

[12] A. K. Gupta, D. K. Nagar; *"Chapter 2: MATRIX VARIATE NORMAL DISTRIBUTION"*; *Matrix Variate Distributions. CRC Press.*, 1999.

[13] *"Toolbox for segmentation and characterisation of transient connectivity"*; https://github.com/OHBA-analysis/HMM-MAR.

[14] *"Probabilistic Modeling Toolkit for Matlab/Octave"* https://github.com/probml/pmtk3

[15] Bhar R. Hamori S.; *"Hidden Markov Models, Applications to Financial Econometric"*; *Advanced Studies in Theoretical and Applied Econometrics, vol. 40, Springer*, 2004.

[16] Sharma K.; *"Bioinformatics, Sequence Alignment and Markov Models"*; *McGraw-Hill Professional*, 2008.

[17] MacKay, David; *"An Example Inference Task: Clustering"*; *Information Theory: Inference and Learning Algorithms, Cambridge University Press*, chap. 20, pp. 284-292, 2003.

[18] D.A. Reynolds, R.C. Rose; *"Robust Text-Independent Speaker Identification using Gaussian Mixture Speaker Models"*; *IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 3, issue 1*, pp: 72-83, 1995.

[19] R. H. Shumway, D. S. Stoffer; *"Time Seires Analysis and Its Applications, with R examples"*; *Springer Texts in Statistics,* 3^{rd} *edition*, pp: 84-89, 2011.

[20] Monson H. Hayes; *"Statistical Digital Signal Processing and Modelling"*; *John Wiley & Sons*, pp: 108-115, 1996.

[21] Noman Naseer, Keum-Shik Hong; *"fNIRS-based brain-computer interfaces: a review"*; emphFrontiers in Human Neuroscience, vol 9., art. 3 2015.

[22] Iber C, Ancoli-Israel S, Chesson A, and Quan SF; *The AASM Manual for the Scoring of Sleep and Associated Events: Rules, Terminology and Technical Specifications, 1st ed.*; *American Academy of Sleep Medicine,* 1^{st}, 2007.

[23] Shing-Tai Pan, Chih-En Kuo, Jian-Hong Zeng and Sheng-Fu Liang; *"A transition-constrained discrete hidden Markov model for automatic sleep staging"*; *BioMedical Engineering OnLine*, 2012.

[24] B. Obermaier, C. Guger, C. Neuper, G. Pfurtscheller; *Hidden Markov models for online classification of single trial EEG data*; *Pattern Recognition, vol. 22*, pp. 1299-1309, 2001.

[25] Jagdish Chandra Joshi; *"Prediction of weather states using Hidden Markov model"*; *Conference: International Conference on Cryosphere and Climate Change*, 2012.

[26] W. Penny and S. Roberts; *"Gaussian Observation Hidden Markov models for EEG analysis"*; *Technical Report, Neural Systems Research Group, Department of Electrical and Electronic Engineering, Imperial College of Science, Technology and Medicine, London*, 1998.

[27] L. Harrison, W. Penny and K. Friston; *"Multivariate Autoregressive modelling of fMRI time series"*; *NeuroImage, vol. 19*, pp. 1477-1491, 2003.

[28] J.P. Banquet; *"Spectral Analysis of fMRI Signal and Noise"*; *Springer, Novel Trends in Brain Science,* pp. 63-76, 2008.

[29] P. Brockwell, R. Davis; *"Introduction to time series and forecasting"*; *Springer Texts on Statistics, 2nd Edition,* 2002.

Anexos

Anexo A

Diseño de un proceso mAR estable

Para comenzar, primero se lleva el modelo mAR(p) a la forma de un modelo mAR(1), condensando toda la información de las matrices $A_1, A_2, ..., A_p$ y su relación con las observaciones $Y(t), Y(t-1), ..., Y(t-p)$, en una sola matriz $\bar{A}$ (ecs. A.1 y A.2)

$$\bar{Y}(t) = \bar{A}\bar{Y}(t-1) + \bar{e}_t \tag{A.1}$$

$$\begin{pmatrix} Y(t) \\ Y(t-1) \\ \vdots \\ Y(t-(p-1)) \end{pmatrix} = \begin{pmatrix} A_1 & A_2 & A_3 & ... & A_p \\ I & 0 & 0 & ... & 0 \\ 0 & I & 0 & ... & 0 \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & ... & I & 0 \end{pmatrix} \begin{pmatrix} Y(t-1) \\ Y(t-2) \\ \vdots \\ Y(t-p) \end{pmatrix} + \bar{e}_t \tag{A.2}$$

Para asegurar que las matrices $A_1, ..., A_p$ generen un proceso mAR(p) estable, es necesario asegurar que la matriz $\bar{A}$ tenga todos sus valores propios λ_i dentro del circulo unitario. Para asegurar que se cumpla la estabilidad del sistema con la matriz $\bar{A}$, se busca una manera de generar la matriz $\bar{A}$, a partir de los valores propios λ_i (que se escogerán arbitrariamente, manteniendose en el circulo unitario).

Aprovechandose de propiedades de diagonalización de matrices, se puede escibir una matriz A cualquiera, en términos de sus valores y vectores propios (ec. A.3)

$$A = Q\Lambda Q^{-1} \quad \Lambda = \begin{pmatrix} \lambda_1 & 0 & ... & 0 \\ 0 & \lambda_2 & ... & 0 \\ \vdots & ... & \ddots & \vdots \\ 0 & 0 & ... & \lambda_{k*p} \end{pmatrix} \quad Q = \left(\vec{q_1}\vec{q_2}\ldots\vec{q_{p*k}}\right) \tag{A.3}$$

en la ec. A.3, cada q_i corresponde al un vector propio (columna) asociado al valor propio λ_i.

En la misma ecuación de diagonalización, multiplicando por la derecha por Q, se tiene que $AQ = Q\Lambda$, donde, se puede ver que se cumple $A\vec{q_i} = \vec{q_i}\lambda_i$, con esto, y definiendo arbitraria-

mente $q_i = [v_1 v_2 \dots v_p]^T \in R^{k*p}$ con cada $v_i \in R^k$, se puede escribir la relación como en la ec. A.4.

$$A\vec{q}_i = \begin{pmatrix} A_1 v_1 + A_2 v_2 + \cdots + A_p v_p \\ v_1 \\ v_2 \\ \vdots \\ v_{p-1} \end{pmatrix} = \begin{pmatrix} \lambda_i v_1 \\ \lambda_i v_2 \\ \vdots \\ \lambda_i v_p \end{pmatrix} = \lambda_i \vec{q}_i \tag{A.4}$$

de donde se desprende una dependencia de $v_1, \dots, v_{p-1}$ con v_p (ec. A.5)

$$\begin{gathered} v_1 = \lambda_i v_2 = \lambda_i^p v_p \\ v_2 = \lambda_i v_3 = \lambda_i^{p-1} v_p \\ \vdots \\ v_{p-2} = \lambda_i v_{p-1} = \lambda_i^2 v_p \\ v_{p-1} = \lambda_i v_p \end{gathered} \tag{A.5}$$

con esta relación basta solo con diseñar el vector v_p arbitrariamente, y escoger una colección de $k * p$ valores propios λ_i (dentro del circulo unitario) para poder crear los vectores $\vec{q}_i$ y contruir la matriz Q, con lo que se obtiene la matriz $\bar{A}$ con la ecuación de diagonalización $\bar{A} = Q\Lambda Q^{-1}$, de donde se obtienen directamente A_i.

Anexo B

Filtro de Kalman

El Filtro de Kalman es una algoritmo que busca, a partir de una señal observada x_t, poder hacer una estimación de una variable interna de un tipo de modelo que representa a dicha señal, a esta variable se le llama estado oculto, y al modelo se le dice modelo de espacio de estados. A continuación se presenta la derivación de las ecuaciones del Filtro de Kalman para hacer la estimación de estados ocultos para un modelo *AR(p)* y *mAR(p)*, para luego utilizarlos para la inicialización de los parámetros del modelo MAR.

B.1. Filtro de Kalman AR

En el parrafo anterior, se habló de las ecuaciones de espacio de estados, que es un tipo de modelo estadístico que consiste en un par de ecuaciones (matriciales) que describen la dependencia de un proceso observado con una variable interna del mismo (estado oculto), una de las ecuaciones corresponderá a la dinámica del estado ($\mathbf{a_t}$ en la ec. B.1) y la otra corresponde a la dinámica de la observación (x_t en ec. B.2), en esta última se ve explicitamente la dependencia del estado con la señal observada.

$$\mathbf{a_t} = G_t \mathbf{a_{t-1}} + v_t,\ v_t \sim N(0, \Sigma_{v_t}) \tag{B.1}$$

$$x_t = F_t \mathbf{a_t} + w_t,\ w_t \sim N(0, \sigma_t) \tag{B.2}$$

En este caso de representación de espacio de estados, las señales de error (w_t y v_t), tienen una función de distribución gaussiana, de media $\mu = 0$ y varianza σ_t (para w_t) y covarianza Σ_{v_t} para v_t conocidas. Este supuesto es importante para que las ecuaciones de estimación del Filtro de Kalman sean consistentes.

Este modelo de espacio de estados puede verse como el modelo *AR(p)* si se define a_t como el vector de coeficientes autoregresores $(a_1\ a_2\ ...\ a_p)_t^T$, la matriz de transición de la observación $F_t = -(x_{t-1}\ x_{t-2}\ ...\ x_{t-p})$, y la matriz de transición de estado como una matriz identidad ($G_t = I_{p\times p}$), las ecs. B.1 y B.2 quedan como las B.3 y B.4.

$$\mathbf{a_t} = \mathbf{a_{t-1}} + v_t,\ v_t \sim N(0, \Sigma_{v_t}) \tag{B.3}$$

$$x_t = - \begin{pmatrix} x_{t-1} \\ x_{t-2} \\ \vdots \\ x_{t-p} \end{pmatrix}^T \mathbf{a_t} + w_t,\ w_t \sim N(0, \sigma^2_{w_t}) \tag{B.4}$$

En esta representación de modelo AR(p)*, los coeficientes* AR *van cambiando junto con la dinámica de la observación, las ecuaciones del Filtro de Kalman se encargan de estimar los coeficientes de manera de seguir el comportamiento de la señal observada. Dicho de otra manera, para cada instante, el Filtro de Kalman, estimará un modelo* AR(p)*, para cada muestra observada de la señal* x_t.

Ahora en la ecuación B.4 se tiene explicitamente la representación de la dinámica de un modelo *AR(p)* para un instante t, y el estado oculto es el vector de coeficientes $\mathbf{a_t}$ que modelan el proceso autoregresivo en dicho instante. En la ecuación B.3, se asume que el estado, el vector los p coeficientes autoregresores $(a_1, ..., a_p)$, tiene una dinámica donde todas las variable son independientes entre si, dependen unicamente de su estado anterior y tiene una variación aleatoria v_t.

Como se muestra en (4), las ecuaciones del Filtro de Kalman para la estimación del estado $\mathbf{a_t}$ viene dada por las ecs. B.5 a la B.8. *En las ecuaciones, la predicción de la observación en un instante t, corresponde a* $\hat{x}_t = F_t \mathbf{a_t}$

$$\hat{\mathbf{a_t}} = \hat{\mathbf{a_{t-1}}} + K_t(x_t - \hat{x}_t) \tag{B.5}$$

$$K_t = \frac{(\Sigma_{t-1} + \Sigma_{v_t}) F_t^T}{\sigma^2_{\hat{x}_t}} \tag{B.6}$$

$$\Sigma_{\mathbf{a_t}} = \Sigma_{\mathbf{a_{t-1}}} + \Sigma_{v_t} - K_t F_t (\Sigma_{\mathbf{a_t}} + \Sigma_{v_t}) \tag{B.7}$$

$$\sigma^2_{\hat{x}_t} = \sigma^2_{w_t} + F_t (\Sigma_{\mathbf{a_t}} + \Sigma_{v_t}) F_t^T \tag{B.8}$$

Una cantidad útil a considerar es la *likelihood o verosimilitud* de una observación, dados los parámetros del modelo, que es una gaussiana con una media $\hat{x}_t = F_t \hat{\mathbf{a_{t-1}}}$, que se trata de la predicción de la observación en el tiempo t con el estado estimado $\hat{\mathbf{a_{t-1}}}$, y una varianza de la predicción de $\sigma^2_{\hat{x}_t}$ (ec. B.9).

$$p(x_t) = N(\hat{x}_t, \sigma^2_{x_t}) \tag{B.9}$$

B.2. Filtro de Kalman MAR

Recordando el modelo *mAR(p)* (ecs. 2.8 y 2.9), donde a diferencia del modelo *AR(p)*, en vez de coeficientes autoregresores a_i, se tienen matrices A_i de coeficientes, que capturan las interdependencias que tiene la observaciónes actuales de las multiples señales ($X_t = (x_{1,t}\ x_{2,t}\ ...\ x_{k,t})$) con sus valores anteriores, para las cuales, para estimar sus coeficientes con el Filtro de Kalman, hay que hacer un adecuado diseño de las ecuaciones de espacio de estados.

Para poder representar un proceso *mAR(p)* como el modelo de espacio de estados de las ecs. B.1 y B.2. Es necesario definir adecuadamente la matriz de transición (F_t) de la observación X_t y el vector de estados $\bar{\mathbf{A}}_\mathbf{t}$.

Antes del diseño de las ecuaciones de espacio de estados, considerar el siguiente desarrollo del modelo *mAR(p)* (este desarrollo algebráico será de utilidad para comprender mejor el que se presentará).
Si se hace la multiplicación (término a término) de la expresión del modelo *mAR(p)* (ec. 2.9), se llega a la ec. B.10.

$$\begin{pmatrix} x_{1,t} \\ x_{2,t} \\ \vdots \\ x_{k,t} \end{pmatrix} = \begin{pmatrix} \begin{bmatrix} x_{1,t-1}a_{11,1} + x_{2,t-1}a_{12,1} + ... + x_{k,t-1}a_{1k,1} + \\ x_{1,t-2}a_{11,2} + x_{2,t-2}a_{12,2} + ... + x_{k,t-2}a_{1k,2} + \\ \vdots \\ x_{1,t-p}a_{11,p} + x_{2,t-p}a_{12,p} + ... + x_{k,t-p}a_{1k,p} \end{bmatrix} \\ \begin{bmatrix} x_{1,t-1}a_{21,1} + x_{2,t-1}a_{22,1} + ... + x_{k,t-1}a_{2k,1} + \\ x_{1,t-2}a_{21,2} + x_{2,t-2}a_{22,2} + ... + x_{k,t-2}a_{2k,2} + \\ \vdots \\ x_{1,t-p}a_{21,p} + x_{2,t-p}a_{22,p} + ... + x_{k,t-p}a_{2k,p} \end{bmatrix} \\ \vdots \\ \begin{bmatrix} x_{1,t-1}a_{k1,1} + x_{2,t-1}a_{k2,1} + ... + x_{k,t-1}a_{kk,1} + \\ x_{1,t-2}a_{k1,2} + x_{2,t-2}a_{k2,2} + ... + x_{k,t-2}a_{kk,2} + \\ \vdots \\ x_{1,t-p}a_{k1,p} + x_{2,t-p}a_{k2,p} + ... + x_{k,t-p}a_{kk,p} \end{bmatrix} \end{pmatrix} \tag{B.10}$$

Si se observa detenidamente la equivalencia entre cada $x_{i,t}$ y su respectivo escalar (encerrado en corchetes) al lado derecho de la ecuación, se ve que todos comparten los terminos $x_{1,t-1}$, $x_{2,t-1}$, ..., $x_{k,t-1}$, $x_{1,t-2}$, $x_{2,t-2}$, ..., $x_{k,t-2}$, ..., $x_{1,t-p}$, $x_{2,t-p}$, ... y $x_{k,t-p}$ y lo que cambia en cada término es el sets de pesos $a_{i1,1}$ al $a_{ik,p}$.
Se define un nuevo vector auxiliar $\mathbf{a^i}$ (ec. B.11), este vector guarda las dependencias que tiene

la señal $x_{i,t}$ con sus valores pasados $x_{i,t-m}$ y los valores pasados de las otras señales $x_{j\neq i,t-m}$.

$$\mathbf{a^i} = (a_{i1,1}\ a_{i2,1}\ \ldots\ a_{ik,1}\ a_{i1,2}\ a_{i2,2}\ \ldots\ a_{ik,2}\ \ldots\ a_{i1,p}\ a_{i2,p}\ \ldots\ a_{ik,p})_t \in \mathbb{R}^{k*p\times 1} \tag{B.11}$$

Ahora, con el vector extendido $\tilde{X}_t = (X_{t-1}\ X_{t-2}\ X_{t-p}) \in \mathbb{R}^{1\times k*p}$ definido para escribir el modelo *mAR(p)* como en la ec. 2.9, se puede expresar cada una de las señales $x_{i,t}$ como producto de la ec. B.12

$$x_{i,t} = -\tilde{X}_t(\mathbf{a^i})_t^T \tag{B.12}$$

Con esta idea, se diseña la matriz de transición F_t como en la ec B.13 y el vector de estados $\bar{\mathbf{A}}_\mathbf{t}$ como en la ec. B.14.

$$F_t = -\begin{pmatrix} \tilde{X}_t & 0 & 0 & 0 \\ 0 & \tilde{X}_t & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \tilde{X}_t \end{pmatrix} \in \mathbb{R}^{k\times k*p*p} \tag{B.13}$$

$$\bar{\mathbf{A}}_\mathbf{t} = (\mathbf{a^1}\ \mathbf{a^2}\ \ldots\ \mathbf{a^k})_t^T \in \mathbb{R}^{k*k*p\times 1} \tag{B.14}$$

Con esto ya se pueden plantear las ecuaciones de espacio de estados para el modelo *mAR(p)* (ecs. B.15 y B.16)

$$\bar{\mathbf{A}}_\mathbf{t} = \bar{\mathbf{A}}_{\mathbf{t-1}} + V_t,\ V_t \sim N(0,\Sigma_{V_t}) \tag{B.15}$$

$$X_t = F_t\bar{\mathbf{A}}_\mathbf{t} + W_t,\ W_t \sim N(0,\Sigma_{W_t}) \tag{B.16}$$

Básicamente lo que se hizo fue "aplanar" la matriz $\mathbf{A}$ *del modelo* mAR(p) *en la forma de la ec. 2.9, y buscar una matriz* F_t *de forma que al multiplicarla por* $\mathbf{A}$*, resultara la señal* X_t.

Ahora que se tiene el modelo como espacio de estados, bastará con las ecuaciones del Filtro de Kalman (ecs. B.17 a la B.20) para poder estimar el estado $\bar{\mathbf{A}}_\mathbf{t}$ (los coeficientes de las matrices del modelo *mAR(p)*), con el cual se puede predecir las observaciones en un tiempo t, como $\hat{X}_t = F_t\bar{\mathbf{A}}_{\mathbf{t-1}}$.

$$\hat{A}_{t-1} = \hat{A}_{t-1} + K_t(X_t - \hat{X}_t) \tag{B.17}$$

$$K_t = (\Sigma_{W_{t-1}} + \Sigma_{V_t})F_t^T\Sigma_{\hat{X}_t}^{-1} \tag{B.18}$$

$$\Sigma_{\bar{\mathbf{A}}_\mathbf{t}} = \Sigma_{\bar{\mathbf{A}}_{\mathbf{t-1}}} + \Sigma_{V_t} - K_tF_t(\Sigma_{\bar{\mathbf{A}}_\mathbf{t}} + \Sigma_{v_t}) \tag{B.19}$$

$$\Sigma_{\hat{X}_t} = \Sigma_{W_t} + F_t(\Sigma_{\bar{\mathbf{A}}_\mathbf{t}} + \Sigma_{V_t})F_t^T \tag{B.20}$$

La verosimilitud de las observaciones, dado el modelo predecido, está dada por la ec. B.21.

$$p(X_t) = N(\hat{X}_t, \Sigma_{\hat{X}_t}) \tag{B.21}$$

Anexo C

Códigos para las simulaciones

C.1. Diseño de proceso MAR

```
function W_mat = generate_stable_MAR(K,p)
%
% Creates matrix of mAR model (dimension: K*p,K)
%
% INPUT:
% K: number of channels
% p: order of mAR model
%
% Author: Weinstein's PhD Adviser

% eigenvectors of A
Qmatrix = zeros(K*p,K*p);
% eigenvalues of A
LambdaMatrix = zeros(K*p,K*p);

for kk = 1:(K*p);
    % select one eigenvector/eigenvalue at a time.

    % random eigenvalue inside unit circle
    %lambda = randn + 1j*randn; lambda = rand*lambda/norm(lambda);
    lambda = randn; lambda = rand*lambda/norm(lambda);
    % the eigenvector has the form [u_1; u_2; ...; u_p], where
    % u_1 = lambda*u_2, u_2 = lambda*u_3, ..., u_{p-1} = lambda*u_p
    % choose the last block randomly and then all blocks are determined.
    upBlock = randn(K,1);
    eigvec = zeros(K*p,1);
    eigvec(K*(p-1)+1:K*p) = upBlock;
    for pp = (p-1):-1:1;
        eigvec(K*(pp-1)+1:K*pp) = lambda*eigvec(K*pp+1:K*(pp+1));
    end
```

```
    % populate column kk of the Q and Lambda matrices
    Qmatrix(:,kk) = eigvec;
    LambdaMatrix(kk,kk) = lambda;
end

% construct A
A = Qmatrix*LambdaMatrix*inv(Qmatrix);

% check to make sure eigenvalue magnitudes are less than one
if max(abs(eig(A))) < 1
    fprintf('Matrix A is stable, max eigenvalue magnitude is %4.3f.\n', max(abs(eig(A))))
else
    error('Matrix A is NOT stable.');
end

% figure(1); clf; imagesc(abs(A)); title('A matrix (magnitude)'); axis square;

% figure(2); clf; scatter(real(eig(A)), imag(eig(A))); axis square; hold on;
% theta = linspace(0, 2*pi, 1000); x = cos(theta); y = sin(theta); plot(x, y);
% title('eigenvalues of A');

% figure(3); clf; imagesc(abs(Qmatrix)); title('eigenvectors of A (magnitude)');
W_mat = zeros(K*p,K);
for ii = 1:p;
        W_mat(((ii-1)*K+1):ii*K,1:K) = A(1:K,((ii-1)*K+1):ii*K)';
end
W_mat = W_mat/max(max(abs(W_mat)));

end
```

[language=Octave]

C.2. Simulación de distintos procesos HMM-MAR y selección de modelos

```
function W_mat = generate_stable_MAR(K,p)
%
% Creates matrix of mAR model (dimension: K*p,K)
%
% INPUT:
% K: number of channels
% p: order of mAR model
%
% Author: Weinstein's PhD Adviser

% eigenvectors of A
Qmatrix = zeros(K*p,K*p);
% eigenvalues of A
LambdaMatrix = zeros(K*p,K*p);
```

```
for kk = 1:(K*p);
    % select one eigenvector/eigenvalue at a time.

    % random eigenvalue inside unit circle
    %lambda = randn + 1j*randn; lambda = rand*lambda/norm(lambda);
    lambda = randn; lambda = rand*lambda/norm(lambda);
    % the eigenvector has the form [u_1; u_2; ...; u_p], where
    % u_1 = lambda*u_2, u_2 = lambda*u_3, ..., u_{p-1} = lambda*u_p
    % choose the last block randomly and then all blocks are determined.
    upBlock = randn(K,1);
    eigvec = zeros(K*p,1);
    eigvec(K*(p-1)+1:K*p) = upBlock;
    for pp = (p-1):-1:1;
        eigvec(K*(pp-1)+1:K*pp) = lambda*eigvec(K*pp+1:K*(pp+1));
    end

    % populate column kk of the Q and Lambda matrices
    Qmatrix(:,kk) = eigvec;
    LambdaMatrix(kk,kk) = lambda;
end

% construct A
A = Qmatrix*LambdaMatrix*inv(Qmatrix);

% check to make sure eigenvalue magnitudes are less than one
if max(abs(eig(A))) < 1
    fprintf('Matrix A is stable, max eigenvalue magnitude is %4.3f.\n', max(abs(eig(A)))
else
    error('Matrix A is NOT stable.');
end

% figure(1); clf; imagesc(abs(A)); title('A matrix (magnitude)'); axis square;

% figure(2); clf; scatter(real(eig(A)), imag(eig(A))); axis square; hold on;
% theta = linspace(0, 2*pi, 1000); x = cos(theta); y = sin(theta); plot(x, y);
% title('eigenvalues of A');

% figure(3); clf; imagesc(abs(Qmatrix)); title('eigenvectors of A (magnitude)');
W_mat = zeros(K*p,K);
for ii = 1:p;
        W_mat(((ii-1)*K+1):ii*K,1:K) = A(1:K,((ii-1)*K+1):ii*K)';
end
W_mat = W_mat/max(max(abs(W_mat)));

end


addpath(genpath('MATLOV/paketes_nueos/OHBA_HMMMAR/'))
addpath(genpath('PEGA/AC3E/Memoria/variational_HMMMAR/'))

%%%aqui se cambia la el tipo de modelo HMM-MAR para cada simu%%%
K = 3; % number of states
```

```
p = 3; % order of MAR
ndim = 2; % number of channels
%%%aqui se cambia la el tipo de modelo HMM-MAR para cada simu%%%

N = 1; % number of trials
Fs = 200;
T = 10000 * ones(N,1); % number of datagam points

hmmtrue = struct(); %la simu del paper de los qliaos pulentos
hmmtrue.K = K;
hmmtrue.train.Fs = Fs;
hmmtrue.train.covtype = 'full';
hmmtrue.train.zeromean = 1;
hmmtrue.train.order = p;
hmmtrue.train.orderoffset = 0;
hmmtrue.train.timelag = 1;
hmmtrue.train.exptimelag = 0;
hmmtrue.train.S = ones(ndim);
hmmtrue.train.Sind = ones(1,ndim);
hmmtrue.train.multipleConf = 0;

%% Generate Data

hmmtrue.P = rand(K) + 100 * eye(K);
for j=1:K
    hmmtrue.P(j,:) = hmmtrue.P(j,:) ./ sum(hmmtrue.P(j,:));
end;
hmmtrue.Pi = rand(1,K);
hmmtrue.Pi = hmmtrue.Pi./sum(hmmtrue.Pi);

for k = 1:K
    hmmtrue.state(k).W.Mu_W = generate_stable_MAR(ndim,p)*0.7; % generados por yo
    r = randn(ndim);
    hmmtrue.state(k).Omega.Gam_shape = 100;
    hmmtrue.state(k).Omega.Gam_rate = 0.1 * hmmtrue.state(k).Omega.Gam_shape ...
                                      * r' * r + eye(ndim);
end

[X,T,Gammatrue] = simhmmmar(T,hmmtrue,[],1,0);
figure(1);
subplot(2,1,2);
plot(X); xlabel('samples'); xlim([1000 2000]);
ylabel('Amp.'); title('Time␣Series')
subplot(2,1,1);
area(Gammatrue); xlim([1000 2000])
ylabel('Prob'); title('True␣States')
legend('#1','#2','#3')

%% Training a HMM model with Gaussian states:
options = struct();

options.Fs = Fs;
```

```
options.covtype = 'full';
options.DirichletDiag = 1000;
options.zeromean = 1;
options.verbose = 1;
options.symmetricprior = 0;
options.updateGamma = 1;
options.dropstates = 0;
options.initrep = 5;

minp=1; maxp=5;
mink=1; maxk=5;
FE=zeros((maxk-mink),(maxp-minp));

hmm = {}; Gamma = {}; Xi = {}; vpath = {};

for kk = mink:maxk
    options.K=kk;
    kay = kk-mink+1;
    for pp = minp:maxp
        pee = pp-minp+1;
        options.order = pp;
        [hmm{kay,pee}, Gamma{kay,pee}, ...
        Xi{kay,pee}, vpath{kay,pee}] = hmmmar(X,T,options);
        FE(kay,pee) = hmmfe(X,T,hmm{kay,pee},Gamma{kay,pee},Xi{kay,pee});
        if(pp == minp)
            hmmdef=hmm{kay,pee}; Gammadef=Gamma{kay,pee};
            Xidef=Xi{kay,pee}; vpathdef=vpath{kay,pee};
            FEmin = FE(kay,pee);
        else
            if FE(kay,pee) < FEmin
                hmmdef=hmm{kay,pee}; Gammadef=Gamma{kay,pee};
                Xidef=Xi{kay,pee}; vpathdef=vpath{kay,pee};
                FEmin = FE(kay,pee);
            end
        end
        fprintf('K = %d, ',kk); fprintf('p=%d ',pp);
        fprintf('-> Free Energy: %5f',FE(kay,pee)); fprintf('\n');
    end
end

%% Plot (a segment of the) true state path
figure(10); subplot(3,1,1);
INIT=1000; END=2000;

samples = floor(linspace(INIT,END,(END-INIT)));

Gammatru=[Gammatrue(:,1),Gammatrue(:,3),Gammatrue(:,2)];

Gammawea=multinomipath(vpath{K,p},K);
area(samples,Gammatru(samples,:)),
set(gca,'Title',text('String','True States'))
set(gca,'ylim',[-0.2 1.2]); ylabel('Prob.')
```

```
legend('#1','#2','#3')

subplot(3,1,2)
area(samples,Gammawea(samples,:)), set(gca,'Title',text('String','Estimated States'))
set(gca,'ylim',[-0.2 1.2]); ylabel('Prob.')
legend('#1','#2','#3')

subplot(3,1,3)
plot(samples,X(samples,:))
ylabel('Amp.'); xlabel('samples'); title('Time Series')

%% ver aciertos
arciertos(Gammatru,Gammawea)
```

C.3. Comparación de aprendizaje variacional y algoritmo EM

```
%% VARIATIONAL

addpath(genpath('MATLOV/paketes_nueos/OHBA_HMMMAR/'))
addpath(genpath('PEGA/AC3E/Memoria/variational_HMMMAR/'))

K = 3; % number of states
p = 0; % order of MAR
ndim = 5; % number of chanels
N = 1; % number of trials
Fs = 200;
T = 10000 * ones(N,1); % number of datagam points

hmmtrue = struct(); %da simu del paper de los qliaos pulentos
hmmtrue.K = K;
hmmtrue.train.Fs = Fs;
hmmtrue.train.covtype = 'full';
hmmtrue.train.zeromean = 1;
hmmtrue.train.order = p;
hmmtrue.train.orderoffset = 0;
hmmtrue.train.timelag = 1;
hmmtrue.train.exptimelag = 0;
hmmtrue.train.S = ones(ndim);
hmmtrue.train.Sind = ones(1,ndim);
hmmtrue.train.multipleConf = 0;

%% Generate Data

hmmtrue.P = rand(K) + 100 * eye(K);
for j=1:K
    hmmtrue.P(j,:) = hmmtrue.P(j,:) ./ sum(hmmtrue.P(j,:));
end;
hmmtrue.Pi = rand(1,K);
hmmtrue.Pi = hmmtrue.Pi./sum(hmmtrue.Pi);
```

```
for k = 1:K
    r = randn(ndim);
    hmmtrue.state(k).W.Mu_W = randn(1,ndim);
    hmmtrue.state(k).Omega.Gam_shape = 100;
    hmmtrue.state(k).Omega.Gam_rate = 0.1 * hmmtrue.state(k).Omega.Gam_shape * ...
                                      r' * r + eye(ndim);
end

[X,T,Gammatrue] = simhmmmar(T,hmmtrue,[],1,0);
figure(1);
subplot(2,1,2);
plot(X); xlabel('samples'); xlim([1000 2000]);
ylabel('Amp.'); title('Time Series')
subplot(2,1,1);
area(Gammatrue); xlim([1000 2000])
ylabel('Prob'); title('True States')
legend('#1','#2','#3')

%% Training a HMM model with Gaussian states:
options = struct();

options.Fs = Fs;
options.covtype = 'full';
options.DirichletDiag = 1000;
options.zeromean = 0;
options.verbose = 1;
options.symmetricprior = 0;
options.updateGamma = 1;
options.dropstates = 0;
options.initrep = 5;
options.order = 0;
options.K = K;

FE=0;

[hmm, Gamma, Xi, vpath] = hmmmar(X,T,options);
FE = hmmfe(X,T,hmm,Gamma,Xi);

%% Plot (a segment of the) true state path

figure(10); subplot(3,1,1)
INIT=500; END=3500;

samples = floor(linspace(INIT,END,(END-INIT)));

Gammatru=[Gammatrue(:,2),Gammatrue(:,3),Gammatrue(:,1)];

Gammawea=multinomipath(vpath,K);
area(samples,Gammatru(samples,:)),
set(gca,'Title',text('String','True States'))
set(gca,'ylim',[-0.2 1.2]); ylabel('Prob.')
```

```
legend('#1','#2','#3')

subplot(3,1,2)
area(samples,Gammawea(samples,:)),
set(gca,'Title',text('String','Estimated States'))
set(gca,'ylim',[-0.2 1.2]); ylabel('Prob.')
legend('#1','#2','#3')

subplot(3,1,3)
plot(samples,X(samples,:))
ylabel('Amp.'); xlabel('samples'); title('Time Series')

%% ver aciertos
aciertokos = arciertos(Gammatru,Gammawea);

%% Simple test of hmmFitEm with Gaussian observations

% acuerdate de cargar el workspace K3P3
addpath(genpath('MATLOV/paketes_nueos/PMTK3/'));
% ndim, K y X ya est\'an del workspace cargado

prior.mu = randn(1,ndim);
prior.Sigma = abs(randn*eye(ndim)+randpd(ndim)/100)

prior.k = ndim;
prior.dof = prior.k + 1;

pipriori = [1 1 1];

model = hmmFitEm(X', K, 'gauss', 'verbose', true, 'piPrior', pipriori, ...
    'emissionPrior', prior, 'nRandomRestarts', 10, 'maxIter', 200);

pat=hmmMap(model,X');
patwea=multinomipath(pat,K);
area(samples,patwea(samples,:));
ylabel('Amp.'); xlabel('samples'); title('Time Series');

%%
figure(10);
INIT=2000; END=5000;
samples = floor(linspace(INIT,END,(END-INIT)));

subplot(3,1,1);
area(samples,Gammatru(samples,:)), set(gca,'Title',text('String','True States'))
set(gca,'ylim',[-0.2 1.2]); ylabel('Prob.')
legend('#1','#2','#3')

subplot(3,1,2)
area(samples,Gammawea(samples,:)), set(gca,'Title',text('String','Estimated States'))
set(gca,'ylim',[-0.2 1.2]); ylabel('Prob.')
legend('#1','#2','#3')
```

```
patweatru=[patwea(:,2),patwea(:,3),patwea(:,1)];
subplot(3,1,3)
area(samples,patweatru(samples,:)), set(gca,'Title',text('String','Estimated States'))
set(gca,'ylim',[-0.2 1.2]); ylabel('Prob.')
legend('#1','#2','#3')
%%
aciertokosEM = arciertos(Gammatru,patweatru);

%% comparing convergence
loglikhist = [61957.8 45196.7 45191.7 45136.7 44666.2 ...
              42899 40322.8 37252.1 33327.6 31067.3 ...
              30709.4 30694.7 30694.7];

fehist = [70531.1 70370.2 70370.1 70370.1];

figure();

subplot(1,2,1); plot(loglikhist,'LineWidth',5); title('EM')
xlabel('iterations'); legend('-LogLikelyhood'); axis('tight')

subplot(1,2,2); plot(fehist,'Linewidth',5); title('VAR EM')
xlabel('iterations'); legend('Free Energy'); axis('tight')
```