BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS

# PROCESOS DE MARKOV APLICADOS AL ESTUDIO DE SECUENCIAS DE ADN

T E S I S

Presentada al

**Posgrado de Matemáticas**

como requisito parcial para la obtención del grado de

**Maestro en Ciencias**

Por:

Aranzazú Ortega Ponce

Director de tesis:

Hugo Adán Cruz Suárez

Agosto, 2015

# Índice general

# Capítulo 1

# ADN (Ácido Desoxirribonucleico)

El ADN, es un ácido nucleico que contiene la información genética de los seres vivos, por la cual se sintetizan las proteínas y se desarrollan. Visto desde el punto de vista químico es un polímero de nucleótidos, es decir, una macromolécula formada por la unión de nucleótidos, donde a su vez, cada nucleótido es una molécula formada por una pentosa, una base nitrogenada y un grupo fostato, como se muestra en la Figura 1.1.

En el caso del ADN la pentosa es la desoxirribosa y la base nitrogenada puede ser Adenina, Citosina, Guanina o Timina (A,C, T, G); la Adenina sólo puede unirse a la Timina y la Guanina a la Citosina, por eso decimos que la Adenina es complementaria de la Timina y la Citosina es complementaria de la Guanina. Así pues, lo que distingue a un nucleótido de otro es, entonces, la base nitrogenada; y por ello la secuencia del ADN se especifica nombrando sólo la secuencia de sus bases. Un ejemplo de secuencia de ADN puede ser ATCCGATTAAGCTCATGGCT.

En 1953, el bioquímico estadounidense James Watson y el biofísico británico Francis Crick publicaron la primera descripción de la estructura del ADN. Su modelo adquirió tal importancia, que los científicos obtuvieron en 1962 el Premio Nobel de Medicina por su trabajo. La estructura del ADN es una especie de escalera retorcida denominada *doble*

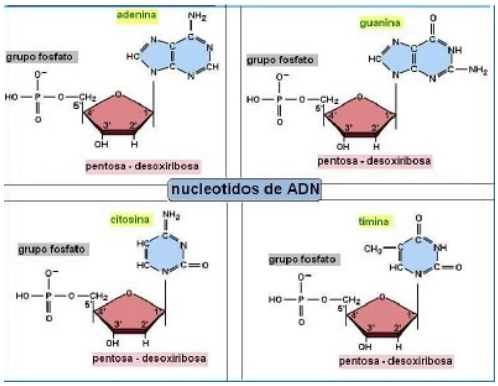


Figura 1.1: Nucleótidos del ADN

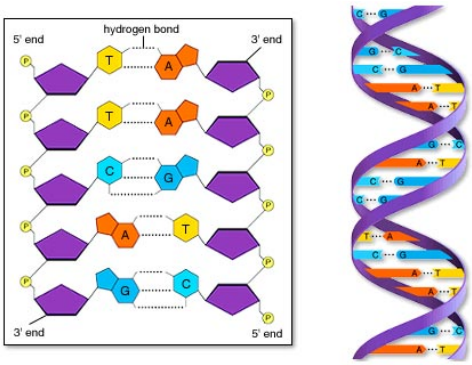


Figura 1.2: Estructura del ADN

*hélice*, es llamada de esta forma pues está formada por dos hebras cada una formada por una secuencia de nucleótidos encadenados, donde cada hebra contiene toda la información presente en la otra y están unidas entre sí por unas conexiones denominadas puentes de hidrógeno.
En concreto, la molécula de desoxirribosa ocupa el centro del nucleótido, de un lado tiene un grupo fosfato y del otro una base. El grupo fosfato está a su vez unido a la desoxirribosa del nucleótido adyacente de la cadena. Estas subunidades enlazadas desoxirribosa-fosfato forman los lados de la escalera; las bases están enfrentadas por parejas, mirando hacia el interior, y forman los escalones como se muestra en la Figura 1.2.

En la célula, el ADN está organizado en estructuras llamadas cromosomas. Los cromosomas varían ampliamente entre los diferentes organismos. Las células humanas contienen 23 pares de cromosomas, un miembro de cada par heredado de la madre y el otro del padre. Los dos cromosomas en un par son prácticamente idénticos, con la excepción del cromosoma sexual, para el cual hay dos tipos, X e Y (XX para la mujer y XY para el hombre, siendo la pareja la que determina el sexo). Cada célula del cuerpo contiene copias idénticas del conjunto entero de 23 pares de cromosomas. En los organismos eucariotas (por ejemplo, animales, plantas y hongos) la mayor parte del ADN se almacena dentro del núcleo celular y una mínima parte en elementos celulares llamados mitocondrias, y en los plastos y los centros organizadores de microtúbulos o centríolos, en caso de tenerlos. Los organismos procariotas (bacterias y arqueas) lo almacenan en el citoplasma de la célula. Podemos ver la situación del ADN dentro de la célula en la Figura 1.3.

Podríamos hacer la comparación del ADN con una receta, puesto que contiene la información genética necesaria para el desarrollo del organismo. Sin embargo para poder hacer uso de esa información debemos copiar la información del ADN en moléculas de ARN mediante un proceso llamado *transcripción*, éstas una vez procesadas en el ncleo celular, pueden salir al citoplasma para su utilización posterior.

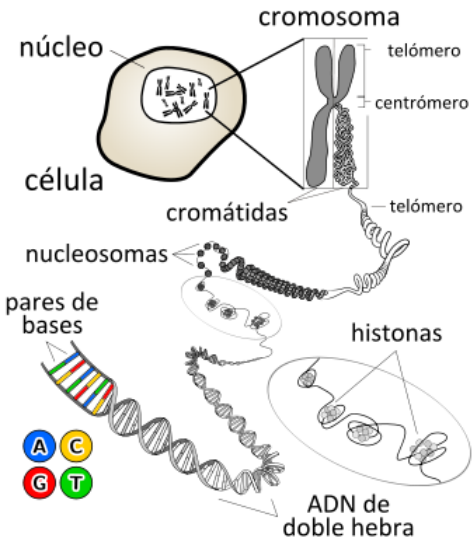


Figura 1.3: ADN dentro de la célula

## 1.1. Funciones Biológicas del ADN

Las funciones biológicas del ADN incluyen el almacenamiento de información (genes y genoma), la codificación de proteínas (transcripción y traducción) y su autoduplicación (replicación del ADN) para asegurar la transmisión de la información a las células hijas durante la división celular.

### 1.1.1. Genes y Genoma

El ADN se puede considerar como un almacén cuyo contenido es la información (mensaje) necesaria para construir y sostener el organismo en el que está presente, ésta se transmite de generación en generación. Al conjunto de información que cumple esta función se le denomina genoma, y el ADN que lo constituye, ADN genómico.

La información genética de un genoma está contenida en los genes, y al conjunto de toda la información que corresponde a un organismo se le denomina su genotipo. Un gen es una unidad de herencia que influye en una característica particular de un organismo (por ejemplo en el color de los ojos, cabello), ya que son éstos las secuencias de ADN que constituyen la unidad fundamental, física y funcional de la herencia. Contienen un "marco de lectura abierto" (open reading frame) que puede transcribirse, además de secuencias reguladoras, tales como promotores y enhancers, que controlan la transcripción del marco de lectura abierto. La estructura general de un gen puede verse en la Figura 1.4.

Desde este punto de vista, las obreras de este mecanismo son las proteínas. Estas pueden ser estructurales, como las proteínas de los músculos, cartílagos, pelo, etc., o funcionales, como la hemoglobina o las innumerables enzimas del organismo. La función principal de la herencia es la especificación de las proteínas, siendo el ADN una especie de plano o rece-

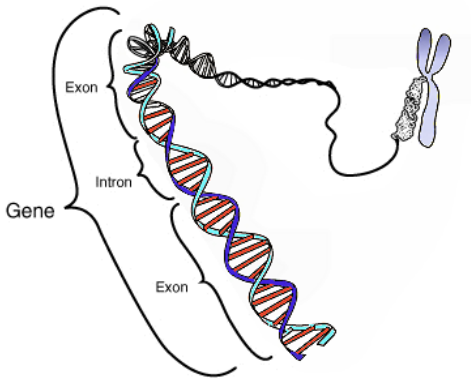


Figura 1.4: Estructura del gen

ta para producirlas. La mayor parte de las veces la modificación del ADN provocará una disfunción proteica que dará lugar a la aparición de alguna enfermedad. Pero en determinadas ocasiones, las modificaciones podrán provocar cambios beneficiosos que darán lugar a individuos mejor adaptados a su entorno.

Las aproximadamente treinta mil proteínas diferentes en el cuerpo humano están constituidas por veinte aminoácidos diferentes, y una molécula de ADN debe especificar la secuencia en que se unen dichos aminoácidos.

En el proceso de elaborar una proteína, el ADN de un gen se lee y se transcribe a ARN. Este ARN sirve como mensajero entre el ADN y la maquinaria que elaborará las proteínas y por eso recibe el nombre de ARN mensajero o ARNm. El ARN mensajero sirve de molde a la maquinaria que elabora las proteínas, para que ensamble los aminoácidos en el orden preciso para armar la proteína.

El dogma central de la biología molecular establecía que el flujo de actividad y de información era: ADN $\rightarrow$ ARN $\rightarrow$ proteína. No obstante, en la actualidad ha quedado demostrado que este "dogma" debe ser ampliado, pues se han encontrado otros flujos de información: en algunos organismos (virus de ARN) la información fluye de ARN a ADN; este proceso se conoce como "transcripción inversa o reversa", también llamada "retrotranscripción". Además, se sabe que existen secuencias de ADN que se transcriben a ARN y son funcionales como tales, sin llegar a traducirse nunca a proteína: son los ARN no codificantes.

### 1.1.2. Transcripción y Traducción

Uno de los paradigmas principales de la biología establece que a partir de la información genética codificada en el ADN podemos obtener proteínas, el paradigma también establece que hay una molécula intermedia entre el ADN que está en el núcleo y las proteínas, las cuales están presentes principalmente en el citosol de la célula, esa molécula intermedia es conocida como ARN y es la intermediaria, pues es capaz de salir del núcleo, llegar al citosol y generar las proteínas. El proceso por el cual transferimos el ADN hacia el ARN

se le conoce como transcripción del ADN y el proceso por el cual transferimos el ARN y lo codificamos en proteínas se conoce como la traducción.

La secuencia conocida como promotor se une a los llamados factores de transcripción y marcan el inicio del proceso, una vez que están unidas, la enzima llamada ARN polimerasa separa los segmentos del ADN para preparar la llegada de los nucleótidos complementarios del ARN y así se forma la nueva molécula ARN mensajero o ARNm, cabe mencionar que la diferencia entre los nucleótidos de ADN y ARN radica en el azúcar y una de las bases que es reemplazada por Uracilo. Por otra parte la célula produce el ARN de transferencia (ARNt) y ARN ribosomal (ARNr) las cuales en su forma final juegan un rol vital en el proceso de traducción y por lo tanto, en la síntesis de proteínas.

### 1.1.3. El ADN no Codificante

El ADN del genoma de un organismo puede dividirse conceptualmente en dos: el que codifica las proteínas (los genes) y el que no codifica. Sólo alrededor del 1,5 % del genoma humano consiste en exones que codifican proteínas (20.000 a 25.000 genes), mientras que más del 90 % consiste en ADN no codificante.

El ADN no codificante (también denominado ADN basura o junk DNA) corresponde a secuencias del genoma que no generan una proteína (procedentes de transposiciones, duplicaciones, translocaciones y recombinaciones de virus, etc.), incluyendo los intrones. Hasta hace poco tiempo se pensaba que el ADN no codificante no tenía utilidad alguna, pero estudios recientes indican que eso es inexacto. Entre otras funciones, se postula que el llamado "ADN basura" regula la expresón diferencial de los genes. Por ejemplo, algunas secuencias tienen afinidad hacia proteínas especiales que tienen la capacidad de unirse al ADN (como los homeodominios, los complejos receptores de hormonas esteroides, etc.), con un papel importante en el control de los mecanismos de trascripción y replicación. Estas secuencias se llaman frecuentemente "secuencias reguladoras", y los investigadores suponen que sólo se ha identificado una pequeña fracción de las que realmente existen. Un grupo de investigadores de la Universidad de Yale ha descubierto una secuencia de ADN no codificante que sería la responsable de que los seres humanos hayan desarrollado la capacidad de agarrar y/o manipular objetos o herramientas.

# Capítulo 2

# Análisis de las Secuencias de ADN

Como hemos dicho, el ADN puede representarse como secuencias de nucleótidos. Dentro de estas secuencias hay secuencias codificantes (gen), separadas por largas regiones intergénicas de secuencias no codificantes. La mayoría de los genes eucariotas tienen un nivel adicional de organización: dentro de cada gen, las secuencias codificantes, conocidas como exones, son interrumpidas por tramos de secuencias no codificantes, conocidas como intrones. Durante la transcripción del ADN en ARNm, estos últimos se eliminan mediante el proceso de maduración en la etapa conocida como splicing. Las regiones intergénicas y los intrones tienen diferentes propiedades estadísticas que los exones, para identificarlas podemos utilizar procedimientos estadísticos y comprobar si una parte no caracterizada de ADN es parte de la región codificante del gen. El modelo está basado en un conjunto de datos de "entrenamiento" y en el más sencillo se asume que los nucleótidos en diferentes posiciones son independientes e idénticamente distribuidos (iid). Si éste es el caso, las diferencias entre ADN codificante y no codificante podrían detectarse por las diferencias entre las frecuencias de los cuatro nucleótidos en los casos distintos.

## 2.1. Contrastes de Independencia

La precisión de los procedimientos que se llevan a cabo depende de la precisión de las suposiciones hechas. Podemos intuir que suponer que hay independencia entre los nucleótidos suele ser una simplificación excesiva, puesto que puede haber correlaciones, por ejemplo, entre los nucleótidos debido a su pertenencia a uno u otro codón (triplete de nucleótidos). Por ello, es importante, entre otras cosas, desarrollar un contraste de independencia sobre la secuencia de nucleótidos. Este contraste está basado en el análisis de una cadena de Markov. Al utilizar cadenas de Markov en el análisis de secuencias, el concepto del "tiempo $t$" se reemplaza por la "posición $a$ en la secuencia". Nuestras variables aleatorias o sucesos serán "los nucleótidos en posiciones dadas de la secuencia de ADN" y, por tanto, el espacio de estados estará compuesto por los cuatro nucleótidos: $E = \{A, C, G, T\}$, de tamaño $k = 4$. Desarrollamos entonces un test de independencia chi-cuadrado $\chi^2$ sobre la secuencia de nucleótidos que queremos analizar, que contrasta la hipótesis nula de que los

| | A | C | G | T | Total |
|---|---|---|---|---|---|
| A | $n_{11}$ | $n_{12}$ | $n_{13}$ | $n_{14}$ | $n_{1,}$ |
| C | $n_{21}$ | $n_{22}$ | $n_{23}$ | $n_{24}$ | $n_{2,}$ |
| G | $n_{31}$ | $n_{32}$ | $n_{33}$ | $n_{34}$ | $n_{3,}$ |
| T | $n_{41}$ | $n_{42}$ | $n_{43}$ | $n_{44}$ | $n_{4,}$ |
| Total | $n_{,1}$ | $n_{,2}$ | $n_{,3}$ | $n_{,4}$ | $n$ |

Cuadro 2.1: Tabla de contingencia

nucleótidos en posiciones distintas sean independientes, frente a la hipótesis alternativa de que un nucleótido en una posición dada dependa del nucleótido en la posición anterior. De esta forma, la distribución del contraste es una cadena de Markov en la cual todas las filas de la matriz de transición son idénticas. Por lo tanto, la hipótesis alternativa la pensamos como una cadena de Markov donde la probabilidad de un nucleótido en una posición dada, dependa del nucleótido en la posición anterior, es decir, una cadena de Markov de orden uno. Así, bajo la hipótesis nula de independencia entre nucleótidos, y dada la secuencia de ADN $X = ACGATTA$, por ejemplo, la probabilidad de tal secuencia bajo el modelo de independencia será $P(X) = p_A p_C p_G p_A p_T p_T p_A = p_A^3 p_C^1 p_G^1 p_T^2$ , donde $p_i$ indica la probabilidad del nucleótido $i$ en la secuencia, $i \in \{A, C, G, T\}$.

Este contraste es de interés para evaluar si el modelo de Markov bajo la hipótesis alternativa describe la realidad significativamente mejor que el modelo de independencia y, por tanto, podría aumentar la precisión de nuestros procedimientos de predicción. Además, si lo hace, podría considerarse incluso el modelo con un nivel de dependencia más complejo (una cadena de Markov con orden mayor). Supongamos que la longitud de la secuencia de ADN que queremos analizar (nuestra muestra) es n. El contraste estadístico de independencia es un contraste de asociación en una tabla $4 \times 4$ de doble entrada como la mostrada en el Cuadro 2.1, denominada *tabla de contingencia*.

Tenemos entonces un contraste de independencia con la hipótesis nula

$$H_0 : p_{ij} = p_i \times p_i, i, j \in E.$$

Los datos deben aparecer como se muestra en la tabla anterior, donde las filas representan el nucleótido en la posición $a$ y las columnas el nucleótido en la posición $a+1$. El elemento $(i, j)$ de la tabla representa la frecuencia observada $n_{ij}$, que indica el número de transiciones del estado o nucleótido $i$ al estado o nucleótido $j$ en una etapa, para $i, j \in E$; y $\Sigma_{i,j \in E} n_{ij} = n$. Así, cada uno de los elementos $n_{i,}$, y $n_{,j}$ representan la suma de la fila $i$ (número de transiciones desde el estado $i$) y la columna $j$ (número de transiciones que llegan al estado $j$), respectivamente, $i, j \in E$. La idea es realizar el contraste anterior comparando las frecuencias esperadas bajo la hipótesis nula, $T_{ij} = np_i p_j$, con las observadas, $O_{ij} = n_{ij}$. Si las cantidades $p_i$ y $p_j$ no son conocidas, han de ser estimadas a partir de las

frecuencias observadas de la siguiente forma

$$\widehat{p_i} = \frac{n_{i,}}{n}, \widehat{p_j} = \frac{n_{,j}}{n}, \tag{2.1}$$

y por lo tanto

$$T_{ij} = n\widehat{p_i}\widehat{p_j} = \frac{n_{i,}n_{,j}}{n}, \tag{2.2}$$

lo que hace perder $(k-1)+(k-1)$ con $k=4$, grados de libertad adicionales al estadístico de contraste

$$\chi^2_{(k-1)+(k-1)} = \sum_{i=1}^{k}\sum_{j=1}^{k}\frac{(n_{ij}-T_{ij})^2}{T_{ij}} \simeq \chi^2_{(k-1)+(k-1)} \tag{2.3}$$

que sigue, bajo la hipótesis nula, una distribución aproximada chi-cuadrado con $(k-1)+(k-1)$ grados de libertad. De esta forma, rechazaremos $H_0$ si

$$\chi^2_{(k-1)+(k-1)} > \chi^2_{(k-1)+(k-1),1-\alpha},$$

y el p-valor correspondiente será menor o igual que $\alpha$, para $\alpha = 0{,}05$. Por lo tanto, la hipótesis nula de independencia resulta ser la hipótesis nula de no asociación en la tabla. Pruebas de este tipo muestran que nucleótidos contiguos en posiciones de secuencias de ADN son a veces dependientes, y el modelo de una cadena de Markov de primer orden se ajusta a los datos reales significativamente mejor que el modelo de independencia, tanto en regiones de intrones como de exones. Modelos homogéneos de Markov de orden mayor y modelos no homogéneos a veces pueden ajustarse mejor a los datos.

## 2.2. Contraste de Bondad de Ajuste

Si el contraste de independencia no es aceptado, se aplica otro test de interés para saber si la distribución estacionaria de la cadena de Markov es uniforme. De esta forma, si fuese así, los cálculos llevados a cabo con los datos de entrenamiento para hallar los parámetros del modelo se facilitarían bastante. Aplicaremos entonces un contraste de bondad de ajuste a tal distribución. La hipótesis nula es que las distribuciones de probabilidad estacionaria y uniforme son idénticas, y la hipótesis alternativa que ambas distribuciones son diferentes. Podemos discutirlo en el caso de una cadena de Markov de primer orden, ya que para cadenas de órdenes superiores es análogo. Si denotamos por $\phi$ la distribución estacionaria y por $U$ a la función de distribución uniforme, el contraste de hipótesis planteado se puede escribir de la forma

$$\begin{cases} H_0 : \phi \cong U \\ H_a : \phi \ncong U \end{cases}$$

En el caso de primer orden, una condición necesaria y suficiente para que la distribución estacionaria sea uniforme es que las probabilidades de transición en cada columna de la matriz de transición sumen 1. Así, tenemos el siguiente teorema.

**Teorema.** *Si la matriz de transición $P$ de una cadena de Markov con $k$ estados es doblemente estocástica, entonces la distribución uniforme $U(i) = \frac{1}{k}$ para todo $i \in E$, es una distribución estacionaria.*

**Demostración.** *Observamos que*

$$\sum_{i \in E} U(i) p_{ij} = \frac{1}{k} \sum_{i \in E} p_{ij} = \frac{1}{k}, \forall j \in E \tag{2.4}$$

*de modo que la distribución uniforme satisface la condición $UP = U$ que define una distribución estacionaria (y además, por ser distribución de probabilidad, $\sum_{i \in E} U(i) = 1$). Vemos también, que si la distribución estacionaria es uniforme, necesariamente la matriz $P$ es doblemente estocástica, es decir, la suma por filas y columnas de $P$ es 1.*

En nuestro caso, la distribución uniforme es $U(i) = \frac{1}{4}$ para $i \in E$. Esto implica que, por ejemplo, los elementos en la cuarta fila de la matriz de transición están determinados por los elementos de las primeras tres filas. La probabilidad de cualquier secuencia observada de ADN puede calcularse bajo la hipótesis nula de que la distribución estacionaria es uniforme. Ya que bajo esta hipótesis los elementos de cada fila y cada columna de la matriz de transición deben sumar 1, hay 9 parámetros libres (había $4 \times 3$ cuando solo las filas sumaban 1, menos 3 que se restan cuando también las columnas lo hacen). La probabilidad de la secuencia observada puede maximizarse con respecto a estos parámetros. Bajo la hipótesis alternativa, la única restricción es que los elementos de cualquier fila de la matriz de transición deben sumar 1, y por lo tanto habrá 12 parámetros libres. Así, de la misma forma, la probabilidad de la secuencia observada puede ser maximizada respecto a estos parámetros. De estas dos probabilidades puede calcularse un estadístico de verosimilitud $-2log\lambda$, donde $\lambda$ es

$$\lambda = \frac{supL_0}{supL} = \frac{f_n(x, \widehat{\theta}_0)}{f_n(x, \widehat{\theta})} = \frac{\frac{1}{4}}{sup_E \phi}. \tag{2.5}$$

Según este contaste, bajo la hipótesis nula, este estadístico tiene una distribución asintótica chi-cuadrado con $m - r = 12 - 9 = 3$ grados de libertad. De esta forma, podemos aplicar un contraste de bondad de ajuste entre ambas distribuciones y comprobar si la distribución estacionaria se identifica significativamente con la distribución uniforme, lo que facilita los cálculos a la hora de modelizar secuencias de ADN.

## 2.3. Modelado de "Señales" en el ADN

En el contexto de la genómica, la anotación o puntuación es el proceso de marcado de los genes y otras características biológicas de la secuencia de ADN. El primer sistema software de anotación de genomas fue diseñado en 1995 por Owen White, éste construyó un software para localizar los genes, el ARN de transferencia, y otras características; así como para realizar las primeras atribuciones de función a esos genes.

Se conoce que los genes contienen "señales" en el ADN, que son secuencias de ADN con un propósito específico: indicar, por ejemplo, el comienzo y final de la región transcrita, los límites de los exones e intrones, así como otras características. La maquinaria de la célula utiliza estas señales para reconocer el gen, para editarlo correctamente, y para traducirlo apropiadamente en proteína. Si la naturaleza fuese bondadosa, cada señal consistirá en una única secuencia de ADN que no aparecería en ningún otro lugar del ADN excepto donde sirviese para su propósito específico. En la realidad, hay muchas secuencias de ADN que realizan la misma función de señal; llamamos a éstas "miembros" de una señal. Además, los miembros de las señales también aparecen aleatoriamente en el ADN no funcional, haciendo difícil clasificar las señales funcionales de las no funcionales. En la práctica, no todos los miembros de una señal son conocidos. Nuestro objetivo es utilizar los miembros conocidos para evaluar la probabilidad de que una nueva secuencia no caracterizada de ADN sea también un miembro de la señal. Supondremos que los diferentes miembros surgen de antepasados comunes mediante procesos estocásticos, por lo tanto, es razonable construir un modelo estadístico de los datos. Algunas señales requieren solamente modelos simples, mientras que otras necesitan modelos más complejos. Cuando el modelo requerido es complejo y los datos son limitados, debemos tener cuidado eligiendo suposiciones simplificadoras en el modelo para utilizar los datos de la manera más eficiente posible. Asumimos que todos los miembros de la señal de interés tienen la misma longitud, que denotaremos por $n$. Esta suposición no es demasiado restrictiva, ya que los miembros de muchas señales tienen la misma longitud, y para los que no la tengan, podemos capturar porciones de ellos, que es normalmente suficiente. Para modelar las propiedades de cualquier señal debemos tener un conjunto de datos de entrenamiento, esto es, una gran cantidad de datos en la cual los miembros de las señales sean conocidos. Consideraremos ahora algunos de los modelos de señales básicos que se utilizan en Bioinformática.

### 2.3.1. Matrices de Peso. Independencia

El contraste de hipótesis sobre si una secuencia no caracterizada de ADN es miembro de una señal dada, es más fácil de llevar a cabo cuando el nucleótido en cualquier posición de la señal es independiente de los nucleótidos en cualquier otra posición de esa señal. Es, por lo tanto, necesario realizar un contraste de independencia sobre si el nucleótido en la posición $a$ en una señal es independiente del nucleótido en la posición $b$, sin necesidad de que $a$ y $b$ sean contiguos. Esto puede hacerse de varias formas. Es natural generalizar el contraste de independencia descrito en el cuadro 2.1, que comprueba la independencia en posiciones contiguas. En esta generalización, "posición $a$" es reemplazada por "posición $a$ en la señal" y "posición $a + 1$" se reemplaza por "posición $b$ en la señal". Así, $n_{ij}$ es interpretado como el número de veces que, en los datos, el nucleótido $i$ ocurre en la posición $a$ de la señal y el nucleótido $j$ ocurre en la posición $b$ de la señal. Este contraste se utiliza luego para todos los pares $a$ y $b$; con $i, j \in E$ y $a, b = 1, ..., n$. Por tanto, nos planteamos un contraste de independencia chi-cuadrado entre las posiciones de los nucleótidos en la señal, es decir la hipótesis nula es

$H_0$ : el nucleótido en la posición $a$ es independiente del de la posición $b$, $\forall a, b = 1, ..., n$.

El contraste se desarrolla análogamente al test de independencia para cadenas de Markov de orden uno, expuesto en el presente capítulo (donde las posiciones en la secuencia sí eran contiguas).
Supongamos que, como resultado de este contraste, puede asumirse independencia. Se construye entonces una matriz $4 \times n$, donde cada fila corresponde a uno de los 4 nucleótidos y cada columna a las posibles posiciones de la señal de longitud $n$. Su posición o entrada $(i, a)$ es la proporción de casos en los datos que el nucleótido $i$ ocurre en la posición $a$ de la señal. Nos referimos a ésta como una *matriz de peso* y la denotaremos por $M$.

Los elementos de la columna $a$ de la matriz dan (de forma aproximada) las probabilidades para los cuatro nucleótidos (A, G, C y T, respectivamente) en la posición $a$ de la señal, $a = 1, ..., n$. La matriz $M$ define la probabilidad $P(X \mid M)$ para cualquier secuencia (señal) $X$ de longitud $n$. Las matrices de peso se utilizan como una componente de un algoritmo para búsqueda de genes. En general, al modelar un gen, se necesitan matrices de peso para modelar ciertas "regiones" de éste [1].

## 2.4. Búsqueda de Repeticiones en Secuencias de ADN

Describimos ahora las posibles repeticiones de secuencias de ADN presentes en un genoma eucariota y su clasificación de acuerdo con su longitud y forma. También abordaremos el problema computacional para detectarlas y los enfoques utilizados.

En un genoma eucariota, los nucleótidos no se encuentran en igual cantidad, A y T se encuentran en un 60 % (30 % cada uno), y C, G en un 40 % (20 % cada uno). Pero un genoma no sólo varía en la cantidad de nucleótidos que contiene, también presenta una distribución no uniforme de los nucleótidos a lo largo de la secuencia genómica. Incluso existen algunos segmentos con una alta concentración de un par de nucleótidos (A, T) o (C, G) denominados *isocoros*. Estos isocoros son objeto de estudio ya que presentan una alta concentración de secuencias codificantes. Oliver et al. [2] propusieron un método para detectar segmentos (isocoros o dominios que presenten una composición similar) denominado segmentación entrópica. Además de los isocoros, es interesante estudiar repeticiones de secuencias en un genoma ya que, principalmente los organismos eucariotas, contienen un alto número de secuencias repetidas (de longitud, composición y frecuencia variables). Estas repeticiones se pueden encontrar en los genomas de dos formas: "en tándem", dos o más copias consecutivas de un patrón, o dispersas aleatoriamente a lo largo del genoma.

### 2.4.1. Repeticiones en Tándem

Estas repeticiones no están definidas claramente todavía, no obstante se pueden clasificar como en el Cuadro 2.2.

| Clase | Long-Nucleótidos | Frec-miles |
|---|---|---|
| Satélites | Superior a 100 | 1000 |
| Minisatélites | 9-100 | 100 |
| Macrosatélites | 1-8 | 10-1000 |

Cuadro 2.2: Repeticiones Tándem

### 2.4.2. Repeticiones Dispersas

Las repeticiones dispersas o intercaladas se clasifican de acuerdo con su longitud en cortas, denominadas SINE (*short interspersed repeat*) y en largas, denominadas LINE (*long interspersed*). Estas repeticiones pueden estar formadas por la combinación de nucleótidos en diferente orden y cantidad. Las repeticiones SINE son fragmentos de ADN cortos repetidos millones de veces y dispersos por todo el genoma. En el genoma humano, se estima que tienen una longitud de 100 a 300 pares de bases y se repiten 1,5 millones de veces (13 % del genoma humano). Las repeticiones LINE son fragmentos de ADN de gran tamaño repetidos miles de veces y dispersos por todo el genoma. En el genoma humano se estima que tienen una longitud de 6,000 a 8,000 pares de bases y se repiten 85 mil veces (21 % del genoma humano).

### 2.4.3. Estudio de Repeticiones

La importancia del estudio de patrones repetidos es evidente desde el proceso de secuenciación. Durante este proceso, se detectó la presencia de bloques de ADN repetidos, lo cual dificultó el ensamblado del genoma. El proceso de secuenciación automático divide el genoma en miles de fragmentos que luego se amplifican y por último se secuencian ensamblándolos por superposición de sus extremos. Pero antes de unir dos fragmentos, el algoritmo debe determinar si los fragmentos en cuestión son producto de la "clonación" (son el mismo fragmento) o se trata de un fragmento repetido en el genoma. Es el procedimiento conocido como "shotgun". En el genoma del hombre, al igual que en el de muchos organimos, se presentan secuencias de nucleótidos repetidos (patrones o secuencias motif). Estos patrones generalmente tienen alguna relación funcional o estructural, ya sea en la secuencia de ADN o en la proteína, y se cree que por esta razón la evolución los ha preservado. Los patrones repetidos pueden ser útiles en tareas como: validación de métodos de clasificación de proteínas, asociación directa con alguna función y/o estructura de una proteína, predicción o identificación de dominios funcionales, o predicción de la estructura tridimensional. Específicamente, para el genoma humano, se estima que el 10 % está formado por secuencias repetidas en tándem. Estas repeticiones son probablemente resultado de una replicación inexacta del ADN que con el tiempo se fijaron en el genoma. El estudio de repeticiones en tándem tiene aplicación directa en medicina debido a que algunos patrones repetidos han sido asociados a enfermedades.

### 2.4.4. El problema de la Detección de los Patrones

La estructura primaria de un genoma se representa computacionalmente como un conjunto de archivos de texto (un archivo para cada cromosoma). Estos archivos contienen la secuencia de nucleótidos. Cada nucleótido se representa con una letra mayúscula *A, C, G* y *T*, y si aún no se ha determinado el nucleótido en alguna posición del genoma se utiliza la letra *N*. El número de repeticiones a buscar se incrementa exponencialmente con la longitud del patrón de interés. En el Cuadro 2.3 se ilustra el número total de combinaciones posibles de un patrón de longitud 1 hasta 20.

| Longitud | Cantidad | Longitud | Cantidad |
|---|---|---|---|
| 1 | 4 | 11 | 4194304 |
| 2 | 16 | 12 | 16777216 |
| 3 | 64 | 13 | 67108864 |
| 4 | 256 | 14 | 268435456 |
| 5 | 1024 | 15 | 1073741824 |
| 6 | 4096 | 16 | 4294967296 |
| 7 | 16384 | 17 | 17179869184 |
| 8 | 65536 | 18 | 68719476736 |
| 9 | 262144 | 19 | 27487790644 |
| 10 | 1048576 | 20 | 1099511627776 |

Cuadro 2.3: Número total de combinaciones posibles de un patrón de longitud 1 hasta 20

Para un patrón de longitud $n$, existen $4^n$ combinaciones posibles, razón por la cual este problema presenta un reto computacional de interés. La identificación de patrones repetidos en un genoma puede estar dirigida a la búsqueda de repeticiones dispersas o en tándem, con coincidencia exacta del patrón o con presencia de "gaps" (inserciones o deleciones) y/o mutaciones. El problema se puede plantear así:

$$descubrir(X, w, f, g, l_t)$$

donde la función "descubrir" debe encontrar una secuencia $X$ de longitud $|X| = n$, el patrón $w$ de longitud $|w|$ que tenga una frecuencia de aparición no mayor o igual a $f$, para descartar secuencias con un bajo número de repeticiones; y presente un número de diferencias, incluyendo gaps, menor o igual a $g$. Si $g$ es igual a cero, el patrón se buscará con una coincidencia exacta. El parámetro $l_t$ indica la longitud de la repetición en tándem. Si $l_t$ es igual a uno, la búsqueda se realizará para patrones repetidos de forma dispersa.

# Capítulo 3

# Modelos Ocultos de Markov

Un modelo oculto de Markov (HMM por sus siglas en inglés) es un modelo estadístico que puede ser usado para describir la evolución de eventos observados que dependen de factores internos, los cuales no son observados. Llamaremos al evento observado "símbolo" y al factor oculto subyacente a la observación "estado". Un HMM consiste de dos procesos estocásticos: un proceso estocástico de estados ocultos y un proceso estocástico de eventos observados. Los estados ocultos forman una cadena de Markov y la distribución de probabilidad de los símbolos observados depende de los estados ocultos subyacentes. Por esta razón, un modelo oculto de Markov es llamado proceso estocástico bivariado. La mayoría de los escritos sobre los modelos de Markov ocultos pertenecen a la literatura del reconocimiento ortográfico, donde fueron aplicados por primera vez a principios de 1970 [3].

## 3.1. Definición

Podemos decir que un Modelo Oculto de Markov (HMM, por sus siglas en inglés), es un proceso estocástico bivariado, ya que consta de un proceso estocástico de estados ocultos, el cual forma una cadena de Markov, y un proceso estocástico de eventos observados.

### 3.1.1. Definición formal

Un Modelo Oculto de Markov discreto se define formalmente como una 5-tupla $(S, V, \Pi, A, B)$ donde

- $S$ es el conjunto finito de estados ocultos del HMM, $S = \{s_1, s_2, ..., s_N\}$, $N$ es la cantidad de estados en el modelo. El estado actual en el instante de tiempo $t$ se denota como $X_t$.

- $V$ es el conjunto de valores o símbolos diferentes que se pueden observar en cada uno de los estados, puede considerarse también un alfabeto finito. Cada uno de los símbolos que un estado puede emitir se denota como $\{v_1, v_2, ..., v_M\}$, donde $M$ es el número

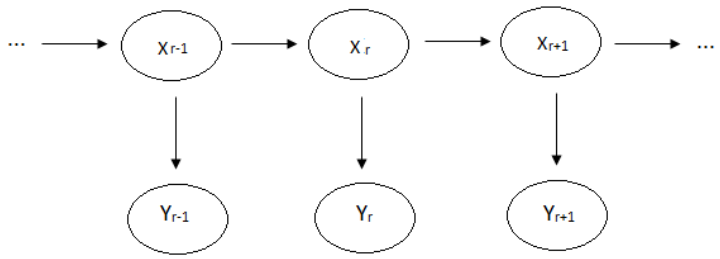


Figura 3.1: Arquitectura general de un modelo oculto de Markov

de símbolos del alfabeto y cada $v_k$ se refiere a un símbolo diferente. Una secuencia de observaciones se denota como un VECTOR $\mathbf{Y} = (Y_1, Y_2, ..., Y_T)$ donde cada observación $Y_i$, es un elemento de $V$, y $T$ es la cantidad de observaciones en la secuencia.

- $\Pi = \{\pi_1, \pi_2, ..., \pi_N\}$ es la distribución de *probabilidad inicial* de los estados. De esta forma, $\pi_i = P(X_1 = s_i)$ constituye la probabilidad de que el sistema inicie en el estado $s_i$.
- $A = \{p_{ij}\}$ es la *matriz de las probabilidades de transición* entre estados, donde $p_{ij} = P(X_t = s_j \mid X_{t-1} = s_i)$ constituye la probabilidad de que el sistema se encuentre en el estado $s_i$ en el instante $t-1$ y pase al estado $s_j$ en el instante $t$.
- $B = \{e_i(k)\}$ es el conjunto de las *probabilidades de emisión*, es decir, $e_i(k) = P(Y_t = v_k \mid X_t = s_i)$ constituye la probabilidad de que el sistema, estando en el instante $s_i$, genere la observación $v_k$.

La notación más compacta para un HMM es usando sus tres medidas de probabilidad $\lambda = (A, B, \Pi)$

Veamos ahora dos ejemplos de cómo aplicar un HMM.

**Ejemplo 1. Un HMM con dos estados:** Imaginemos que en un casino utilizan un dado justo la mayoría de las veces, pero otras utilizan un dado cargado. Así que tenemos un par de dados, uno justo y otro cargado (no caemos en todas las caras con igual probabilidad). De esta forma, nuestros estados serán: {"dado libre", "dado cargado"}. Un modelo de Markov decide cuál de los dos dados juega, y dependiendo del estado del modelo, se aplicarán las probabilidades de emisión para el dado cargado o para el normal. Así, generamos una secuencia de símbolos que es el resultado de estar en dos estados diferentes. Especificaremos nuestra probabilidad de transición de modo que en cualquiera de los estados haya un 90 % de posibilidades de quedarse en ese estado, y un 10 % de cambiar de estado (el casino cambia un dado libre por uno cargado), como se muestra en la Figura 3.2.

Para nuestro dado no cargado o libre, la probabilidad de obtener un número entre 1 y 6 (ambos incluidos) es la misma, y está dada por

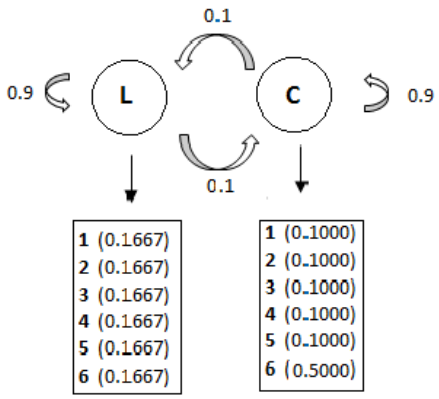


Figura 3.2: Modelo de Markov oculto asociado al ejemplo. Las transiciones entre el dado cargado y el dado libre se modelizan como una cadena de Markov.

$$0.1667\ 0.1667\ 0.1667\ 0.1667\ 0.1667\ 0.1667$$

donde cada columna describe la probabilidad para cada uno de los seis números.
Para el dado cargado, las probabilidades de emisión son

$$0.100\ 0.1000\ 0.1000\ 0.1000\ 0.1000\ 0.5000$$

Aquí las probabilidades de emitir un número entre 1 y 5 son iguales, pero es mucho mayor la probabilidad de obtener un 6 (50 %).
La secuencia observada por tal modelo de Markov oculto podría ser como la siguiente

$$\mathbf{Y}=45536531633635551333626651321416366516 66$$

Si conocemos las propiedades de los dados y de la cadena de Markov subyacente u oculta, ¿podemos encontrar la secuencia de estados más probable de estados ocultos detrás de $Y$? En otras palabras, ¿podemos averiguar qué dado ha sido utilizado por el casino en cada instante o posición de la secuencia de resultados? Más adelante, con el algoritmo de Viterbi, veremos cómo responder a estas preguntas, aquí simplemente mostramos la secuencia oculta que genera nuestra secuencia visible

$$\textit{Oculto:}\mathbf{X}=11111111111111111111222211111112222222 22$$
$$\textit{Visible:}\mathbf{Y}=45536531633635551333626651321416366516 66$$

Observemos que el símbolo 6 ocurre con una frecuencia mayor cuando el estado en la secuencia oculta es 2, correspondiente al dado cargado, pero esta dependencia es simplemente probabilística. En las aplicaciones biológicas, aplicaremos nuestro modelo de Markov oculto en un conjunto de datos donde los estados ocultos son conocidos, incluso cuando no sepamos exactamente cuales son las probabilidades de transición y emisión. Esto permite

calcular las matrices de transición y emisión de nuestro modelo basadas en los datos y, por lo tanto, una mejor inferencia en los estados ocultos de nuevos datos.

**Ejemplo 2:** Queremos transmitir un mensaje codificado como secuencia de bits $b_0, ..., b_m$, donde $b_i \in \{0, 1\}$ son los bits y $m$ es la longitud del mensaje. Deseamos transmitir el mensaje mediante un canal que sabemos que afectará al mensaje puesto que introduce errores aleatoriamente. Consideraremos canales *discretos*, esto es, se supone que las entradas y las salidas del canal pertenecen a alfabetos finitos: $i_1, ..., i_q$ para las entradas y $o_1, ..., o_l$ para las salidas. Ya que consideraremos canales binarios, entonces las entradas y las salidas del canal de transmisión son bits, así $q = l = 2$ y $i_1, i_2 = o_1, o_2 = \{0, 1\}$. Un canal de transmisión se dice "sin memoria" si

$$P(Y_0 = y_0, Y_1 = y_1, ..., Y_n = y_n \mid S_0 = s_0, S_1 = s_1, ..., S_n = s_n) = \prod_{i=0}^{n} P(Y_i = y_i | S_i = s_i).$$

donde $Y_k$ representa la salida del canal y $S_k$ la entrada.

En palabras, las salidas del canal son condicionalmente independientes de la secuencia de entrada. Las probabilidades de transición del canal discreto sin memoria están definidas por el kernel de transición $R : \{i_1, ..., i_q\} \times \{o_1, ..., o_l\} \to [0, 1]$, donde para $m = 1, ..., q$ y $j = 1, ..., l$,

$$R(i_m, o_j) = P(Y_0 = o_j \mid S_0 = i_i) \tag{3.1}$$

El ejemplo más clásico de un canal discreto sin memoria es el *canal simétrico binario* (BSC) con entrada y salida binaria, para las cuales $R(0, 1) = R(1, 0) = \varepsilon$ con $\varepsilon \in [0, 1]$. En palabras, cada vez que un bit $S_k = 0$ o $S_k = 1$ es enviado a través del BSC, la salida es también un bit $Y_k = \{0, 1\}$, el cual difiere del bit de entrada con probabilidad $\varepsilon$; esto es, el error de probabilidad es $P(Y_k \neq O_k) = \varepsilon$. La salida de un canal simétrico binario puede ser modelado como una versión ruidosa de la secuencia de entrada, $Y_k = S_k \oplus V_k$, donde $\oplus$ es la suma módulo 2 (es decir, la suma asigna 0 a una suma par y el valor de 1 a una suma impar) y $\{V_k\}_{k \geq 0}$ es una secuencia de bits independiente e idénticamente distribuida, independiente de la secuencia de entrada $\{X_k\}_{k \geq 0}$ y con $P(V_k = 0) = 1 - \varepsilon$. Si queremos transmitir un mensaje $S_0 = b_0, ..., S_m = b_m$ mediante un BSC sin codificar, la probabilidad de obtener un error será

$$\begin{aligned} P(Y_0 \neq b_0, ..., Y_m \neq b_m \mid S_0 = b_0, ..., S_m = b_m) = \\ 1 - P(Y_0 = b_0, ..., Y_m = b_m \mid S_0 = b_0, ..., S_m = b_m) = \\ 1 - (1 - \varepsilon)^m. \end{aligned}$$

Por lo tanto, cuando m crece, con probabilidad de casi 1, al menos un bit del mensaje sería recibido incorrectamente. Consideremos un codificador convolucional de razón 1/2 con longitud de memoria 2. La razón 1/2 significa que el mensaje de longitud $m$ será transformado en un mensaje de longitud $2m$, esto es, enviamos $2m$ bits a través del canal con el fin de introducir algún tipo de redundancia para incrementar nuestra oportunidad de obtener un mensaje libre de errores. El principio de este codificador convolucional es mostrado en la Figura 3.3.

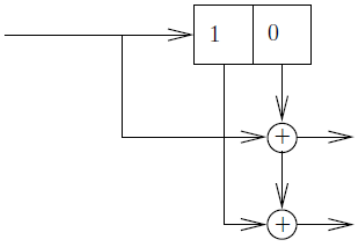


Figura 3.3: Código convolucional de razón $\frac{1}{2}$ con longitud de memoria 2.

Debido a que la longitud de memoria es 2, hay 4 estados diferentes y el comportamiento de este codificador convolucional puede ser modelado como una máquina de 4 estados, donde el alfabeto de estados es $E = \{(0,0),(0,1),(1,0),(1,1)\}$. Denotamos por $X_k$ el valor del estado en el tiempo $k$, $X_k = (X_{k,1}, X_{k,2}) \in E$ . A la llegada del bit $B_{k+1}$, el estado se transforma a

$$X_{k+1} = (X_{k+1,1}, X_{k+1,2}) = (B_{k+1}, X_{k,1})$$

Si la secuencia $\{B_k\}_{k\geqslant 0}$ de los bits de entrada es i.i.d. con probabilidad $P(B_k = 1) = p$, entonces $\{X_k\}_{k\geqslant 0}$ es una cadena de Markov con probabilidades de transición

$$\begin{aligned}
P[X_{k+1} = (1,1) \mid X_k = (1,0)] &= P[X_{k+1} = (1,1) \mid X_k = (1,1)] = p,\\
P[X_{k+1} = (1,0) \mid X_k = (0,1)] &= P[X_{k+1} = (1,0) \mid X_k = (0,0)] = p,\\
P[X_{k+1} = (0,1) \mid X_k = (1,0)] &= P[X_{k+1} = (0,1) \mid X_k = (1,1)] = 1-p,\\
P[X_{k+1} = (0,0) \mid X_k = (0,1)] &= P[X_{k+1} = (0,0) \mid X_k = (0,0)] = 1-p,
\end{aligned}$$

las otras probabilidades de transición son cero. Para cada bit de entrada, el codificador convolucional genera dos salidas de acuerdo a

$$S_k = (S_{k,1}, S_{k,2}) = (B_k \oplus X_{k,2}, B_k \oplus X_{k,2} \oplus X_{k,1})$$

Un codificador convolucional puede ser representado como un diagrama de transición de estado del tipo de la Figura 3.4. Los nodos son los estados y las ramas representan las transiciones que no tienen probabilidad cero. Si indexamos los estados con el índice de tiempo $k$ y el índice de estado $m$, obtenemos el diagrama de Trellis de la Figura 3.5. El diagrama de Trellis muestra la progresión del tiempo de la secuencia de estados. Para toda secuencia de estados, existe un único camino a través del diagrama de Trellis y viceversa.

## 3.2. Los Tres Problemas Básicos en los Modelos Ocultos de Markov

Dada la forma del modelo oculto de Markov en la sección anterior, existen tres problemas básicos que deben resolverse para que el modelo pueda ser aplicado a sistemas reales.

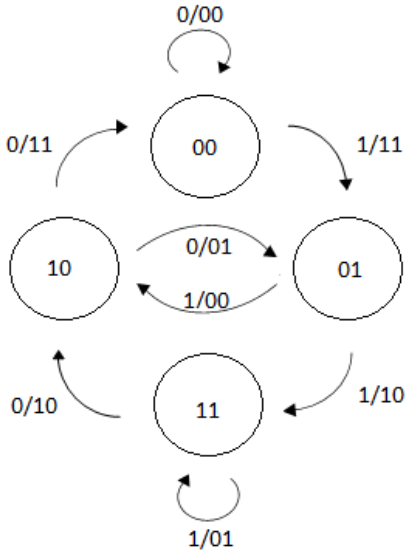


Figura 3.4: Diagrama de estados código convolucional de razón $\frac{1}{2}$ con longitud de memoria 2.

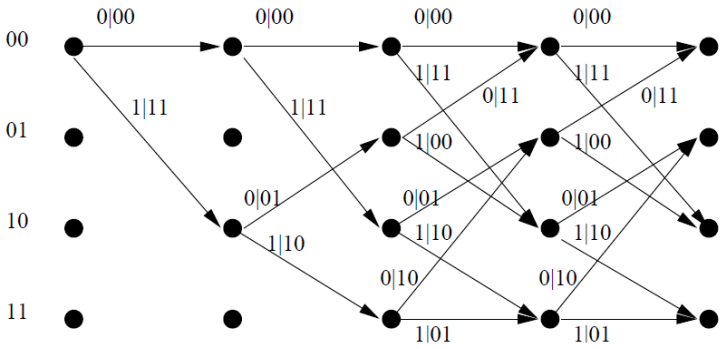


Figura 3.5: Diagrama de Trellis del código convolucional de razón $\frac{1}{2}$ con longitud de memoria 2.

Estos problemas son los siguientes:

- Dada una secuencia de observaciones $\mathbf{Y} = (Y_1, Y_2, ..., Y_T)$ y un modelo $\lambda = (A, B, \Pi)$. ¿Cómo podemos calcular de forma eficiente $P_\lambda(Y_1 = y_1, ..., Y_T = y_T)$?, o dicho de otro modo, ¿Cómo calcular la probabilidad de obtener dicha secuencia de observaciones dado un modelo fijo?

- Dada una secuencia de observaciones $\mathbf{Y} = (Y_1, Y_2, ..., Y_T)$ y un modelo $\lambda = (A, B, \Pi)$. ¿Cómo encontrar la sucesión de estados ocultos más factible que pueda haber generado dicha secuencia de observaciones? (La que mejor explica la secuencia de observaciones dada).

- Dada una secuencia de observaciones $\mathbf{Y} = (Y_1, Y_2, ..., Y_T)$ ¿Cómo estimar los parámetros del modelo $\lambda = (A, B, \Pi)$ para maximizar $P_\lambda(Y_1 = y_1, ..., Y_T = y_T)$? (El modelo que mejor explica la secuencia de observaciones dada), o en otras palabras, *entrenar* los parámetros del HMM dada una secuencia de observaciones, por *entrenar* entenderemos el proponer un modelo a priori e ir mejorándolo al maximizar $P_\lambda(Y_1 = y_1, ..., Y_T = y_T)$.

**Problema 1: Probabilidad de una observación**
El primer problema consiste en la evaluación de una secuencia de observaciones dado el modelo. La resolución de este problema provee la probabilidad de que la secuencia fuera generada por ese modelo. Un ejemplo práctico puede ser presuponer la presencia de varios HMM's pugnando entre sí. La evaluación de la secuencia en cada uno de ellos, dará la oportunidad de elegir el modelo que mejor se ajusta con las observaciones.

**Problema 2: Secuencia de estados más probable**
El segundo problema es encontrar lo la cadena oculta del modelo, es decir , descubrir el camino que siguió la cadena de Markov. Este camino oculto es de utilidad para la clasificación de las observaciones.

**Problema 3: Estimación de los parámetros del modelo**
El tercero de los problemas, es el problema crítico de los HMM puesto que permite adecuar en forma óptima los parámetros de un modelo a una secuencia de observaciones. También resulta el más complejo de resolver. Notemos que la solución a este problema permite la efectiva aplicación de los HMM a una cantidad de casos reales, a los que sería inviable su aplicación, pues determinar a priori las distribuciones correspondientes no es una opción en la mayoría de los casos. En lugar de eso, se dispone de una *secuencia de entrenamiento* con el fin de *enseñar* al HMM cuáles son las distribuciones que mejor se adaptan.

## 3.3. Solución a los Tres Problemas Básicos

### 3.3.1. Solución al problema 1: Probabilidad de una Observación

Se quiere calcular de manera eficiente la probabilidad de una secuencia de observaciones $\mathbf{Y} = (Y_1, Y_2, ..., Y_T)$, dado un modelo $\lambda = (A, B, \Pi)$, es decir $P_\lambda(\mathbf{Y})$.
Recordando que cada $Y_t$ depende solamente de $X_t$, (la observación en el tiempo $t$ depende únicamente del estado en el tiempo $t$), entonces para producir la secuencia $\mathbf{Y}$ se debe tener una secuencia de estados $\mathbf{X} = (X_1, X_2, ..., X_T)$. Luego esto permite inferir un sencillo método de resolución al problema en cuestión, simplemete se podría *calcular la probabilidad de obtener* $\boldsymbol{Y}$*, con cada una de las posibles secuencias de estados de longitud* $T$ *que son posibles con el modelo, y finalmente sumar dichas probabilidades.* Notar que si $N$ es el número de estados del modelo, existen $N^T$ posibles secuencias de estados de longitud $T$.
Como la probabilidad de que ocurra la secuencia de estados $\mathbf{X} = (X_1, X_2, ..., X_T)$ en el modelo $\lambda = (A, B, \Pi)$ está dada por:

$$P_\lambda(\mathbf{X}) = \pi_{x_1} \prod_{t=1}^{T-1} p_{x_t x_{t+1}}. \tag{3.2}$$

La probabilidad de $\mathbf{Y}$, dada la secuencia $\mathbf{X}$ y el modelo $\lambda = (A, B, \Pi)$ es:

$$P_\lambda(\mathbf{Y} \mid \mathbf{X}) = \prod_{t=1}^{T} P_\lambda(Y_t = y_t \mid X_t = x_t) = e_{x_1}(y_1) \cdot e_{x_2}(y_2) \cdots e_{x_T}(y_T). \tag{3.3}$$

donde se asume independencia entre observaciones. La probabilidad conjunta de $\mathbf{Y}$ y $\mathbf{X}$ (la probabilidad de obtener la secuencia de observaciones $\mathbf{Y}$ simultáneamente con la secuencia particular de estados $\mathbf{X}$) dado el modelo, es el producto de (3.2) y (3.3) (usando la definición de probabilidad condicional en conjunto con la regla de la multiplicación), es decir:

$$P_\lambda(\mathbf{Y}, \mathbf{X}) = P_\lambda(\mathbf{Y} \mid \mathbf{X}) P_\lambda(\mathbf{X}). \tag{3.4}$$

Y como se comentó al principio, para obtener $P_\lambda(\mathbf{Y})$ se debe calcular (3.4) para cada secuencia posible de estados de tamaño $T$ del modelo y luego calcular la suma de las mismas, por lo tanto:

$$P_\lambda(\mathbf{Y}) = \sum P_\lambda(\mathbf{Y} \mid \mathbf{X}) P_\lambda(\mathbf{X}). \tag{3.5}$$

Equivalentemente:

$$P_\lambda(\mathbf{Y}) = \sum \left( \pi_{x_1} e_{x_1}(y_1) \prod_{t=1}^{T-1} p_{x_t x_{t+1}} e_{x_{t+1}}(y_{t+1}) \right). \tag{3.6}$$

Se puede interpretar la ecuación (3.6) de la siguiente forma. Al inicio (tiempo t=1), el proceso se encuentra en el estado $x_1$ con probabilidad $\pi_{x_1}$ y es generado el símbolo $y_1$ con probabilidad $e_{x_1}(y_1)$ , en t=2, se produce una transición del estado $x_1$ al estado $x_2$ con

probabilidad $p_{x_1x_2}$, y se genera el símbolo $y_2$ con probabilidad $e_{x_2}(y_2)$. El proceso continua hasta que se llega al tiempo $t = T$, con una transición del estado $x_{T-1}$ al estado $x_T$ con probabilidad $e_{x_{T-1}x_T}$ y generando el símbolo $y_T$ con probabilidad $e_{x_T}(y_T)$. En la figura se puede ver una representación gráfica de lo anterior.

Como ya se mencionó antes, si la cantidad de estados del modelo es $N$, la cantidad de secuencias de estados posibles de longitud $T$ es $N^T$. Analizando la ecuación (3.6), se puede ver que para cada una de las $N^T$ secuencias de la sumatoria se deben realizar $(2T - 1)$ multiplicaciones y $N^T - 1$ sumas. Por lo tanto, esta forma de calcular $P_\lambda(\mathbf{Y})$ es del orden de $(2TN^T)$ lo que la convierte en impracticable.
Para solucionar este problema debe hacerse uso de técnicas de programación dinámica, con el objetivo de recordar soluciones parciales, en vez de recalcularlas. A continuación se mostrarán un par de técnicas para ese fin, denominados *proceso de avance* (forward) y *proceso de retroceso* (backward).

**Proceso de avance**

Considérese la siguiente variable $\alpha_t(i)$, definida de la siguiente manera:

$$\alpha_t(i) = P_\lambda(Y_1 = y_1, Y_2 = y_2, ..., Y_t = y_t, X_t = i), \tag{3.7}$$

la cual representa la probabilidad conjunta de obtener la secuencia de observaciones parcial $(y_1, y_2, ..., y_t)$ hasta el tiempo $t$, y que el proceso se encuentre en el estado $i$ en el tiempo $t$, dado el modelo $\lambda = (A, B, \Pi)$. Los resultados de $\alpha_t(i)$, para los estados del modelo en los diferentes momentos de tiempo, se pueden calcular de forma iterativa y luego pueden ser usados para determinar $P_\lambda(\mathbf{Y})$. El proceso es descrito en el siguiente algoritmo

1. Inicialización:
$$\alpha_1(i) = \pi_i e_i(y_1), \quad 1 \leq i \leq N. \tag{3.8}$$

2. Inducción:
$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) p_{ij}\right] e_j(y_{t+1}), \quad t = 1, 2, ..., T-1; \;\; 1 \leq j \leq N. \tag{3.9}$$

3. Terminación:
$$P_\lambda(Y_1 = y_1, ..., Y_T = y_T) = \sum_{i=1}^{N} \alpha_T(i). \tag{3.10}$$

El primer paso (3.8), inicializa todas las potenciales probabilidades de inicio, es decir, una para cada estado ($N$ en total), como las probabilidades conjuntas de que inicie en el estado $i$ y dicho estado genere la observación $y_1$.
El paso de inducción (3.9) es el más complejo y es el núcleo del algoritmo de avance. Para

entenderlo, primero nótese que $\alpha_t(i)$ representa la probabilidad conjunta de que el proceso genere la sucesión de observaciones $(y_1, y_2, ..., y_t)$, estando en el estado $i$ en el tiempo $t$, luego $\alpha_t(i)p_{ij}$ representa la probabilidad de que en el tiempo $t+1$ exista una transición desde el estado $i$ hasta el estado $j$, simultáneamente a la probabilidad conjunta anterior para el tiempo $t$. Si se suman los productos para cada uno de los $N$ estados posibles en el instante $t$, y se multiplica esa suma por $e_j(y_{t+1})$, (que es la probabilidad de que el símbolo $y_{t+1}$ sea emitido por el $j$), se obtiene $\alpha_{t+1}(j)$, que será la probabilidad conjunta de obtener la secuencia parcial de observaciones $(y_1, y_2, ..., y_{t+1})$ y de encontrarse en el estado $j$ en el instante de tiempo $t+1$. Se realiza el cálculo anterior para cada uno de los $N$ estados y para los instantes de tiempo $t = 1, 2, ..., T-1$. En la figura se puede apreciar una representación del paso de inducción. El paso de terminación 3.10 obtiene $P_\lambda(\mathbf{Y})$ sumando el valor de las $N$ variables $\alpha_T(i)$. Nótese que por definición,

$$\alpha_T(i) = P_\lambda(Y_1 = y_1, Y_2 = y_2, ..., Y_{T-1} = y_{T-1}, Y_T = y_T, X_T = i). \tag{3.11}$$

Es decir, la probabilidad conjunta de obtener la secuencia de observaciones $(y_1, y_2, ..., y_T)$ y que en el tiempo $T$ el proceso se encuentre en el estado $i$, si se hace extensivo eso a todos los $N$ estados, y se calcula la suma, se tendrá precisamente $P_\lambda(\mathbf{Y})$, obteniendo la ecuación (3.10) el mismo resultado que la ecuación (3.6).

La Figura 3.6 muestra los estados y probabilidades necesarias para el cálculo de $\alpha_4(3)$. Donde $\alpha_4(3) = \left(\sum_{i=1}^{5} \alpha_3(i)p_{i3}\right) e_3(y_4)$.

**Ejemplo aplicación del algoritmo de avance**

Consideremos el siguiente HMM simple. El modelo está compuesto por dos estados: H (alto contenido de GC) y L (bajo contenido de GC). Podríamos suponer que el estado H caracteriza el ADN codificante, mientras que el estado L caracteriza el ADN no codificante. Representamos gráficamente el modelo como en la figura 3.7.

Ahora bien, considere la secuencia de observaciones $\mathbf{Y} = GGCA$. ¿Cuál es la probabilidad $P(\mathbf{Y})$ de que la secuencia haya sido generada por el HMM descrito? Esta probabilidad puede ser calculada usando el algoritmo de avance-retroceso descrito anteriormente, vemos que:

| | Inicio | G | G | C | A |
|---|---|---|---|---|---|
| H | 0 | 0.5*0.3=0.15 | 0.15*0.5*0.3+0.1*0.4*0.3=0.0345 | ... + ... | 0.0013767 |
| L | 0 | 0.5*0.2=0.1 | 0.1*0.6*0.2 + 0.15*0.5*0.2=0.027 | ... + ... | 0.0024665 |

Entonces la probabilidad de que la secuencia $\mathbf{Y}$ haya sido generado por el HMM es $P(\mathbf{Y})$=0.0038432.

La probabilidad de que la secuencia $\mathbf{Y} = GGCA$ haya sido generada por el HMM es $PHMM(\mathbf{Y}) = 0{,}0038432$. Para evaluar la significancia de este valor, debemos comprobarlo

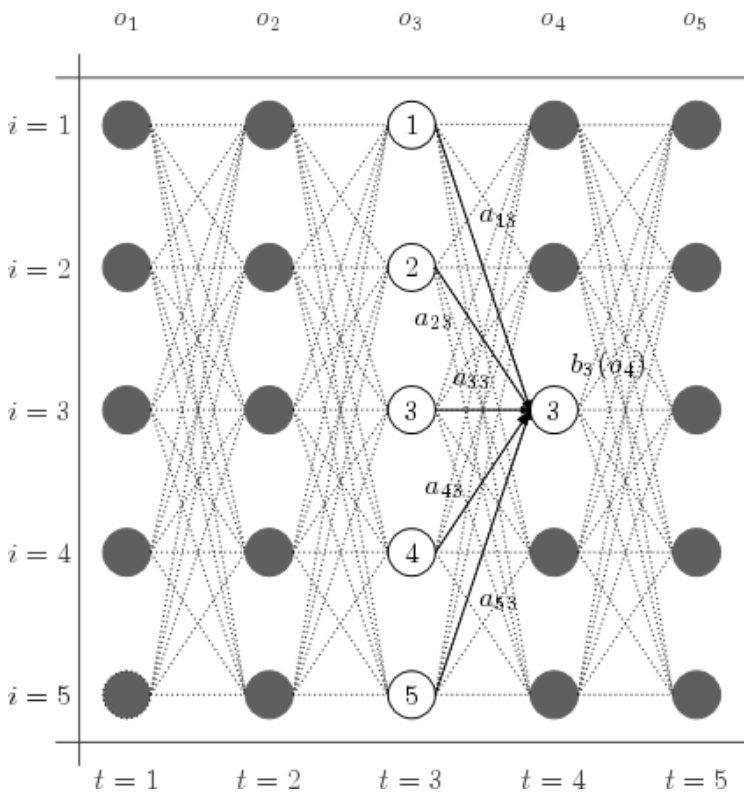


Figura 3.6: Estados y probabilidades necesarias para aplicar el algoritmo de avance.

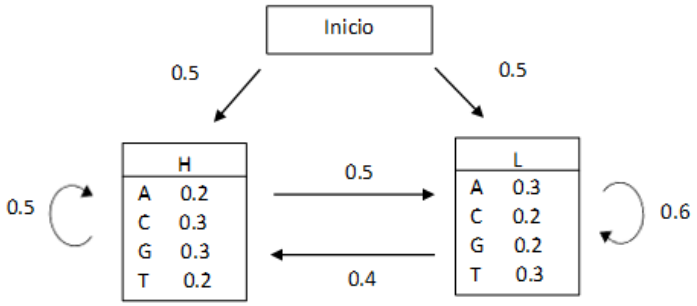


Figura 3.7: Estados y probabilidades necesarias para aplicar el algoritmo de avance.

con la probabilidad de que la secuencia $\mathbf{Y}$ no haya sido generada por el modelo, a esta probabilidad la denotaremos $P_{bg}$. Si todos los nucléotidos tienen la misma probabilidad, $p_{bg} = 0{,}25$; la probabilidad de observar $\mathbf{Y}$ es $P_{bg}(\mathbf{Y}) = p_{bg}^4 = 0{,}25^4 = 0{,}00396$.
Así que para este ejemplo en particular, es más probable que la secuencia no haya sido generada por el HMM ($P_{bg} > P(HMM)$).

**Proceso de retroceso**

Consideramos la variable

$$\beta_t(i) = P_\lambda(Y_{t+1} = y_{t+1}, Y_{t+2} = y_{t+2}, ..., Y_T = y_T \mid X_t = i) \tag{3.12}$$

Es decir, la probabilidad de obtener la secuencia de observación parcial $(y_{t+1}, y_{t+2}, ..., y_T)$ desde el momento de tiempo $t+1$ hasta el momento de tiempo $T$ (final), dado que el estado es $i$ en el momento de tiempo $t$ y dado el modelo $\lambda = (A, B, \Pi)$. Luego se pueden resolver las variables $\beta_t(i)$ nuevamente en forma inductiva y calcular el valor de $P_\lambda(\mathbf{Y})$ a través del siguiente algoritmo:

1. Inicialización:
$$\beta_T(i) = 1, \quad 1 \leq i \leq N. \tag{3.13}$$

2. Inducción:
$$\beta_t(i) = \sum_{j=1}^{N} p_{ij}\beta_{t+1}(j)e_j(y_{t+1}), \quad t = T-1, T-2, ..., 1; \;\; 1 \leq j \leq N. \tag{3.14}$$

3. Terminación:
$$P_\lambda(Y_1 = y_1, ..., Y_T = y_T) = \sum_{i=1}^{N} \beta_1(i)\pi_i e_i(y_1). \tag{3.15}$$

El primer paso (3.13), otorga el valor 1 a las $N$ variables $\beta_T(i)$.
El segundo paso (3.14), es el paso inductivo, nótese que $\beta_{t+1}(j)$ es la probabilidad de obtener la secuencia de observación parcial $(y_{t+2}, y_{t+3}, ..., y_T)$ dado que el estado en el momento $t+1$ es $j$ y dado el modelo, la misma se multiplica por la probabilidad de transición entre el estado $i$ y el $j$ y por la probabilidad de que el estado $j$, en el tiempo $t+1$ , emita el símbolo $y_{t+1}$, o sea, $e_j(y_{t+1})$. Obteniendo dicho producto para cada uno de los $N$ estados posibles y sumándolos, se obtiene $\beta_t(i$, que es precisamente la probabilidad de obtener la secuencia de observaciones parcial $(y_{t+1}, y_{t+2}, ..., y_T)$, dado que el estado en el momento $t$ es $i$, y dado el modelo. En la figura se puede ver una representación de este paso.
El paso final (3.15) obtiene lo que se quería en la dirección inversa al algoritmo anterior, nótese que en este caso a $\beta_t(i)$ se le debe multiplicar la probabilidad de comenzar en el estado $i$ y la probabilidad de que el estado $i$ emita el símbolo $y_1$, esto último se consideraba

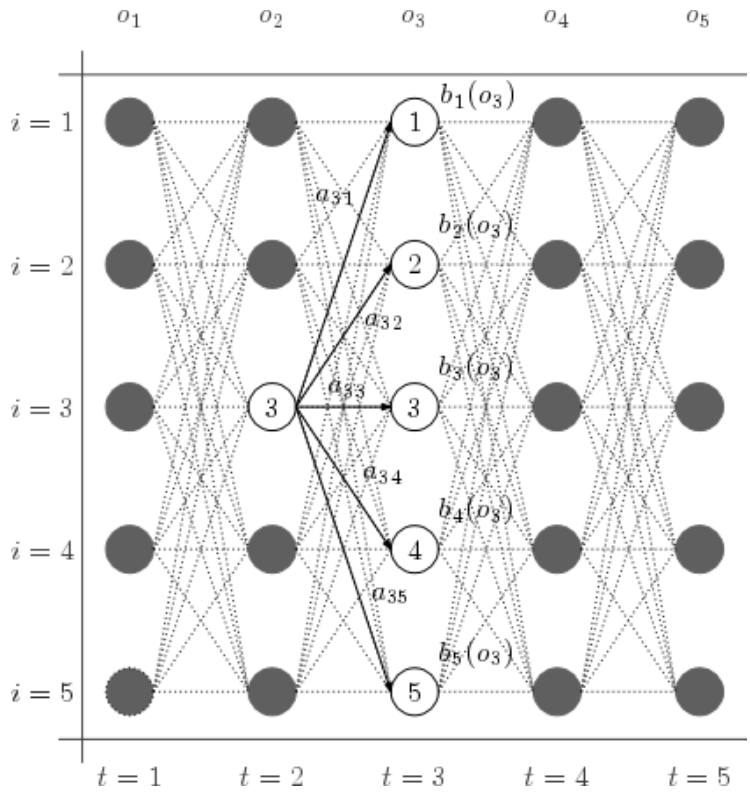


Figura 3.8: Estados y probabilidades necesarias para aplicar el algoritmo de retroceso.

en el algoritmo de avance (siguiendo la simetría) en el paso 1.

**Ejemplo del cálculo de** $\beta_2(3)$

La figura 3.8 muestra los estados y probabilidades necesarios para el cálculo de $\beta_2(3)$ para un modelo de 5 estados y una secuencia de observaciones de longitud 5. Donde $\beta_2(3) = \sum_{j=1}^{5} p_{3j}\beta_3(j)e_j(y_4)$.

### 3.3.2. Solución al problema 2: Secuencia de Estados más Probable

Como mencionamos, el objetivo es encontrar la secuencia de estados oculta óptima, es decir, de entre todas las secuencias de estados $X$ que pueden dar lugar a determinada secuencia de símbolos queremos encontrar aquella que explique mejor la secuencia de símbolos observada. Es decir, queremos encontrar "el mejor alineamiento" entre la secuencia de símbolos y el HMM, es por esto que a veces este procedimiento se denomina *problema del alineamiento óptimo*. Este problema lo podemos resolver haciendo uso del llamado *Algoritmo de Viterbi*.

Sea $X^*$=$(X_1^*, ..., (X_T^*)$, $\mathbf{X}$= $(X_1, ..., X_T)$ y $\mathbf{Y}$= $(Y_1, ..., Y_T)$. Formalmente, queremos en-

contrar la secuencia óptima $X^*$ que satisfaga lo siguiente

$$X^* = argmax_{\mathbf{X}} P(X_1 = x_1, ..., X_T = x_T | Y_1 = y_1, ..., Y_T = y_T)$$

Notamos que esto es lo mismo que encontrar la secuencia de estados que maximiza $P(Y_1 = y_1, ..., Y_T = y_T, X_1 = x_1, ..., X_T = x_T)$, dado que tenemos

$$P(X_1 = x_1, ..., X_T = x_T | Y_1 = y_1, ..., Y_T = y_T) = \frac{P(Y_1 = y_1, ..., Y_T = y_T, X_1 = x_1, ..., X_T = x_T)}{P(Y_1 = y_1, ..., Y_T = y_T)}$$

Encontrar la secuencia de estados óptima $X^*$ comparando todas las $M^L$ posibles secuencias de estados es computacionalmente inviable. Sin embargo, podemos utilizar un algoritmo de programación dinámica, llamado *algoritmo de Viterbi*, para encontrar la secuencia óptima $X^*$ eficientemente. El algoritmo se divide en dos partes: primero encuentra el $max\ P(X_1 = x_1, ..., X_T = x_T | Y_1 = y_1, ..., Y_T = y_T)$, y después realiza un procedimiento "hacia atrás" para encontrar la secuencia $\mathbf{X}$ que satisfaga este máximo.

En primer lugar, se define la variable

$$\gamma(n, i) = max_{X_1,...,X_{n-1}} P(Y_1 = y_1, ..., Y_n = y_n, X_1 = x_1, ..., X_{n-1} = x_{n-1}, X_n = i)$$

y se calcula recursivamente usando la siguiente fórmula

$$\gamma(n, i) = max_k \left[\gamma(n-1, k) p(k, i) e(x_n | i)\right]$$

Al final, podemos obtener la observación con la máxima probabilidad como sigue

$$P^* = max_{\mathbf{X}} P(Y_1 = y_1, ..., Y_T = y_T, X_1 = x_1, ..., X_T = x_T) = max_k \gamma(L, k)$$

El camino óptimo $X^*$ puede ser encontrado fácilmente yendo hacia atrás en las recursiones para llegar a la máxima probabilidad $P^* = P(Y_1 = y_1, ..., Y_T = y_T, X_1 = x_1, ..., X_T = x_T)$. El algoritmo de Viterbi encuentra la secuencia de estados óptima con una eficiencia del orden de $O(LM^2)$

**Ejemplo: el casino deshonesto.** Recordamos que la probabilidad de quedarse en cualquiera de los dos estados es 0.9 y la probabilidad de cambiar de estado es 0.1. Además, la probabilidad de obtener cualquier número en el dado libre es siempre la misma ($\frac{1}{6}$), mientras que en el dado cargado la probabilidad de obtener un 6 es 5 veces mayor que de obtener otro número. Veamos pues, un ejemplo sencillo de cómo aplicar el algoritmo de Viterbi para hallar la secuencia oculta dado que tenemos la secuencia de símbolos observada $\mathbf{Y} = (6, 2, 6)$. Así pues, para el símbolo 6 tenemos

$$\frac{1}{2} \times \frac{1}{6} = \frac{1}{12}$$

para el dado libre, mientras que para el dado cargado obtenemos

$$\frac{1}{2} \times \frac{1}{2} = \frac{1}{4} = \mathbf{0.25}$$

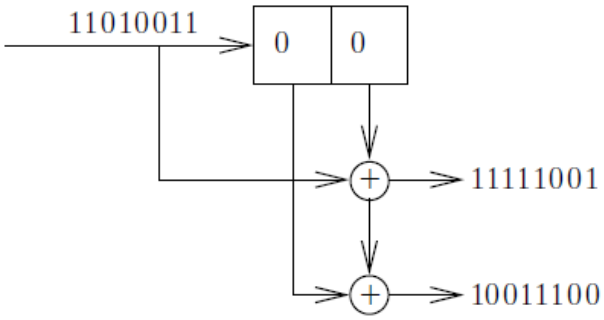


Figura 3.9: Código convolucional de razón 1/2

ahora, con el siguiente número observado que es el 2, obtenemos

$$\frac{1}{6} \times max\{(\frac{1}{12} \times 0{,}9, \frac{1}{4} \times 0{,}1)\} = 0{,}0125$$

$$\frac{1}{6} \times max\{(\frac{1}{12} \times 0{,}1, \frac{1}{4} \times 0{,}9)\} = \mathbf{0.0225}$$

Finalmente lo aplicamos para el último número observado, es decir, el 6

$$\frac{1}{6} \times max\{(0{,}0125 \times 0{,}9, 0{,}0225 \times 0{,}1)\} = 0{,}000375$$

$$\frac{1}{2} \times max\{(0{,}0125 \times 0{,}1, 0{,}0225 \times 0{,}9)\} = \mathbf{0.005625}$$

Por lo tanto, tomando los mayores valores de $\gamma$, que son los que están en negrita, obtenemos que la secuencia de estados ocultos óptima es $\mathbf{X} = (2, 2, 2)$ para la secuencia de símbolos observada $\mathbf{Y} = (6, 2, 6)$.

Ahora veamos cómo aplicar el algoritmo de Viterbi a un caso particular del Ejemplo 2 visto en la sección anterior, el cual se representa en la Figura 3.9.

En la Figura 3.9 vemos los datos de entrada y cuáles deberían ser los datos de salida o codificados, además también podríamos obtener los estados correspondientes. Luego, para ilustrar la aplicación del algoritmo de Viterbi, supondremos que recibimos los datos con algún error en algún bit, todo lo anterior lo vemos representado en la siguiente tabla.

| Datos | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|
| Estado presente | 11 | 10 | 01 | 10 | 00 | 01 | 11 | 10 |
| Codificado | 10 | 00 | 01 | 11 | 11 | 10 | 10 | 11 |
| Recibido | 10 | 00 | 01 | 01 | 11 | 11 | 10 | 10 |

Ahora bien, para aplicar el algoritmo de Viterbi hacemos uso de otro concepto: distancia de Hamming, la cual se define como la cantidad de bits que cambian de un bloque a otro y se denota como $d^H$. Básicamente lo que haremos será recorrer el diagrama de Trellis, hallar

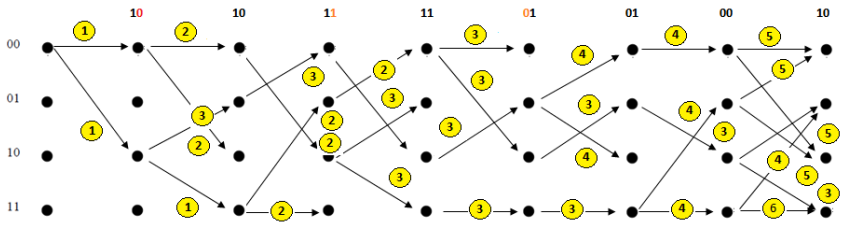


Figura 3.10: Diagrama de Trellis con la distancia de Hamming.

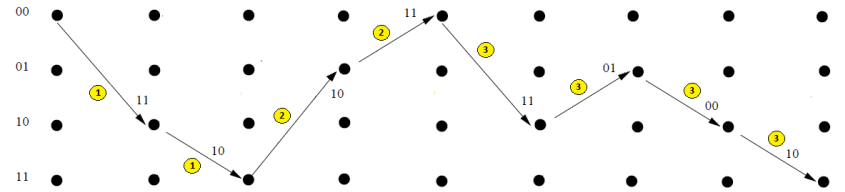


Figura 3.11: Ruta óptima encontrada con el algoritmo de Viterbi.

la distancia de Hamming entre los datos codificados y los datos recibidos para finalmente hallar la ruta óptima que será aquella cuya distancia recorrida sea menor. Veamos la Figura 3.10, notemos que algunas ramas se han borrado puesto que la distancia era muy grande y por ende quedan descartadas, las etiquetas en amarillo representan la distancia de Hamming.

Finalmente, la ruta óptima será la mostrada en la Figura 3.11.

### 3.3.3. Solución al problema 3: Estimación de los Parámetros del Modelo

Este el problema crucial de los HMM, debido a que determinar a priori las probabilidades correctas para un modelo en particular a priori es una tarea inviable en la mayoría de los problemas reales, éste es entre los tres problemas el más complicado de resolver y no se conoce una manera analítica de resolverlo; es decir, no se conoce la forma analítica de estimar un conjunto de parámetros para el modelo, que maximice la probabilidad de una secuencia de observaciones en particular de una forma cerrada.
Por lo tanto es necesario determinar a través de una *secuencia de entrenamiento*, las probabilidades que *mejor se adaptan* al modelo, como se verá en esta sección, para dicha tarea se analizará un algoritmo, denominado algoritmo de Baum-Welch. La idea del mismo es maximizar *localmente* la probabilidad de una secuencia de observaciones, lo que se consigue a través de un *procedimiento iterativo*.

Como ilustración, el algoritmo de Baum-Welch, intuitivamente, se podría describir de la siguiente manera. Inicialmente no se tiene conocimiento de los parámetros que mejor se ajustan a un modelo $\lambda = (A, B, \Pi)$, pero se dispone de una secuencia de observaciones (denominada secuencia de entrenamiento), que puede utilizarse para maximizar $P_\lambda(\mathbf{Y})$.

Como primer paso, se utilizan distribuciones de probabilidad preseleccionadas o se eligen aleatoriamente, entonces se identificarán las transiciones y símbolos de observación más probables. Se incrementarán las probabilidades de dichas transiciones y símbolos, y de esta forma se tendrá un nuevo modelo revisado y *mejor* que el anterior, es decir, con una mayor probabilidad de haber generado la secuencia de entrenamiento dada. Este proceso, denominado *proceso de entrenamiento*, se repite hasta que el modelo generado por el mismo, no difiera del generado en el paso anterior, o sea, hasta que no pueda *mejorarse* el modelo.
Pasando a una descripción formal, primero se define $\xi_t(i,j)$, que será la probabilidad conjunta de que el proceso se encuentre en el estado $i$, en el momento $t$, y que se produzca una transición hacia el estado $j$ en el momento $t+1$, dada la secuencia de observaciones $o$ y dado el modelo $\lambda = (A, B, \Pi)$, es decir, $\xi_t(i,j) = P_\lambda(X_t = i, X_{t+1} = j \mid \mathbf{Y})$. Se puede demostrar que

$$\xi_t(i,j) = \frac{\alpha_t(i) p_{ij} e_j(y_{t+1}) \beta_{t+1}(j)}{\sum_{k=1}^{N} \sum_{l=1}^{N} \alpha_t(k) p_{kl} e_l(y_{t+1}) \beta_{t+1}(l)}. \tag{3.16}$$

Sea $\gamma_t(i)$ la probabilidad de que el proceso se encuentre en el estado $i$ en el momento $t$, dada una secuencia de observaciones y dado el modelo. Razón por la cual, es posible relacionar $\gamma_t(i)$ con $\xi_t(i,j)$, notar que la diferencia, es que la última requiere que el proceso se encuentre en el estado $j$ en el momento $t+1$, es decir,

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j). \tag{3.17}$$

Si se suma $\gamma_t(i)$ a través del tiempo (desde $t = 1$ hasta $t = T-1$) se obtendrá el *número esperado de veces que el proceso efectúa transciones desde el estado i*, y si se efectúa el mismo procedimiento con $\xi_t(i,j)$, se obtienen el *número esperado de veces que el proceso produce transiciones desde el estado i al estado j*. O sea:

$$\sum_{t=1}^{T} \gamma_t(i). \tag{3.18}$$

Será el número esperado de transiciones desde el estado $i$ para una secuencia de observaciones $\mathbf{Y} = (Y_1, Y_2, ..., Y_T)$ y

$$\sum_{t=1}^{T} \xi_t(i,j). \tag{3.19}$$

Será a su vez, el número esperado de transiciones desde el estado $i$ hacia el estado $j$ para $\mathbf{Y} = (Y_1, Y_2, ..., Y_T)$.

Así pues, utilizando las expresiones (3.18) y (3.19) y la definición de $\gamma_t(i)$, se puede obtener un método para lograr la reestimación de los parámetros de un HMM. Sea $\overline{\lambda} = (\overline{A}, \overline{B}, \overline{\Pi})$ el modelo resultante de la reestimación de los parámetros, con $\overline{A} = \{\overline{p_{ij}}\}$ $1 \leq i, j \leq N$, $\overline{B} = \{\overline{e_j(k)}\}$ $1 \leq j \leq N$, $1 \leq k \leq M$ y $\overline{\Pi} = \{\overline{\pi_i}\}$ $1 \leq i \leq N$, entonces se pueden utilizar las siguientes fórmulas de reestimación:

$$\overline{\pi_i} = \gamma_1(i), \tag{3.20}$$

$$\overline{p_{ij}} = \frac{\sum_{t=1}^{T} \xi_t(i,j)}{\sum_{t=1}^{T} \gamma_t(i)}, \tag{3.21}$$

$$\overline{e_j(k)} = \frac{\sum_{t=1}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}, \tag{3.22}$$

donde, la primera ecuación será la cantidad de veces que el proceso se encuentra en el estado $i$ al momento $(t = 1)$, la segunda el número esperado de transiciones desde el estado $i$ al $j$, sobre el número esperado de transiciones desde el estado $i$ y la última el número esperado de veces en las que el proceso se encuentra en el estado $j$, observando el símbolo $v_k$ sobre el número de veces en las que el proceso se encuentra en el estado $j$. Si el modelo inicial es $\lambda = (A, B, \Pi)$, entonces Baum demostró que se verifica que $\lambda = \overline{\lambda}$, o bien, el modelo $\overline{\lambda}$ tiene una probabilidad mayor que $\lambda$, donde la mayor probabilidad se refiere a que se verifica la desigualdad $P_{\overline{\lambda}}(\mathbf{Y}) > P_{\lambda}(\mathbf{Y})$, lo que significa que se ha encontrado un modelo $\overline{\lambda}$, que posee mayor probabilidad de generar la secuencia de observaciones $\mathbf{Y}$. Esto asegura que si se repite el proceso tomando en cada paso el nuevo modelo como la base para el posterior cálculo, se producirá un paulatino acercamiento a los parámetros reales. Lo anterior es la plataforma para el siguiente algoritmo.

1. Elegir distribuciones de probabilidad $(A, B, \Pi)$, tal que $\lambda = (A, B, \Pi)$.
2. Calcular

$$\overline{\pi_i} = \gamma_1(i),\ \overline{p_{ij}} = \frac{\sum_{t=1}^{T} \xi_t(i,j)}{\sum_{t=1}^{T} \gamma_t(i)},\ \overline{e_j(k)} = \frac{\sum_{t=1}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}\ 1 \leq i, j \leq N, 1 \leq k \leq M. \tag{3.23}$$

3. Hacer $\overline{\lambda} = (\overline{A}, \overline{B}, \overline{\Pi})$.
4. Si $\overline{\lambda} = \lambda$ terminar y devolver $\overline{\lambda}$, sino, hacer $\lambda \to \overline{\lambda}$ y volver al paso 2.

# Capítulo 4

# Análisis de Proteínas

Las proteínas son moléculas formadas por cadenas de aminoácidos, donde los aminoácidos están representados por 20 letras del alfabeto, las cuales son: A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y. Así que de forma análoga al caso del ADN, una proteína es representada por una secuencia de letras tomadas del alfabeto anterior.

Las proteínas se pueden agrupar en familias, es decir, en grupos que están relacionadas evolutivamente. Las proteínas que pertenecen una familia descienden de un antepasado común y poseen estructura, funciones y secuencias similares, así pues, para determinar qué proteínas pertenecen a una familia se busca que esa similitud sea significativa, lo cual se evalúa haciendo uso de métodos de alineamiento de secuencias. Luego, al determinar si una secuencia desconocida es similar, en algún sentido, a secuencias conocidas podremos identificar y predecir su estructura y función.

Entre los métodos de alineamiento tenemos los globales y los locales, nosotros nos centraremos en los primeros. Una estrategia general de los primeros es el llamado algoritmo Neddleman-Wunsch, el cual describiremos a continuación. Para implementar el algoritmo de Neddleman-Wunsch, es necesario definir:

- Una función de puntuación ($\sigma$). Definimos la puntuación que daremos a una mutación por sustitución, ésta toma valores en $\mathbb{Z}$. Por ejemplo: 1 para coincidencia y $-1$ para no coincidencia.

- Una penalización por gaps ($\varsigma$). Define la puntuación que daremos a una mutación por inserción o deleción, toma un valor en $\mathbb{Z}^-$ . Por ejemplo: $-1$.

- Una relación de recurrencia $T$. Define las acciones que repetiremos en cada iteración del algoritmo, se define como

$$T(i,j) = max \begin{cases} T(i-1,j-1) + \sigma\left(S_1(i), S_2(j)\right) \\ T(i-1,j) + \varsigma \\ T(i,j-1) + \varsigma \end{cases}$$

  Donde $S_k$ es la k-ésima secuencia a alinear de tamaño $m$ o $n$ y con $i \in \{0,1,...,m+1\}$ y $j \in \{0,1,...,n+1\}$.

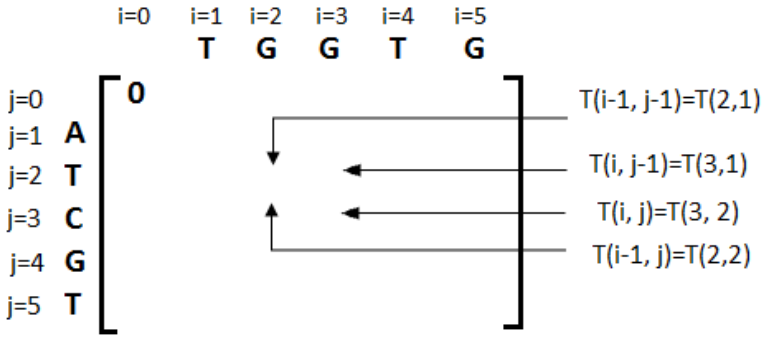


Figura 4.1: Primer paso para llenar la matriz

Ahora bien, el algoritmo consta de dos partes:

1. Llenar una matriz o tabla usando la relación de recurrencia previamente definida. Para construir esta matriz hay algunos aspectos a considerar como lo son:
   - Siendo $S_1$ de longitud $m$ y $S_2$ de longitud $n$, la matriz será de tamaño $m+1 \times n+1$.
   - T(0,0)=0.
   - Se llena de izquierda a derecha y de arriba a abajo.
   - Calculamos los valores de $T(i,j)$ para la fila 0 de izquierda a derecha.
   - Calculamos los valores de $T(i,j)$ para la fila 1, luego la fila 2, etcétera.
2. El camino hacia atrás. Usamos la matriz del paso 1 para encontrar el mejor camino hacia trás y por lo tanto el mejor alineamiento.

**Ejemplo.** Sean $S_1$ = TGGTG y $S_2$ = ATCGT secuencias a alinear, haremos uso del algoritmo de Needleman-Wunsch con el fin de encontrar el mejor alineamiento.
Así pues, primero definimos $\sigma$ como

$$\sigma\left(S_1(i), S_2(j)\right) = \begin{cases} 1 & coincidencia \\ -1 & no \quad coincidencia \end{cases}$$

Ahora, definimos $\varsigma = -2$.
Y finalmente, $T$ como arriba.
Para construir la matriz, como ya hemos mencionado, hacemos uso de la relación de recurrencia y los aspectos antes mencionados. En nuestro caso en particular la matriz será de tamao $6 \times 6$.

El proceso anterior descrito se ilustra en las Figuras 4.2, 4.3, y así sucesivamente, hasta obtener la matriz representada en la Figura 4.4.

Luego, para llevar a cabo la segunda parte, empezamos en la esquina inferior derecha y seguimos las flechas para calcular el mejor valor para esa celda, y así sucesivamente hasta

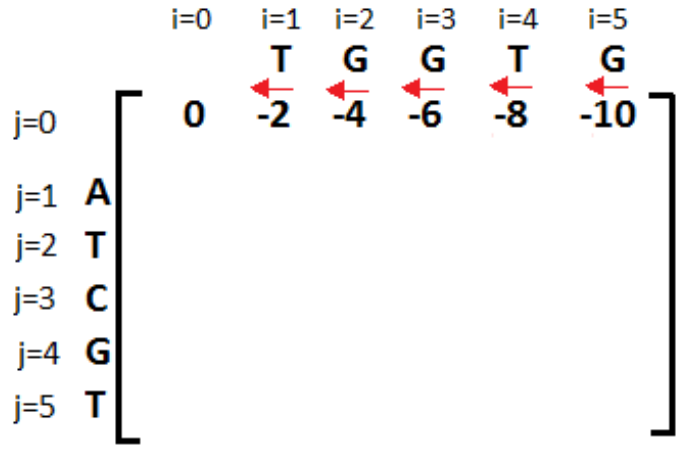


Figura 4.2: Segundo paso para llenar la matriz

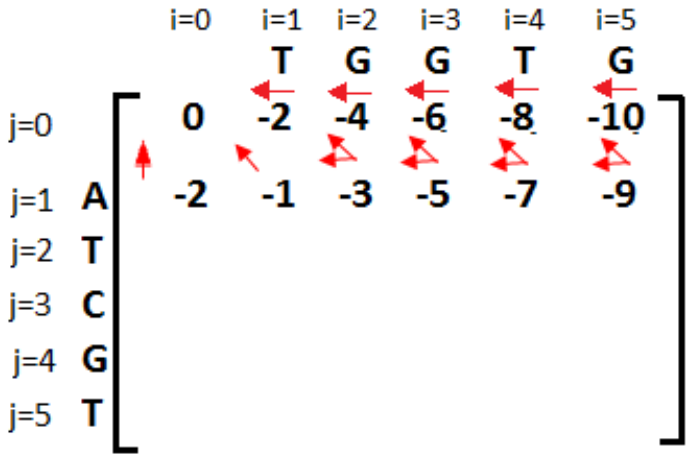


Figura 4.3: Tercer paso para llenar la matriz

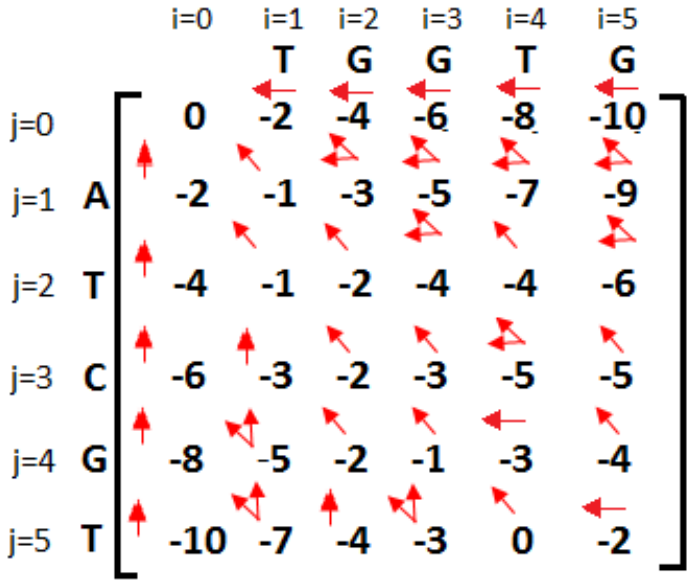


Figura 4.4: Matriz construida usando la relación de recurrencia.

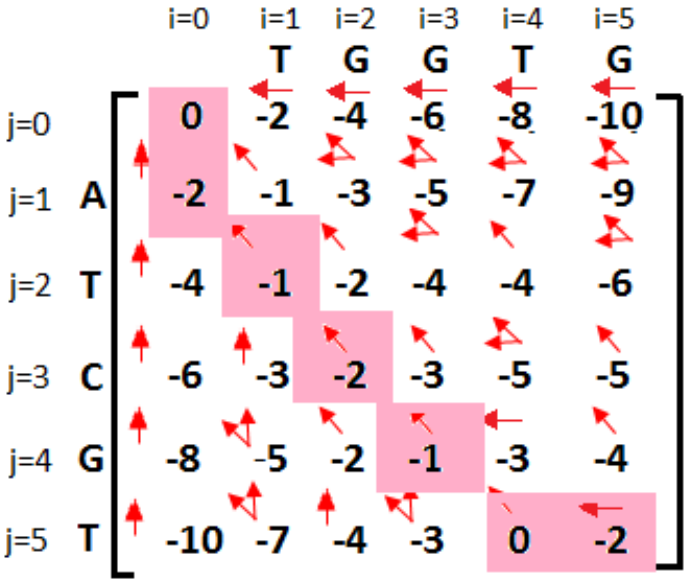


Figura 4.5: Alineamiento usando el algoritmo Needleman-Wunsh.

encontrar el camino hacia atrás que será el mejor alineamiento, lo cual se ilustra en la Figura 4.5.

Por lo tanto, obtenemos que el mejor alineamiento es

```
 -  T  G  G  T  G
    |     |  |
 A  T  C  G  T  -
```

As, pues una vez que tenemos el mejor alineamiento es necesario hacer uso de otra herrameinta para estimar los parámetros del modelo. En un capítulo anterior hablamos de los Modelos Ocultos de Markov, pues bien, para modelar una familia de proteínas utilizaremos una variante de estos llamada Perfiles de Modelos Ocultos de Markov. Veamos un ejemplo para comprender cómo funcionan.

Tomemos la siguiente familia de proteínas

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| HBA_HUMAN | ... | V | G | A | - | - | H | A | G | E | Y | ... |
| HBB_HUMAN | ... | V | - | - | - | - | N | V | D | E | V | ... |
| MYG_PHYCA | ... | V | E | A | - | - | D | V | A | G | H | ... |
| GLB3_CHITP | ... | V | K | G | - | - | - | - | - | - | D | ... |
| GLB5_PETMA | ... | V | Y | S | - | - | T | Y | E | T | S | ... |
| LGB2_LUPLU | ... | F | N | A | - | - | N | I | P | K | H | ... |
| GLB1_GLYDI | ... | I | A | G | A | D | N | G | A | G | V | ... |

La topología básica del HMM consite en tener tres estados por cada columna del alineamiento múltiple, excepto para las columnas en las que más de la mitad de sus elementos son huecos, como las columnas 4 y 5 del ejemplo anterior. Los estados son:

- Estados de emparejamiento $M$.
- Estados de inserción $I$.
- Estados de deleción $D$.

Luego, el modelo que representa la familia de proteínas será el representado en la Figura 4.6.

Ahora, para determinar lo parámetros del modelo, es decir, las probabilidades de emisión y transición de cada estado tomamos en cuenta el número de veces que cada transción y emisión es usada cuando el conjunto de secuencias alineadas es pasada por el modelo. Formalmente:

$$p_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}}$$

y

$$e_k(a) = \frac{E_k(a)}{\sum_{a'} E_k(a')}.$$

Cuando en la fase de entrenamiento se dispone de un número grande no hay problema, sin embargo, si son pocas puede darse que las probabilidades sean 0, así que para evitar

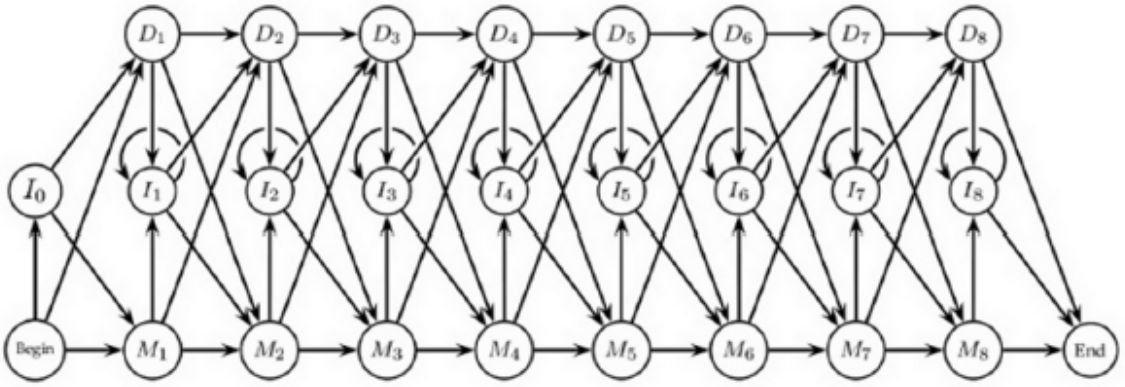


Figura 4.6: Perfil del Modelo Oculto de Markov asociado a la familia de proteínas.

esto se agrega 1 a cada frecuencia. Por ejemplo, para el alineamiento múltiple anterior las frecuencias de aparición en la primera columna son $V-5$, $F, I-1$ y los restantes 17 con 0. Al agregar 1 a cada frecuencia se tiene que ahora la frecuencia de aparición es: $V-6$, $F, I-2$ y los restantes 1.

Así se determina que $e_{M_1}(V) = \frac{6}{27}$, $e_{M_1}(F) = e_{M_1}(I) = \frac{2}{27}$ y $e_{M_1}(a) = \frac{1}{27}$ para $a \neq V, F, I$.

De manera similar se tiene que en la columna 1 hay 6 transiciones del estado de emparejamiento $M_1$ al siguiente, 1 transición a un estado de eliminación y 0 a un estado de inserción. Sumando 1 a cada frecuencia se obtiene $a_{M_1M_2} = \frac{7}{10}$, $a_{M_1D_1} = \frac{2}{10}$ y $a_{M_1I_1} = \frac{1}{10}$.

De forma análoga obtenemos los restantes parámetros, es decir:

- $e_{M_2}(G) = e_{M_2}(E) = e_{M_2}(K) = e_{M_2}(Y) = e_{M_2}(N) = e_{M_2}(A) = \frac{2}{26}$, $e_{M_2}(a) = \frac{1}{26}$ para $a \neq G, E, K, Y, N, A$ y $a_{M_2M_3} = \frac{7}{10}$, $a_{M_2D_2} = \frac{2}{10}$ y $a_{M_2I_2} = \frac{1}{10}$.

- $e_{M_3}(A) = \frac{4}{26}$, $e_{M_3}(G) = \frac{3}{26}$, $e_{M_3}(S) = \frac{2}{26}$, $e_{M_3}(a) = \frac{1}{26}$para $a \neq A, G, S$ y $a_{M_3M_4} = \frac{2}{10}$, $a_{M_3D_3} = \frac{7}{10}$ y $a_{M_3I_3} = \frac{1}{10}$.

- $e_{M_4}(A) = \frac{2}{21}$, $e_{M_4}(a) = \frac{1}{21}$ para $a \neq A$ y $a_{M_4M_5} = \frac{2}{10}$, $a_{M_4D_4} = \frac{7}{10}$ y $a_{M_4I_4} = \frac{1}{10}$.

- $e_{M_5}(D) = \frac{2}{21}$, $e_{M_5}(a) = \frac{1}{21}$ para $a \neq D$ y $a_{M_5M_6} = \frac{7}{10}$, $a_{M_5D_5} = \frac{2}{10}$ y $a_{M_5I_5} = \frac{1}{10}$.

- $e_{M_6}(N) = \frac{4}{26}$, $e_{M_6}(H) = e_{M_6}(D) = e_{M_6}(T) = \frac{2}{26}$ $e_{M_6}(a) = \frac{1}{26}$ para $a \neq N, H, D, T$ y $a_{M_6M_7} = \frac{7}{10}$, $a_{M_6D_6} = \frac{2}{10}$ y $a_{M_6I_6} = \frac{1}{10}$.

- $e_{M_7}(A) = e_{M_7}(Y) = e_{M_7}(I) = e_{M_7}(G) = \frac{2}{26}$, $e_{M_7}(v) = \frac{2}{26}$, $e_{M_7}(a) = \frac{1}{26}$ para $a \neq A, V, Y, I, G$ y $a_{M_7M_8} = \frac{7}{10}$, $a_{M_7D_7} = \frac{2}{10}$ y $a_{M_7I_7} = \frac{1}{10}$.

- $e_{M_8}(G) = e_{M_8}(D) = e_{M_8}(E)) = e_{M_8}(P) = e_{M_2}(A) = \frac{2}{26}$, $e_{M_8}(A) = \frac{3}{26}$, $e_{M_8}(a) = \frac{1}{26}$ para $a \neq G, D, A, E, P$ y $a_{M_8M_9} = \frac{7}{10}$, $a_{M_8D_8} = \frac{2}{10}$ y $a_{M_8I_8} = \frac{1}{10}$.

- $e_{M_9}(K) = e_{M_9}(T) = \frac{2}{26}$, $e_{M_9}(E) = e_{M_9}(G) = \frac{3}{26}$,$e_{M_9}(a) = \frac{1}{26}$ para $a \neq K, T, G, E$ y $a_{M_9M_{10}} = \frac{8}{10}$, $a_{M_9D_9} = \frac{1}{10}$ y $a_{M_9I_9} = \frac{1}{10}$.
- $e_{M_{10}}(Y) = e_{M_{10}}(D) = e_{M_{10}}(S) = \frac{2}{27}$, $e_{M_{10}}(V) = e_{M_{10}}(H) = \frac{3}{27}$, $e_{M_{10}}(a) = \frac{1}{27}$ para $a \neq Y, V, H, D, S$.

# Apéndice

**Algoritmo Baum Welch**

```
HMMsample <- function(nu, Q, g, n){
  cQ <- t(apply(Q, 1, cumsum));
  cg <- t(apply(g, 1, cumsum));
  x <- array(0, n);
  y <- array(0, n);
  x[1] <- 1+sum(as.numeric(runif(1)>cumsum(nu)));
  y[1] <- 1+sum(as.numeric(runif(1)>cg[x[1],]));
  for (t in 2:n){
    x[t] <- 1+sum(as.numeric(runif(1)>cQ[x[t-1],]));
    y[t] <- 1+sum(as.numeric(runif(1)>cg[x[t],]));
    #remark: it is slightly simpler, but also slightly slower to use:
    # x[t] <- sample(1:k, 1, prob = Q[x[t-1],])
    # y[t] <- sample(1:r, 1, prob = g[x[t],]);

  }
  list(x=x,y=y);
}

HMMfilter <- function(y, nu, Q, g){
  n <- length(y);
  dims <- dim(Q);
  res <- list();
  res$phi <- matrix(nrow = dims[1], ncol = n);
  res$c <- array(0, n);
```

```
  Z <- nu*g[,y[1]];
  res$c[1] <- sum(Z);
  res$phi[,1] <- Z/res$c[1];

  for (t in 2:n){
    Z <- (res$phi[,t-1] %*% Q) * g[, y[t]]
    res$c[t] <- sum(Z);
    res$phi[,t] <- Z / res$c[t];
  }
  res;
}

HMMsmoother <- function(y, Q, g, c){
  n<- length(y);
  dims <- dim(Q);
  beta <- matrix(1, nrow=dims[1], ncol=n);
  for (t in seq(n-1, 1, -1)){
    beta[,t] = Q %*% (g[,y[t+1]] * beta[, t+1]) / c[t+1];
  }
  beta;
}

HMMfilter_C = function(y, nu, Q, g){
  n <- length(y);
  N <- dim(Q)[1];
  M <- dim(g)[2];

  res <- .C("HMMfilter", as.integer(N), as.integer(M), as.integer(n), as.double(y), as.double(nu),
as.double(Q), as.double(g), as.double(array(0, N*n)), as.double(array(0, n)));
  list(phi = matrix(res[[8]], nrow = N, ncol = n), c = res[[9]]);
```

```
}

HMMsmoother_C = function(y, Q, g, c){
  n <- length(y);
  N <- dim(Q)[1];
  M <- dim(g)[2];

  res <- .C("HMMsmoother", as.integer(N), as.integer(M), as.integer(n), as.double(y), as.double(Q),
as.double(g), as.double(c), as.double(matrix(0, nrow = N, ncol = n)))[[8]];
  matrix(res, nrow = N, ncol = n);
}

HMMbaumwelch <- function(y, nu, tol = 1e-4, maxIt = 100){
  k <- length(nu);
  r <- max(y);
  n <- length(y);
  Y <- matrix(0, nrow=n, ncol=r);   for (t in 1:n){ Y[t, y[t]]=1; }
  Q <- matrix(runif(k*k), nrow=k);  Q <- Q / apply(Q, 1, sum);
  g <- matrix(runif(k*r), nrow=k);  g <- g / apply(g, 1, sum);
  it <- 0; oldQ <- Q-tol; oldg <- g-tol;
  while ((sum(abs((oldQ-Q))) + sum(abs(oldg-g)) > tol) & (it<maxIt)){
    it <- it+1;
    f <- myfilter(y, nu, Q, g);
    beta <- mysmoother(y, Q, g, f$c);
    post <- f$phi * beta;

    N <- Q * (f$phi[,1:(n-1)] %*% t(beta[, 2:n] * g[, y[2:n]]  / (matrix(1, nrow=k,
ncol=1)%*%f$c[2:n])));
    M <- post %*% Y;

    oldQ <- Q; oldg <- g;
```

```
    Q <- N / apply(N, 1, sum);
    g <- M / apply(M, 1, sum);
  }

  res <- list();
  res$Q <- Q;
  res$g <- g;
  res$l <- sum(log(f$c));
  res$it <- it;
  res;
}
```

# Bibliografía

[1] Warren J. Ewens, Gregory Grant (2005). Statistical methods in bioinformatics. An Introduction. Springer.

[2] Jose L. Oliver et al. (2001). Isocore chromosome maps of eukaryotic genomes. Gene 276, 4756.

[3] Lawrence R. Rabiner. "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition". Proceedings of the IEE, Vol. 77, No. 2