

Reconocimiento de voz, apuntes de cátedra para Introducción a la Inteligencia Artificial

Autores:
Fernando Martinez
Gustavo Portale
Hernan Klein
Osvaldo Olmos

Problemas del análisis y reconocimiento de voz.....	2
Breve historia del reconocimiento de voz	2
Algunas cuestiones sobre producción de la voz.....	2
Modelos de producción de la voz.....	3
Modelo “source-filter”	3
Preprocesamiento	4
Muestreo y cuantificación	4
Ventana.....	4
Análisis Cepstral.....	6
Análisis en escalas no lineales	8
Vector de Observación	9
Cuantificación de Vectores	10
Distintos conceptos de distancia	10
Distancia Euclidiana	10
Distancia No Euclidiana	11
Otras	11
Proceso de Cuantificación de Vectores	11
Componentes	12
Grupo de entrenamiento del VQ para reconocimiento de voz.....	13
Clasificación de Vectores	13
Algoritmos de Clasificación	13
Utilización del cuantificador y del codebook	17
Aplicaciones	17
Conclusiones	19
Modelos Ocultos de Markov (HMM).....	20
Procesos estocásticos.....	21
Clasificación de los procesos estocásticos.....	21
Procesos estocásticos de tipo Markov	24
Elementos de un proceso estocástico de tipo Markov	24
Procesos de tipo Markov según la naturaleza de las variables.....	25
Representación de procesos de tipo Markov	26
Propiedades y estructuras de matrices de estado	26
Ejemplo de cadena de Markov	27
Modelos ocultos de Markov (HMM).....	28
Elementos de un HMM.....	31
Inconvenientes de los HMM.....	32
HMM y problemas asociados	33
Tipos de HMMs.....	33
Aplicaciones de HMM.....	35
Entrenamiento de HMM.....	36
Visión Global del proceso de entrenamiento/reconocimiento utilizando HMM.....	38
Bibliografía	42

Problemas del análisis y reconocimiento de voz

El análisis de la señal de voz y su posterior reconocimiento deben superar algunos problemas que en principio parecen triviales ya que son superados de forma sencilla por los seres humanos, algunos de ellos son, la correcta elección y extracción de las características de la señal de voz, tratar con corrección las variaciones inherentes a género, velocidad de emisión, pronunciación y acentos, tamaños de los vocabularios a reconocer, ruido y distorsión de los entornos donde se utilizan, inclusive hasta el estado de ánimo del locutor.

Pese a las dificultades, se ha logrado gracias a múltiples corrientes independientes de investigación y desarrollo, sistemas de uso real, en los cuales la exactitud es superior al 90%, siempre considerando tareas acotadas de una u otra manera.

Por ejemplo reconocimiento de dígitos, para un solo locutor, en canales sin ruido, se logran niveles de más de 99% de exactitud.

Sistemas comerciales para vocabularios grandes, obtienen de 90% a 95%, cayendo a 87% para diferentes locutores y diferentes canales.

Breve historia del reconocimiento de voz

La historia del reconocimiento de voz, se remonta en el tiempo, hasta 1950 con múltiples paradigmas de trabajo y resultados, inclusive muchas de las técnicas utilizadas con éxito debieron esperar más de 10 años para pasar de la teoría a la práctica inclusive en laboratorios.

Algunos de sus principales hitos fueron 1952 en Bell labs, reconocimiento aislado de dígitos, medición de resonancia espectral en vocales, con rangos de 50 al 100%, 1959 reconocimiento de vocales y algunas consonantes, con analizador de espectro y comparadores de patrones con resultados del 93%, ambos dispositivos de hardware y exclusivamente en laboratorio.

En los '60 se comenzó a experimentar con normalización temporal según la detección de los puntos de comienzo y fin de las palabras, utilizando en gral. hardware específico e inicios de uso de computadoras.

En los '70 hubo avances significativos en reconocimientos de palabras aisladas, y comienzos de experimentación en reconocimiento independiente del locutor (speaker independent).

Se advierte que las fuentes de información semántica, sintáctica y contextual, ayudan a mejorar la calidad de los sistemas.

El reconocimiento de una sentencia completa de gramática acotada requería de 50 computadoras (HARPY system del Carnegie Mellon University).

En los '80 se aplicaron los conceptos de dynamic time warping. Se produce un importante cambio de paradigma de comparación de plantillas hacia el modelado estadístico/probabilístico como un gran avance de aproximación al reconocimiento de voz.

A mitad de los '80 se hizo masiva una técnica que revolucionó el campo de reconocimiento se trata de los modelos ocultos de Markov o HMM que obtuvo excelentes resultados en el modelado de señales de voz y virtualmente indispensable hoy en día.

Se reintroduce el uso de redes neuronales (ANN) que habían vencido algunos obstáculos de tipo conceptual y de recursos necesarios para su implementación. También se comenzó a experimentar con reconocimiento continuo de vocabularios largos independientes del locutor.

En los '90 se comenzó a hacer énfasis en interfaces de lenguaje natural, y recuperación de la información en grandes documentos de voz, continuó la investigación de reconocimiento continuo en vocabularios grandes y a usarse masivamente a través de redes telefónicas, también en el estudio de sistemas en condiciones de ruido.

Antes y durante la mitad de los '90 se dio la investigación de sistemas híbridos HMM-ANN, que también han dado excelentes resultados siendo la excelencia de los motores de reconocimiento de voz de hoy en día.

Algunas cuestiones sobre producción de la voz

Los sonidos vocales son producidos por la acción del aire que impulsan los pulmones sobre el tracto vocal, en la laringe existen dos membranas (cuerdas vocales) que permiten variar el área de la tráquea por la cual circula (glotis).

Durante el habla, dichas membranas permanecen en continuo movimiento (abrir y cerrar) lo que origina una mezcla de características en la generación de la voz, que se dan en llamar "voiced" y "unvoiced", o sonorizado y no sonorizado.

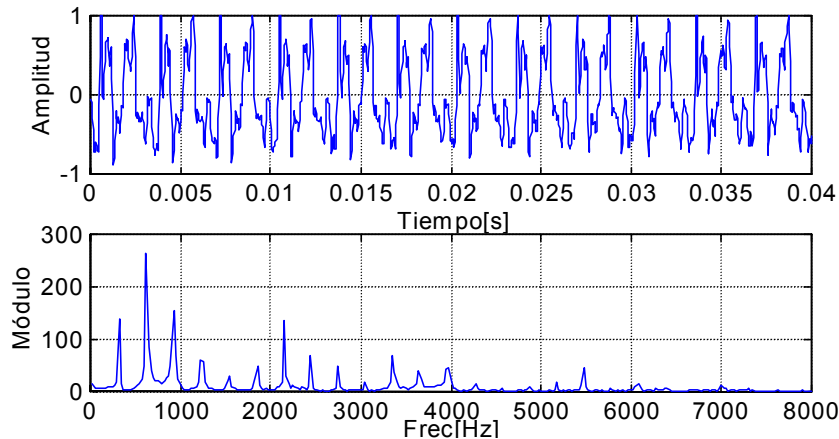
En los tramos sonorizados, las cuerdas vocales están normalmente cerradas y vibran al ser recorridas por la corriente proveniente de los pulmones, su frecuencia (pitch) se determina por el largo y tensión de las membranas y está en el orden de 50 a 400 Hz. El efecto de este continuo abrir y cerrar de la glotis aparece como un tren de pulsos casi continuo.

En los no sonorizados, las cuerdas vocales permanecen abiertas y el aire pasa al resto del tracto vocal en forma de turbulencia.

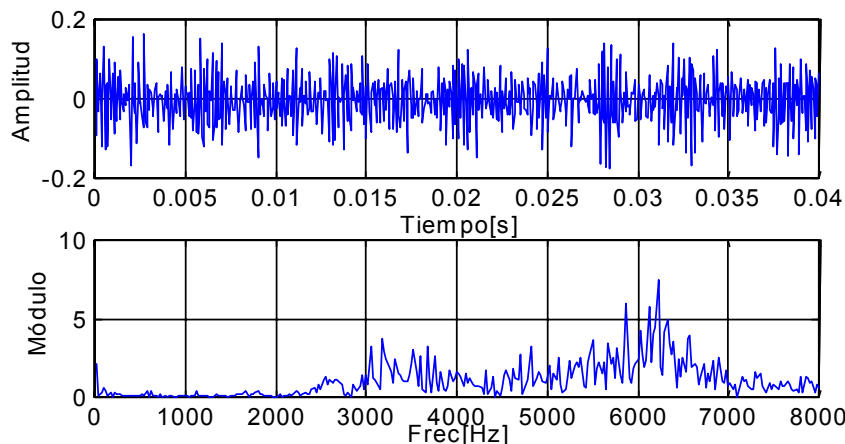
Queda así establecido que el resultado del habla en la región de la glotis será, una sucesión de pulsos o ruido blanco, según sea sonorizado o no, pero lo que termina de moldear el aspecto de la voz es el resto del tracto vocal que da "forma" al sonido actuando como un filtro que impone su propia respuesta en frecuencia.

Para los sonorizados, como las vocales, el tracto vocal actúa como cavidad resonante, que produce picos en el espectro resultante conocidos como "formants" o formantes (Normalmente cerca de los 500 Hz), que en sí mismos contienen la mayoría de la información de la señal y están formados de impulsos correspondientes a la naturaleza vibratoria de las cuerdas vocales. La forma y ubicación de los formantes depende de forma general de su tamaño y características particulares, y de forma particular del tracto vocal, posición de la lengua, labios, mandíbulas, etc. es decir todo lo que forma parte de la articulación de los sonidos.

La figura muestra un tramo de señal de voz sonorizada y su espectro, en el que se observan 3 formants" (a 750Hz y armónicos impares).



A continuación se observa un segmento no sonorizado, con su correspondiente espectro, sobresale su característica similar al ruido y su espectro tipo pasa altos.

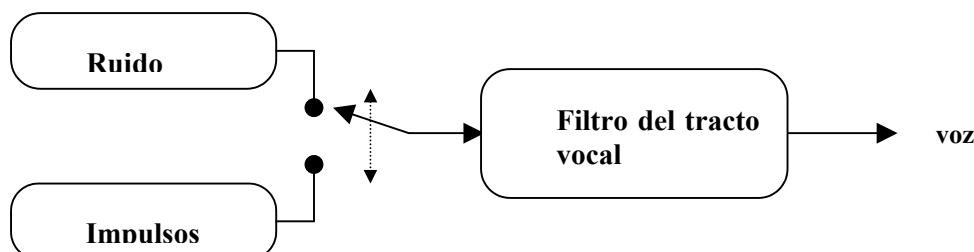


Modelos de producción de la voz

Trataremos a continuación uno de los posibles modelos que se utilizan para caracterizar la producción vocal y las técnicas con las que pueden ser aproximados.

Modelo "source-filter"

En él se considera a la voz producida por una señal de excitación en forma de impulsos que provienen de la acción de las cuerdas vocales, alternado de forma aleatoria con ruido blanco, que alimenta a un filtro de características variables



aunque con una constante de tiempo mucho más lenta.

De esta manera se puede incorporar al modelo la información de cuerdas vocales más la información del tracto vocal, cuyas piezas móviles están consideradas con las características variables del filtro. La información del tracto vocal está contenida en la envolvente del espectro resultante.

Preprocesamiento

Muestreo y cuantificación

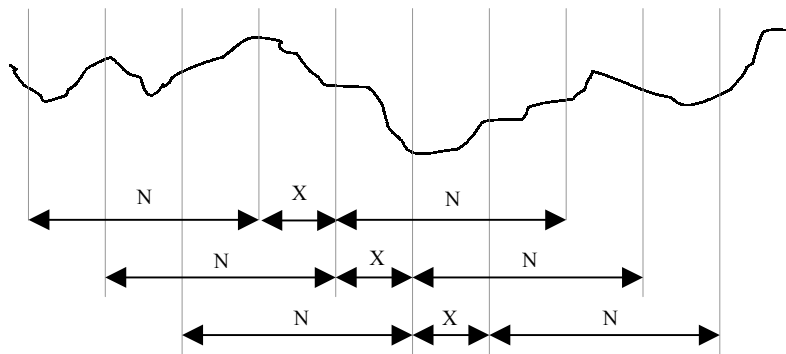
Según Nyquist sabemos que es necesaria una frecuencia de muestreo de por lo menos el doble del ancho de banda de la señal a caracterizar, sobre esta base y para un análisis mínimo (en lo que respecta a frecuencia) de la señal de voz se utiliza una frecuencia de muestreo f_s de 8kHz, aunque se suele usar 16kHz si se desea obtener mayor detalle en frecuencia lo que mejora la resolución para tratamiento de la señal.

La cuantificación más comúnmente usada, es de 8 bits, mínimo requerido para una calidad baja, puede mejorarse su S/R con una técnica no lineal de cuantificación (ley A), se obtienen excelentes resultados aumentando la cuantificación a 16 bits.

Ventana

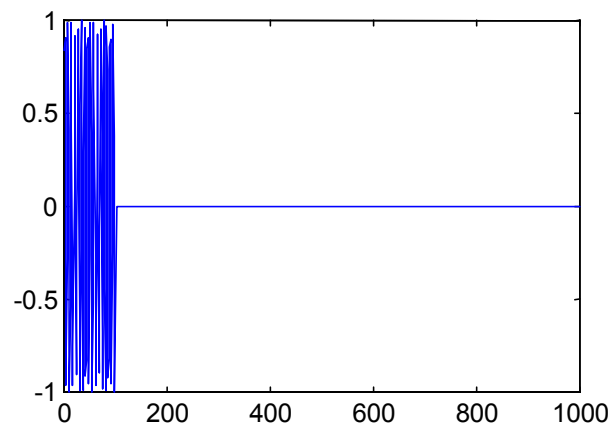
La señal vocal debe ser separada en bloques de una duración tal que pueda ser considerada estacionaria. Para la señal de voz, según las experimentaciones realizadas, esta duración (o ancho de la ventana temporal) está dentro de un intervalo comprendido entre 10ms y 45ms.

Independientemente los bloques son superpuestos desplazándolos en una cantidad de muestras quedando una distribución como la siguiente:

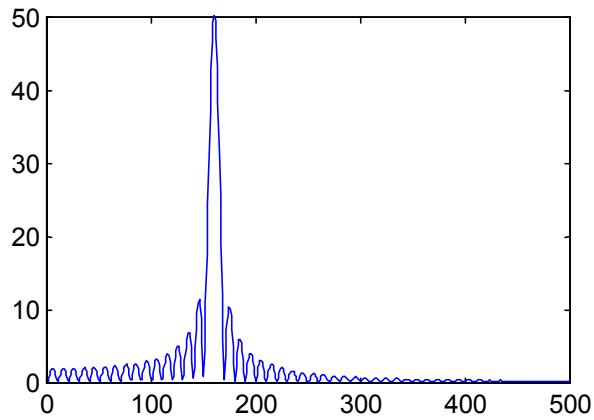


Es evidente que una función de ventana rectangular sería la más sencilla de implementar pero su uso trae algunos inconvenientes en el dominio de la frecuencia.

Supongamos una señal senoidal a la que aplicamos una función de ventana rectangular:



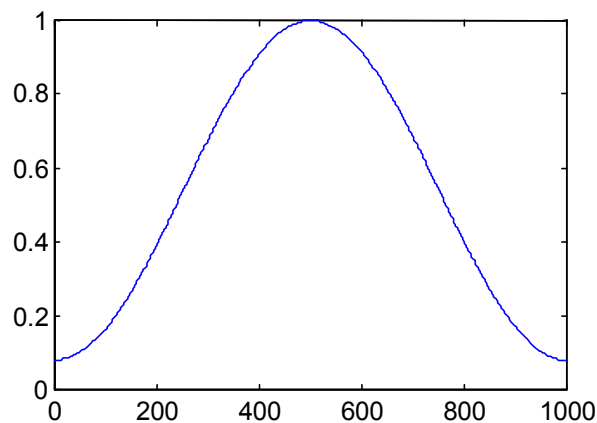
Si le aplicamos la transformada de Fourier resulta:



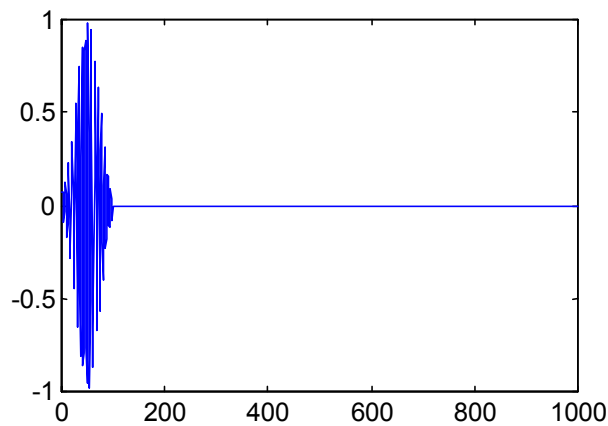
Sabemos que el espectro de una señal senoidal es un impulso a la frecuencia correspondiente, se observa entonces la deformación de dicho espectro debido al truncamiento temporal y discontinuidad en los extremos.

Este problema puede minimizarse multiplicando en el dominio temporal por una ventana que reduzca la influencia de los extremos. Una ventana muy utilizada es la ventana de Hamming cuya función es:

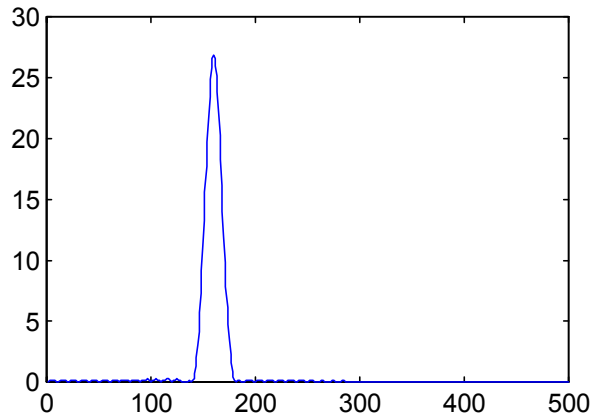
$$w_n = \begin{cases} 0,54 - 0,46 \cdot \cos(2\pi n / (N-1)) & \text{si } 0 \leq n \leq N-1 \\ 0 & \text{si } 0 > n > N \end{cases}$$



Siguiendo con el ejemplo anterior, y aplicando la ventana de Hamming a la función senoidal queda:



Su DFT es:



Como se observa, el error queda minimizado, pero se pierde en cuanto a la energía total, ya que resulta menor que la original.

Análisis Cepstral

El modelo source-filter, descompone la señal de voz S , en excitación E y un filtro lineal caracterizado por su respuesta en frecuencia $H(f)$.

Sabemos que en el dominio temporal se puede expresar con una convolución:

$$S(t) = H(t) * E(t)$$

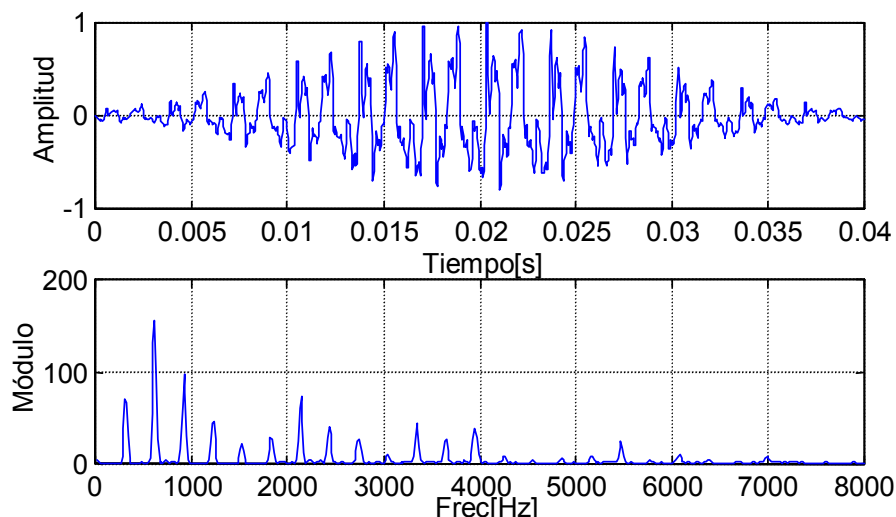
siendo H la respuesta al impulso del filtro

En el dominio de la frecuencia:

$$S(f) = H(f) \cdot E(f) \quad \text{siendo } S(f) = |S(f)| e^{j\phi}$$

$H(f)$ es la envolvente del espectro y $E(f)$ los detalles que se aproximan a impulsos.

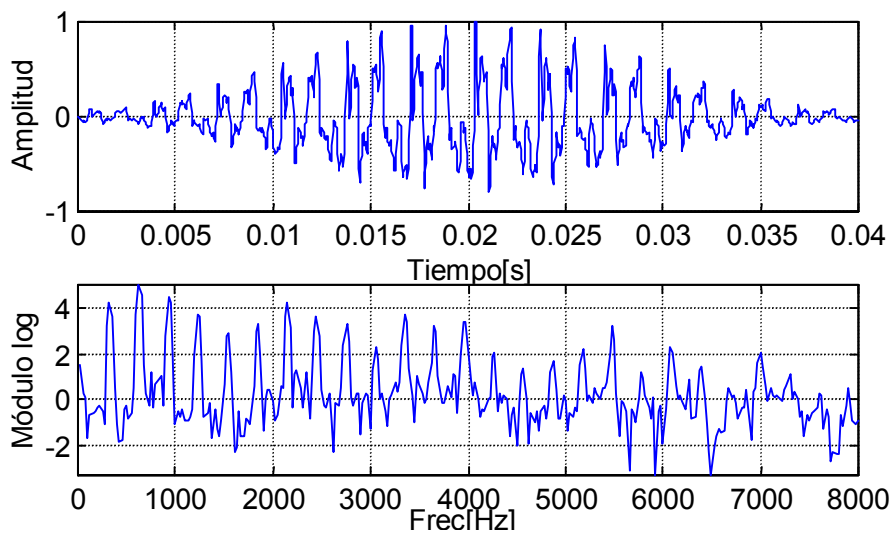
Observemos el siguiente ejemplo del espectro obtenido de la FFT de un segmento de una vocal tratado con una ventana de Hamming.



Si aplicamos logaritmos a ambos miembros, sin considerar su fase, que normalmente no es necesaria en el campo de señales de voz, queda:

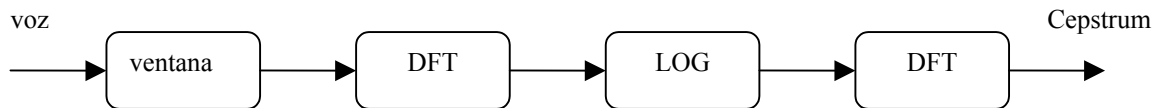
$$\log |S(f)| = \log |H(f)| + \log |E(f)|$$

En la figura observamos el espectro logarítmico a partir del gráfico anterior:

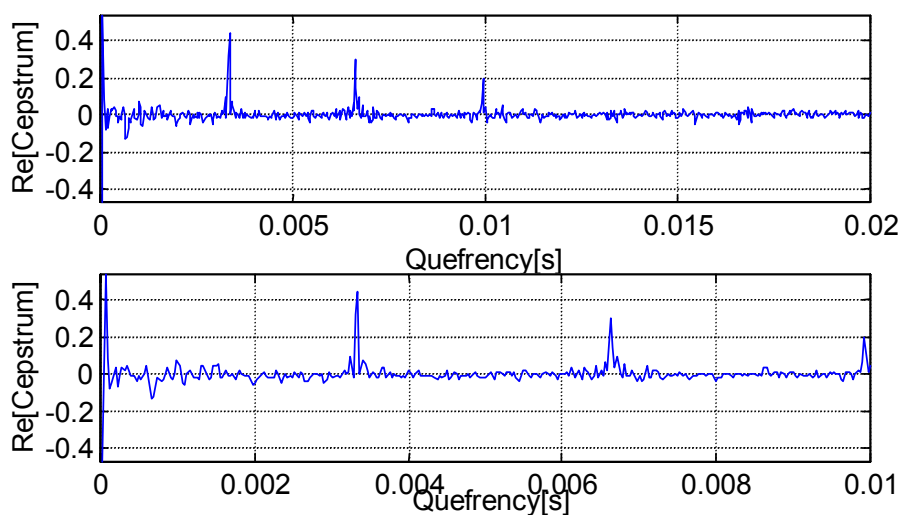


Logramos pasar de una convolución de difícil resolución a una suma de logaritmos, en definitiva una suma de dos señales de muy distintas características en frecuencia, ya que la señal de excitación tiene variaciones temporales muy superiores a la respuesta de filtro, si se realizara una nueva transformada de Fourier al resultado actual, se separarían claramente las señales en regiones apartadas ya que la respuesta del filtro ocupará las bajas frecuencias y la excitación las regiones de más alta frecuencia.

Se logra separar la envolvente de los impulsos o detalles de la voz, mediante el análisis Cepstral, que se diagrama a continuación.



Si se calcula la parte real del Cepstrum del ejemplo anterior se obtiene:

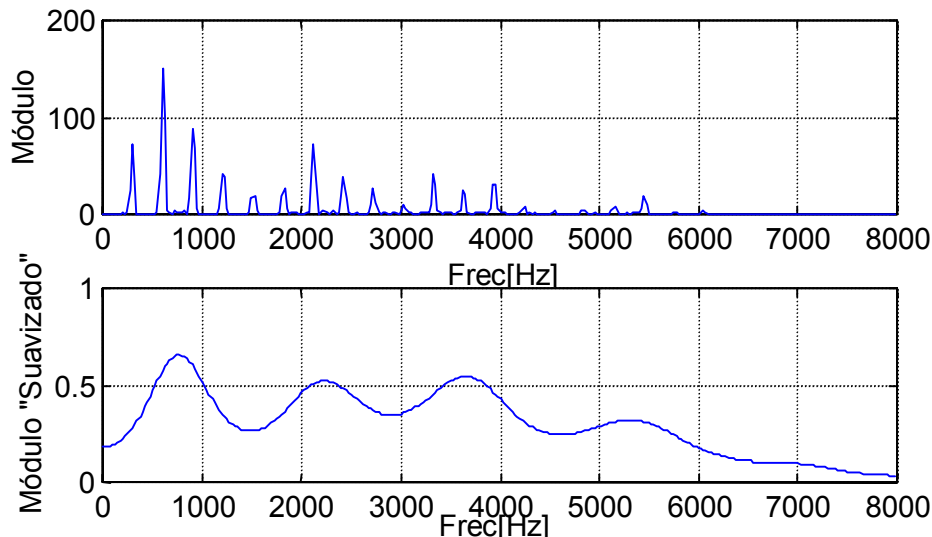


En el se observa en dos escalas, la gran cantidad de información cerca del origen y más alejado sólo los impulsos aislados que caracterizan el tono (pitch), separados por el período del mismo. Los coeficientes de orden bajo proveen entonces información sobre la envolvente.

En general el análisis Cepstral logra pasar de una convolución a una multiplicación por medio de la transformada de Fourier y luego a una suma con el uso del logaritmo, finalmente con una nueva transformada se logra el Cepstrum, la palabra está originada por la alteración de las letras que forman Spectrum ya que en realidad su dominio no es temporal ni frecuencial sino un nuevo dominio Cepstral. Se tiene así un método para separar la información del tracto vocal de la correspondiente a la excitación.

Si esta excitación es de alguna manera eliminada y se vuelve al dominio de la frecuencia, se obtiene un espectro "suavizado" o alisado, proceso que se conoce como filtrado Homomórfico y se observa a continuación.

En la figura se observa el espectro original y la señal luego del filtrado Homomórfico en el que se destacan los 3 *formants* que lo componen.



Análisis en escalas no lineales

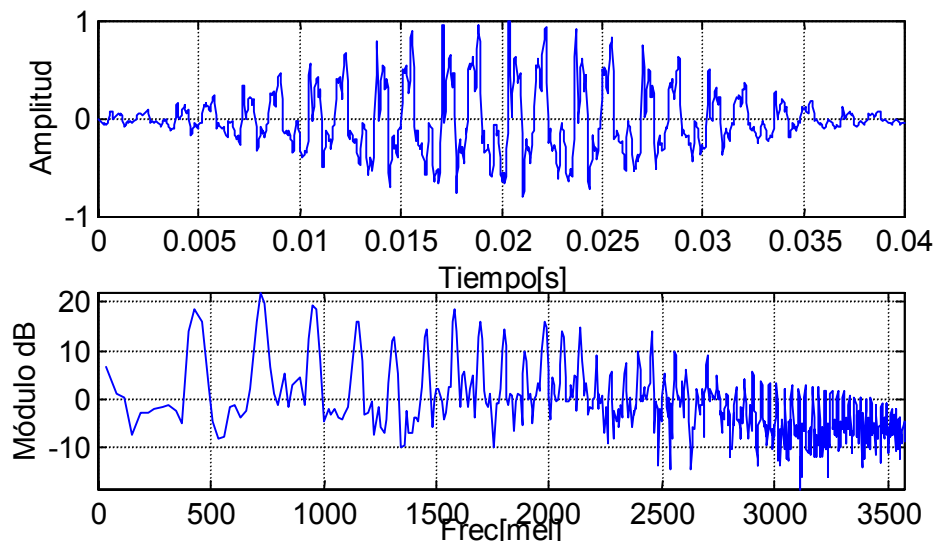
Se considerará la escala de frecuencia Mel que aproxima la sensibilidad del oído humano y es:

$$m = 2595 \cdot \log(1 + f/700) \quad \text{ó} \quad f : \text{frecuencia en Hertz}$$

$$m = \frac{1000}{\ln(1 + 1000/700)} \cdot \ln(1 + f/700)$$

Usando una escala de este tipo, se trata de tener el mejor compromiso entre la resolución frecuencia tiempo, ya que usa un ancho de banda pequeño en baja frecuencia lo que permite resolver armónicos, y en alta frecuencia un ancho de banda más grande que permita buena resolución de ráfagas temporales.

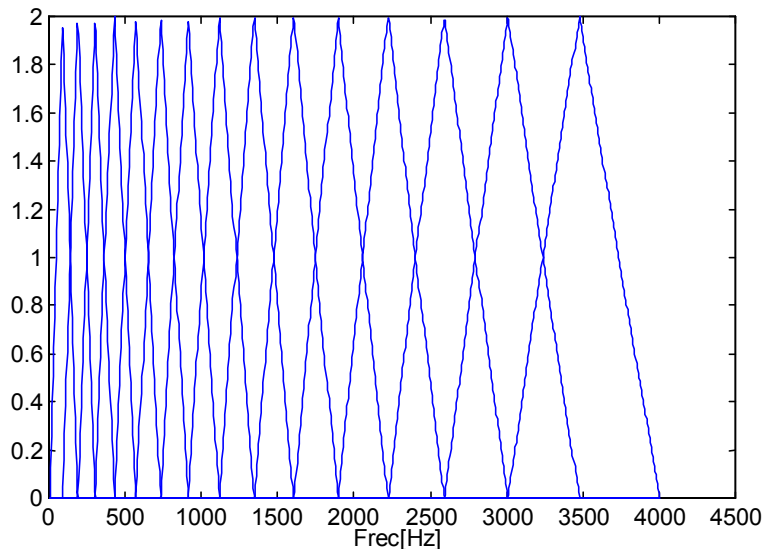
A continuación tenemos el espectro de potencia en dB transportado a la escala Mel:



Una caracterización de la señal vocal que da actualmente grandes resultados es la extracción de los coeficientes cepstrum obtenidos a partir de la escala de Mel, dichos coeficientes se denominan, "Mel-Cepstrum".

Para obtenerlos una vez hallado el espectro de la señal vocal, se procede a filtrar mediante un banco de filtros en el dominio de Mel a partir de los cuales se obtienen las correspondientes bandas energía que luego debidamente tratadas formarán parte del cepstrum. A dichos coeficientes se pueden agregar para incrementar su eficiencia, la potencia o logaritmo de la potencia, la primera y segunda derivadas del cepstrum llamadas respectivamente *diferencia* y *aceleración*, que aportan información dinámica del proceso.

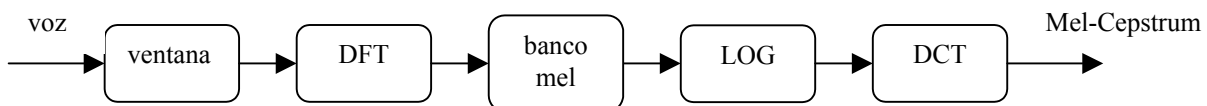
En general se toman filtros triangulares para formar el banco Mel, en el siguiente gráfico se observa un banco de 15 filtros triangulares en el dominio de la frecuencia, (equiespaciados en el dominio Mel):



De esta manera se logra no sólo aproximar la escala Mel sino reducir de una manera significativa la cantidad de información a procesar en posteriores etapas.

Por último, en lugar de una transformada de Fourier, suele usarse en el campo del tratamiento de voz, ya sea para codificación o reconocimiento, la transformada del coseno, usada por su propiedad de compresión de energía, lo que otorga un mejor modelo hacia las frecuencias bajas que en definitiva es la información correspondiente al tracto vocal.

Si representamos en bloques la sucesión de pasos queda:



Vector de Observación

Al final de los distintos pasos para el tratamiento de la señal de voz, se obtendrá un vector que contendrá la información vocal que representará a la ventana temporal correspondiente, de alguna manera una colección de características que describen de la mejor manera posible la voz humana. Estos vectores son conocidos en la literatura del reconocimiento de voz como vectores de *Observación*.

Cabe aclarar que existen varias formas de representación de estas características como LPC (Linear Prediction Code) o Auditory System, pero la que en la actualidad da los mejores resultados es el análisis Cepstral, en particular los coeficientes MFCC (Mel Frequency Cepstral Coefficients). También suele incorporarse al vector de *Observación* la información de la primera y segunda derivadas del Cepstrum con respecto al tiempo para agregar información de las características dinámicas del sistema y el logaritmo de la energía total de la ventana.

Gráficamente, se observa de la siguiente manera:

[illegible]

Cuantificación de Vectores

Una parte importante en cualquier tipo de procesamiento de voz es la optimización de los algoritmos en cuanto a velocidad y almacenamiento, entonces, la cuantificación de vectores trae consigo la idea de clasificar un conjunto de vectores, luego de lo cual se buscarán los mejores representantes para reducir el tamaño de la información a manejar. La forma de medir la fidelidad de un cuantificador es determinar el error que éste produce al reemplazar los datos de entrada que recibe por los vectores representantes o *codewords*, dicho parámetro es llamado error por distorsión. La finalidad de un cuantificador es obtener un conjunto de vectores representativos llamado *codebook*, que presente el menor error por distorsión, por ejemplo para cuantificar los vectores de observación. Veamos algunos conceptos importantes:

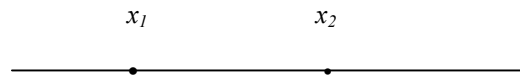
Distintos conceptos de distancia

Existen tres formas principales para medir distancias:

Distancia Euclidiana

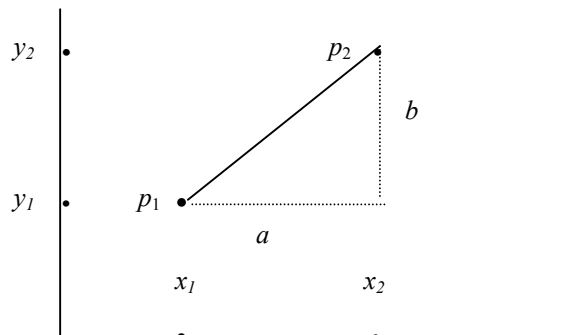
Mide la línea recta que une dos puntos en un espacio Euclidiano. Si tomamos como ejemplo un espacio unidimensional la distancia será la resta de ambas coordenadas:

$$d(x_1, x_2) = x_2 - x_1$$



en un plano será la hipotenusa del triángulo rectángulo formado por los puntos (Pitágoras)

$$d(p_1, p_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} = \sqrt{a^2 + b^2} \quad \text{con} \quad \begin{cases} p_1 = (x_1, y_1) \\ p_2 = (x_2, y_2) \end{cases}$$



en tres dimensiones:

$$d(p_1, p_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \quad \text{con} \quad \begin{cases} p_1 = (x_1, y_1, z_1) \\ p_2 = (x_2, y_2, z_2) \end{cases}$$

y por extensión en un espacio multidimensional de orden n , se calcula una "hipotenusa" n dimensional, lo que generaliza el cálculo de la distancia mínima:

$$d(p_1, p_2) = \sqrt{\sum_n (x_i - y_i)^2} \quad \text{con} \quad \begin{cases} p_1 = x_1, x_2, \dots, x_n \\ p_2 = y_1, y_2, \dots, y_n \end{cases}$$

Variantes de la distancia Euclidiana

También es usado por cuestiones de simplificación de la potencia de cálculo, la distancia Euclidiana al cuadrado:

$$d(p_1, p_2) = \sum_n (x_i - y_i)^2 \quad \text{con} \quad \begin{cases} p_1 = x_1, x_2, \dots, x_n \\ p_2 = y_1, y_2, \dots, y_n \end{cases}$$

Distancia de Chebychev:

En ella se usa el valor absoluto de la máxima diferencia entre las coordenadas o elementos del vector.

$$d(p_1, p_2) = \max |x_i - y_i| \quad \text{con} \quad \begin{cases} p_1 = x_1, x_2, \dots, x_n \\ p_2 = y_1, y_2, \dots, y_n \end{cases}$$

Distancia "Potencia":

Se utiliza en aquellos casos en los que se desea actuar sobre la contribución progresiva que produce cada una de las coordenadas multidimensionales.

$$d(p_1, p_2) = \left(\sum |x_i - y_i|^p \right)^{\frac{1}{p}} \quad \text{con} \quad \begin{cases} p_1 = x_1, x_2, \dots, x_n \\ p_2 = y_1, y_2, \dots, y_n \end{cases}$$

El parámetro p , controla la contribución progresiva de la diferencia de coordenadas individuales, mientras que r actúa sobre el "peso" que tienen grandes diferencias entre los vectores comparadas con otras muy pequeñas.

Distancia No Euclidiana

No se trata de una línea recta entre los puntos, sólo se definen ciertas reglas que debe cumplir una distancia no Euclidiana:

$$d(p_1, p_2) \geq 0$$

$$d(p_1, p_2) = d(p_2, p_1)$$

$$d(p_1, p_1) = 0$$

$$d(p_1, p_2) \leq d(p_1, p_x) + d(p_x, p_2)$$

La última expresión hace referencia a que si se consideran 3 objetos, la distancia entre dos de ellos no puede superar la suma de las distancias entre los dos restantes.

Existe otra forma de medición de distancias que obedece a las tres primeras reglas pero no a la última de ellas, se trata de una medición semi métrica. Un ejemplo es la distancia de bloques o manzanas como en las calles de una ciudad.

Otras**Medición Coseno**

También llamada correlación ya que da una medida de la relación entre vectores de datos, el coseno entre dos vectores es igual a su correlación:

$$d(p_1, p_2) = \frac{\sum (x_i \cdot y_i)}{\sqrt{\sum x_i^2 \cdot \sum y_i^2}} \quad \text{con} \quad \begin{cases} p_1 = x_1, x_2, \dots, x_n \\ p_2 = y_1, y_2, \dots, y_n \end{cases}$$

Proceso de Cuantificación de Vectores

Las técnicas de parametrización de la señal de voz se realizan tomando una secuencia de ventanas temporales, cada una de las cuales se representa por un número de D parámetros. Entonces la información de cada ventana se representaría por un vector de observación de D posiciones.

Cuando se almacenan estos parámetros lo que generalmente se hace es cuantificar cada parámetro con un determinado número de bits, este proceso se denomina cuantificación escalar y no es la manera más eficiente para almacenar la información, además, implica la ocurrencia uniforme de las ventanas de información. Una forma más conveniente es realizar una cuantificación vectorial.

Si comparamos la información del vector representante con respecto a la forma de onda original de la señal de voz, concluimos que el análisis espectral contiene significativamente menos información.

Por ejemplo, una señal de voz se muestrea a 10Khz y la cuantificación es de 16 bits, se necesita una velocidad de 160000 bps para almacenar las muestras de la señal de voz en el formato original. Si realizamos el análisis en el espectro, consideremos vectores de dimensión $n=10$ usando 100 vectores de observación por segundo. Si representamos cada parámetro en 16 bits, se requiere aproximadamente $100 \times 10 \times 16$ bps o 16000 bps con una reducción de diez veces sobre la señal original.

Las compresiones en ancho de banda y almacenamiento son imprevisibles, se basan en el concepto de la necesidad de la representación única para cada fonema (sonido diferenciable de una lengua, generalmente representado por una letra), esto puede ser posible para reducir la representación espectral original de la señal de voz sacando a los vectores de observación desde un pequeño, finito número de vectores espectrales "únicos", donde cada uno corresponde a las unidades básicas de la voz o "fonemas".

La representación ideal es impracticable porque hay mucha variabilidad en las propiedades espectrales de cada uno de los fonemas.

De cualquier forma, el concepto de construir un *codebook* de vectores de análisis, "distintos" y "únicos", aunque con mas palabras de código que el grupo o set básico de fonemas, sigue siendo una idea atractiva y es el fundamento de un conjunto de técnicas denominadas métodos de cuantificación de vectores.

Basándose en este razonamiento, se necesita un *codebook* con aprox. 1024 vectores espectrales únicos (25 variantes para cada uno de los 36 fonemas básicos).

Si para representar un vector espectral arbitrario tenemos un número de 10 bits, tomando una velocidad de 100 vectores por segundo, obtenemos una velocidad de 1000 bps para representar los vectores espectrales de una señal de voz.

Esta velocidad es aprox. 1/16 de la velocidad necesaria para vectores espectrales continuos.

Por lo tanto la representación cuantificada es eficiente para representar información espectral de la señal de voz.

Principales ventajas

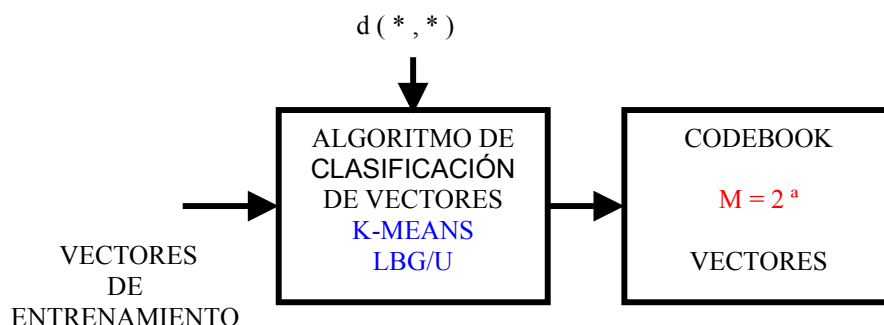
1. Reduce el almacenamiento de la información de análisis.
2. Se reduce el cálculo para determinar distancias entre vectores espectrales. La representación del VQ se limita a una tabla que contiene las distancias entre pares de vectores del *codebook*.
3. Representación discreta de las señales de voz. Asociando una característica fonética con cada vector del *codebook*, el proceso de elección del vector que mejor lo representa es equivalente a asignar una característica fonética a cada segmento de voz.

Principales desventajas

1. Distorsión en la representación del vector. Hay un número finito de vectores en el *codebook*, el proceso de "elección" del mejor representante es equivalente a cuantificar el vector y conduce a un cierto nivel de error de cuantificación. De cualquier modo con cualquier *codebook* finito siempre habrá un nivel de ruido o error.
2. El almacenamiento requerido para los vectores del *codebook* no es pequeña. Cuanto más grande sea el *codebook* menor es el error. Para un *codebook* de 1000 o más entradas, el almacenamiento no es irrelevante. Hay que realizar un balance entre error de cuantificación, procesamiento y almacenamiento del *codebook*.

Componentes

Para construir un VQ se necesita:



1. Un gran número de vectores de observación, V_1, V_2, \dots, V_n , que conforman el grupo de entrenamiento. El grupo de entrenamiento se usa para crear el grupo de vectores del *codebook* "optimo" que representa la variabilidad

espectral observada en el grupo de entrenamiento. Determinamos el tamaño del *codebook* como $M = 2^a$, siendo a el número de bits necesarios para codificar M palabras de código, *por lo tanto* se necesitan $n \gg M$ vectores para que sea eficaz.

2. *Una medición de distancia* entre cada par de vectores espectrales de observación para agrupar el conjunto de vectores de entrenamiento como así también para asociar o clasificar vectores arbitrarios a cada entrada del *codebook*.
3. *Un procedimiento de clasificación para ubicar y calcular los centroides*. Sobre la base del particionamiento que clasifica el grupo de n vectores en M clusters o sectores primero elegimos el número M , *codewords* del *codebook*, para luego proceder a la clasificación.
4. *Finalmente*, luego del proceso de clasificación (entrenamiento) queda como resultado del mismo un libro de códigos o *codebook*.

Grupo de entrenamiento del VQ para reconocimiento de voz

Para entrenar apropiadamente el *codebook* y mejorar la implementación, para el grupo de vectores de entrenamiento, se deberá tener en cuenta:

1. Para las señales de voz:
 - Rangos de edad, acentuación, genero, velocidad de discurso, niveles y otras variables.
2. Condiciones de discurso:
 - Ambiente ruidoso o silencioso, movilidad de la persona.
3. Transductores y sistemas de transmisión:
 - Ancho de banda del micrófono, canal telefónico, ancho de banda del canal y otros dispositivos.
4. Reconocimiento discreto o de palabras aisladas y reconocimiento continuo.

Clasificación de Vectores

El objetivo de un módulo clasificador es agrupar una cantidad de vectores característicos, N , en una cantidad M ($M < N$), discreta, de sectores o celdas de clasificación logrando que las características en cada sector sean similares. Existen muchos criterios para lograr dicho objetivo y a continuación veremos algunos de los más comunes. Imaginemos que la media multidimensional de un determinado sector i , es μ_i (con $1 < i < M$), y a continuación ingresa al clasificador un vector de observación o , se puede clasificar dicho vector calculando la "distancia" a la que se halla de cada una de las M medias y asignándolo al sector más "cercano".

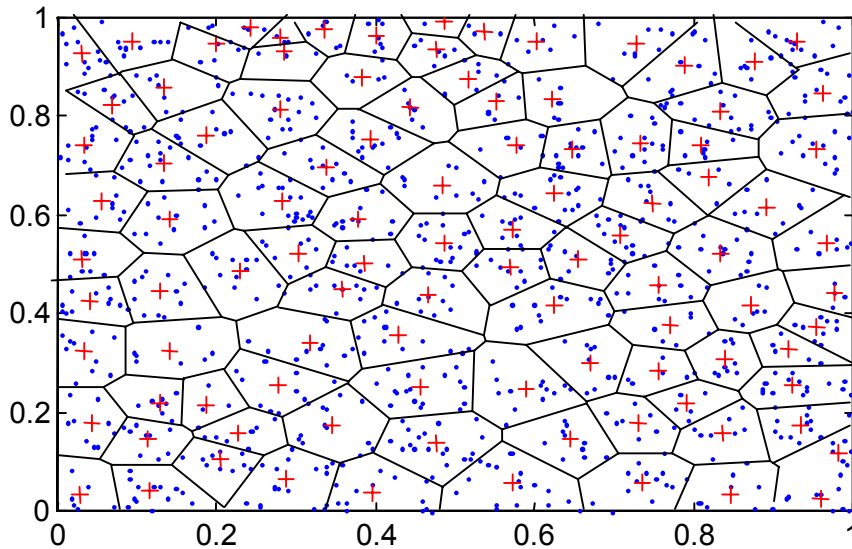
Este método de clasificación se denomina *k-Means* debido a que se agrupan los vectores en torno a k valores medios, quedando formados k sectores (en nuestro caso $k=M$). Existe el problema de inicialización de los valores de μ_i , y su reestimación a medida que progresa el algoritmo.

Algoritmos de Clasificación

Podemos decir, en general, que los N vectores originales de tamaño D quedarán representados por M vectores, cada uno de los cuales es llamado "palabra de código" o *codeword* (Cw), el grupo entero de dichos vectores, forma un "libro de códigos" o *codebook*, quedan entonces delimitadas M regiones o sectores, llamados regiones de *Voronoi*, determinados por la siguiente expresión:

$$V_i = \left\{ x \in \mathbb{R}^D : d(x, Cw_i) \leq d(x, Cw_j) \right\} \forall j \neq i \text{ con } 1 \leq i \leq M$$

En la figura se observa un diagrama de *Voronoi* con sus correspondientes sectores, conformando un *codebook* con sus correspondientes *codewords* como centroides.



La forma en la cual un grupo de N vectores de observación de entrenamiento pueden ser clasificados en un grupo de M vectores del *codebook* es mediante el algoritmo *K-Means*.

Algoritmo K-Means

1. INICIALIZACIÓN: Arbitrariamente elegimos M vectores o palabras de código, *codewords*, como el grupo inicial del *codebook*.
2. BÚSQUEDA DEL MÁS CERCANO: Por cada vector de observación, se busca el *codeword* en el *codebook* que es el más cercano (en términos de distancia), y asigna a ese vector a la celda correspondiente.
3. ACTUALIZACIÓN DEL CENTROIDE: actualiza el *codeword* en cada celda o sector usando el centroide de los vectores de entrenamiento asignados a un sector.
4. ITERACIÓN: Repite los pasos 2 y 3 hasta que la distancia media caiga debajo de un umbral prefijado.

La forma de cada sector o celda o partición es muy dependiente de la medida de distorsión espectral y las estadísticas de los vectores en el grupo de entrenamiento.

Este método es el más simple y por tanto existen numerosas modificaciones y mejoras, algunos de sus puntos débiles son:

1. Los resultados dependen en forma muy acentuada de los valores iniciales elegidos como palabras de código.
2. También hay gran dependencia del número de sectores M así como de la implementación de la "distancia" usada.
3. Puede suceder que algunos de los sectores resulten vacíos.

Algoritmo LBG

Se analizará con algún detalle debido a su buen desempeño, para eso comenzaremos por el algoritmo fundamental LBG. El algoritmo LBG, lleva su nombre debido a sus autores Y. Linde, A. Buzo y R. M. Gray, en él se elige 1 *codeword* inicial de entre los vectores de datos a clasificar, luego se utiliza el algoritmo de división binaria para duplicar el número de *codewords*, los vectores de observación se agrupan en torno a los *codewords* que les presentan menor distancia, se recalculan los *codewords* como la media multidimensional de cada sector y se agrupan nuevamente los datos, el proceso se detiene cuando el *codebook* no presenta variación significativa y al llegar al número de *codewords* deseados.

Este algoritmo de gran popularidad (que utiliza el algoritmo *k-Means*) produce *codebooks* que logran un mínimo local en la función de error por distorsión.

Para generar un *codebook* de M sectores o palabras de código:

En primer lugar designando un *codeword* inicial para luego utilizando una técnica de división llegar a obtener un *codebook* inicial, luego iterando la misma técnica de división en los *codewords* hasta que llegamos a obtener el número de *codewords* igual a M que va a ser el tamaño del *codebook* deseado.

El procesamiento se denomina división binaria:

1. Designar 1 vector del *codebook* o *codeword* inicial, éste resulta ser el centroide del grupo de los vectores de entrenamiento.
2. Calcular la media del grupo de entrenamiento:

$$Cw_i = \frac{1}{N} \sum_{n=1}^N x_n$$

- 2.1. Calcular el error o distancia media entre el *codeword* inicial y los vectores de entrenamiento:

$$D = \frac{1}{N} \sum_{n=1}^N \|x_n - Cw_i\|^2$$

3. Duplicar el tamaño del *codebook* mediante la división de cada *codeword*:

$$Cw_i^+ = Cw_i(1+\epsilon)$$

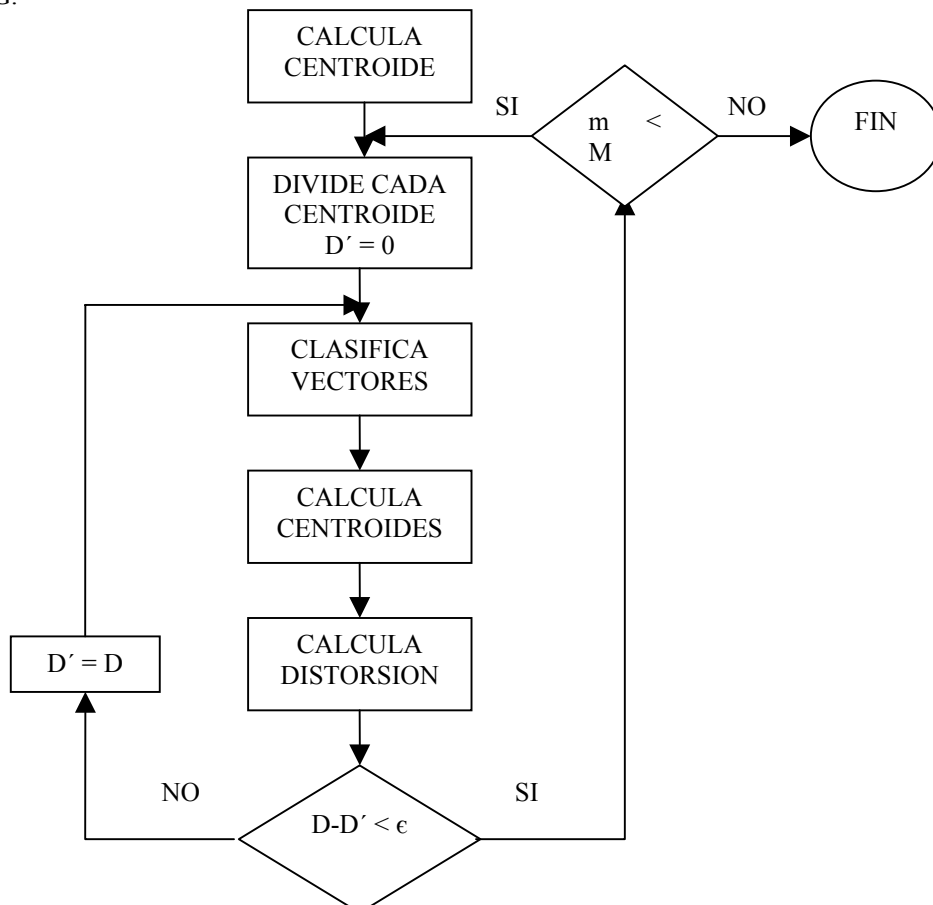
$$Cw_i^- = Cw_i(1-\epsilon)$$

$$0 < \epsilon < 1$$

4. Usar el algoritmo *K-Means* para tomar el mejor grupo de centroides para la separación del *codebook*.
5. Iterar pasos 3 y 4 hasta llegar a un *codebook* de tamaño *M*.

Una de las causas que motivo el uso de un VQ fue la suposición que, en el limite, el *codebook* debería idealmente tener 36 vectores, uno por cada fonema, suposición que es incorrecta.

En el esquema de la figura se representa el algoritmo que sintetiza el proceso de generación del *codebook* mediante el método LBG.



Algoritmo de generación del *codebook* por división binaria

Algoritmo LBG-U

Existe una modificación que permite obtener una sustancial mejora, y el que la logra es el algoritmo LBG-U. Si existiera algún parámetro que describiera la influencia de cada *codeword* al error total, podría tratar de modificarse el que fuera más importante, es decir podría situarse otro *codeword* muy cercano a él de forma de reducir el error. El problema es ahora encontrar el *codeword* que debemos tomar de nuestro *codebook* para realizar esta tarea, la solución sería contar con otro parámetro que informara sobre la contribución de cada *codeword* a la reducción del error, esto se logra gracias a la medida de utilidad que da el nombre al método.

La utilidad de cada *codeword* se obtiene si, teniendo el error por distorsión total restamos el error por distorsión que resulta de un *codebook* al que se le extrae únicamente el *codeword* representativo i . Refresquemos entonces la idea de error por distorsión E :

$$E(D, C) = \sum_{x \in D} d(x, Cw_{i(x)}) \text{ siendo } D : \text{Conjunto de vectores de datos } x$$

C : Codebook

$Cw_{i(x)}$: Codeword representativo del sector i que le corresponde al vector x

Se trata de la sumatoria de las distancias de todos los vectores a cuantificar, a su *codeword* representativo.

La medida de utilidad U para cada sector se logra comparando la distorsión del *codebook* C , con la que tendría si el *codeword* a evaluar no existiera:

$$U(Cw_i) = E(D, C - Cw_i) - E(D, C)$$

La remoción de un *codeword* afecta la distorsión sólo para aquellos vectores de dicho sector, que en su ausencia se incorporarán al sector cuyo representante sea el segundo en cuanto a su distancia, mientras que el resto no se verá afectado.

Entonces se puede reescribir U como:

$$U(Cw_i) = \sum_{x \in V_i} d(x, Cw_{i'(x)}) - d(x, Cw_i) \quad \text{con } Cw_{i'(x)} \text{ codeword secundario para el vector } x$$

V_i región i

El *codeword* que presente menor utilidad será el más apto para ser removido pues será el que ejerza menor incidencia sobre el error, marca lo útil que es cada *codeword* para el *codebook*.

El error correspondiente a cada sector $E(Cw_i)$:

$$E(Cw_i) = \sum_{x \in V_i} d(x, Cw_i)$$

Se trata de la sumatoria de las distancias de los vectores de una determinada región V_i , a su vector representativo.

A esta altura se cuenta con información para elegir; el *codeword* menos útil Cw_a , que será eliminado, y el que produce mayor contribución al error Cw_b que debe ser reforzado por otro cercano. La medida de que tan cerca estará se puede aproximar calculando la desviación standard de los elementos del sector V_b , tomando una longitud mucho menor que ésta y la dirección del vector será tomada aleatoriamente.

En resumen se ejecuta el algoritmo LBG y se obtiene un *codebook*, de este se elimina el *codeword* de menor utilidad, que será reemplazado por otro ubicado junto al de mayor error desplazado una longitud y dirección especificadas:

$$Cw_a = \arg \min_{Cw \in C} U(Cw)$$

$$Cw_b = \arg \max_{Cw \in C} E(Cw)$$

$$Cw_a = Cw_b + \left(\varepsilon \cdot \sqrt{E(Cw_b) / N} \right) u \quad \text{con} \quad 0 < \varepsilon \ll 1$$

N siempre será mayor que *std* de sector V_b

u vector n dimen. aleatorio

El proceso se itera mientras que se produzca una disminución en el error por distorsión.

En la práctica se obtiene una reducción del error de más del 10%, a costa de un incremento del tiempo de procesamiento de 3 a 7 veces. Por último la gran ventaja de éste método se pone de manifiesto al manejar datos con grandes diferencias de densidad entre grupos, en los que es capaz de mover *codewords* de forma óptima.

Utilización del cuantificador y del codebook

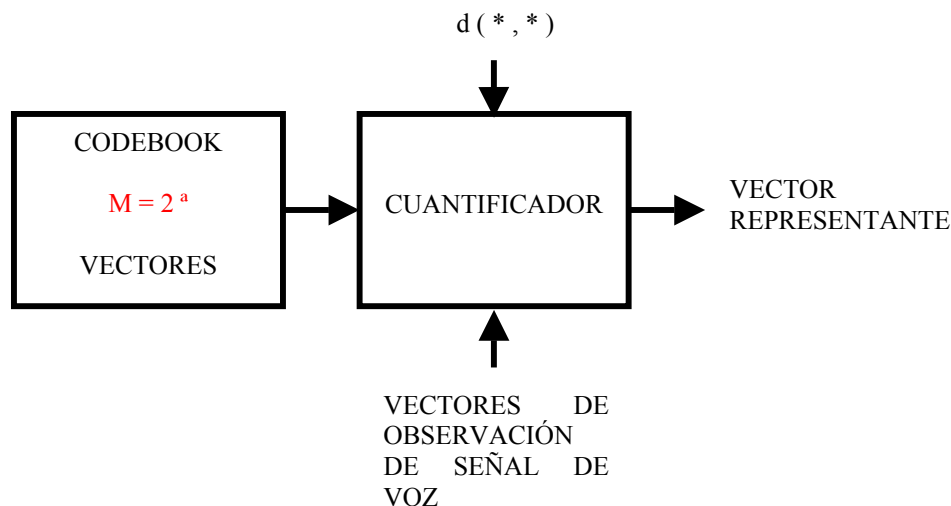
Una vez construido el *codebook*, el procedimiento para cuantificar vectores es básicamente realizar una búsqueda completa a través del *codebook* para encontrar el mejor representante.

Si anotamos los vectores del *codebook*, de tamaño M , como C_w , $1 \leq w \leq M$, y tomamos al vector de observación a ser cuantificado como V , luego el vector representante o *codeword*, V_m^* , es:

$$V_m^* = \arg \min d(V, C_w)$$

$$1 \leq w \leq M$$

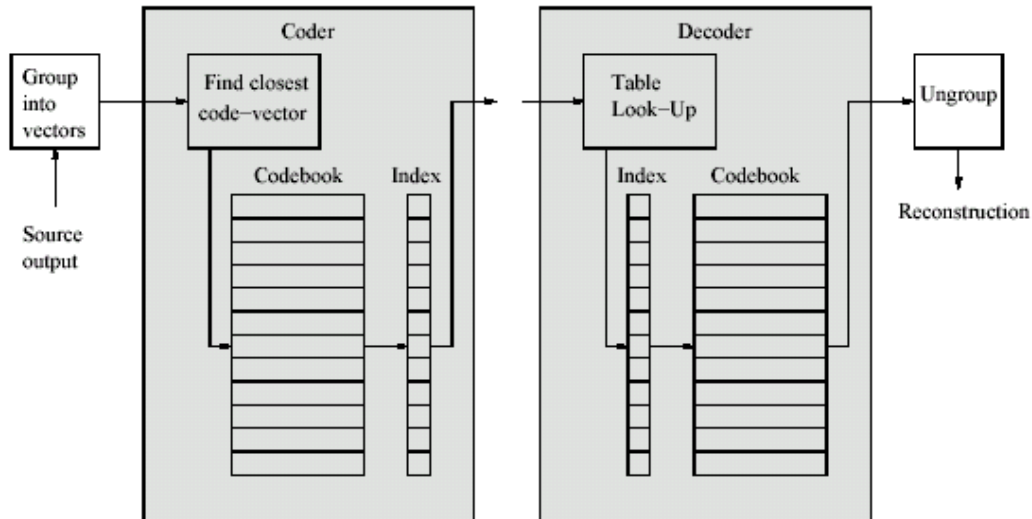
Un procedimiento de cuantificación para señal de voz elige el vector más cercano del *codebook* al vector de observación y utiliza ese vector denominado *codeword*, como la representación resultante para etapas posteriores. Se refiere como al vector "vecino" más cercano, toma como entrada, vectores de señal de voz y da como respuesta, a su salida, el vector que mejor representa esa entrada.



Aplicaciones

En General

1. Transmisión de señales y datos con el resultado de una reducción en el ancho de banda debido a que solamente se transmite un índice al vector representante, en el que se necesita un *codebook* en el lado transmisor como codificador y otro *codebook*, idéntico al primero, en la función decodificadora en el lado del receptor obteniendo a su salida, además, el error por distorsión.



Esquema de transmisión de señales o datos

- Compresión de imágenes, tomando como ejemplo la figura, vemos que se divide la imagen en rectángulos fijos, conteniendo cada uno de ellos muchos píxeles, y que van a ser representados, en la cuantificación, por el índice al vector representante o *codeword* contenido en el *codebook*. Cada sector de la imagen se transmite usando el índice correspondiente al *codeword* del *codebook*.

0	0	1	0
2	2	2	0
3	2	2	0
0	4	0	0

Imagen dividida y cuantificada

0
1
2
3
4

Índice del *codebook*

En Reconocimiento de Voz

- El uso de múltiples *codebooks* en los cuales cada *codebook* se crea separadamente (e independientemente) para cada una de las representaciones de la señal de voz (espectral o temporal). Por ejemplo se podría crear un *codebook* para las representaciones de los parámetros del *cepstrum* y otro en forma separada conteniendo las representaciones de las derivadas del *cepstrum*.
- K-tuplos cuantificadores en el cual K-tramas de señal de voz se codifican a la vez, en lugar de una única trama como es común. La idea es utilizar las correlaciones en el tiempo entre los sonidos vocales puros y los que tienen componente vocal. La desventaja ocurre cuando los sonidos donde la correlación a través del cuantificador es baja, como son los sonidos transitorios y consonantes.
- Cuantificación de matrices en las cuales se crea un *codebook* de sonidos o palabras de secuencia de longitud variable. El concepto es manejar la variación temporal vía algunos tipos de procedimientos dinámicos y por medio de eso crear un *codebook* de secuencias de vectores que representan sonidos típicos.
- Modelos ocultos de Markov en los cuales ambas reducciones, en tiempo y espectro, se usan para cuantificar la emisión completa de la voz de una manera definida y eficiente.

Conclusiones

Vimos que la idea básica del VQ es reducir la cantidad de datos a manejar a través del uso de un *codebook* con un número relativamente pequeño de *codewords*.

El éxito de este procedimiento justamente es la habilidad de representar la información espectral de la señal de voz de una forma eficiente y de una manera que conecte o relacione directamente a los vectores de observación con los fonemas acústicos.

Como toda técnica presenta una serie de ventajas importantes y algunas desventajas:

A su favor se presentan las siguientes características:

Se reduce el tratamiento de máquina en secciones posteriores. Se discretiza la representación de los sonidos de voz, por lo que se puede asociar una representación a cada *codeword* del libro de códigos.

En contra:

Una distorsión inherente al representar cada vector.

El almacenamiento necesario para el libro de códigos puede ser muy elevado al igual que el tiempo de computación que insume su creación, se debe llegar a un compromiso entre error por distorsión, almacenamiento y tiempo de procesamiento. En general se realizan para reconocimiento de voz cuantificadores de 256 sectores.

Para *codebooks* con $M \geq 1024$, el cálculo es muy tedioso y limita el procesamiento.

Modelos Ocultos de Markov (HMM)

Suponiendo salvado el inconveniente de obtener los parámetros que caracterizan a la señal hablada y su cuantificación, se tendrá como resultado un conjunto de vectores llamados normalmente de *Observación*,
 $O = \{\hat{o}_1, \hat{o}_2, \hat{o}_3, \hat{o}_4, \dots, \hat{o}_T\}$.

Se entiende así que O , fue generado por una secuencia de símbolos W aún desconocidos. La función de un reconocedor de voz que opera sobre la base de un tratamiento estadístico, debe lograr entonces obtener la secuencia de símbolos que hacen máxima la siguiente probabilidad a posteriori:

$$P(W|O)$$

Es decir la secuencia que hace máxima la probabilidad de que si se da O , provenga de W . Simbólicamente :

$$W^* = \arg \max_w (P(W \setminus O))$$

Lamentablemente, esto no puede resolverse directamente (lo que daría fin al problema), pero usando el teorema de Bayes se puede escribir:

$$P(W \setminus O) = \frac{P(O \setminus W).P(W)}{P(O)}$$

quedando:

$$W^* = \arg \max_w \left(\frac{P(O \setminus W).P(W)}{P(O)} \right)$$

Dado que con relación a la maximización requerida, $P(O)$ es constante, puede eliminarse:

$$W^* = \arg \max_w (P(O \setminus W).P(W))$$

Aún en esta situación, es impracticable el hecho de computar $P(O \setminus W)$, ya que si recordamos $O = \{\hat{o}_1, \hat{o}_2, \hat{o}_3, \hat{o}_4, \dots, \hat{o}_T\}$ y $W = (w_1, w_2, w_3, \dots, w_T)$

Es aquí donde hacen su aparición los modelos ocultos de Markov y técnicas relacionadas, que dan auxilio para una posible resolución.

Si se hace que las observaciones sean estadísticamente independientes unas de otras puede simplificarse:

$$P(O \setminus W) = \prod_{i=1}^T P(\vec{o}_i \setminus W)$$

y se asume independencia del contexto, es decir que cada observación o_i sólo depende del estado actual w_i :

$$P(O \setminus W) = \prod_{i=1}^T P(\vec{o}_i \setminus w_i)$$

Si se restringe la probabilidad de transición entre estados w_i , queda:

$$P(W) = \prod_{i=1}^T P(w_i \mid w_{i-1})$$

Y la secuencia buscada:

$$W^* = \arg \max_w \left(\prod_{i=1}^T P(\tilde{o}_i \mid w_i) \cdot P(w_i \mid w_{i-1}) \right)$$

Esta última restricción (la de transición entre estados) corresponde a restricciones impuestas por los modelos de Markov.

Uno de los principales inconvenientes al intentar resolver el problema planteado se da en la complejidad computacional de los algoritmos necesarios para hallar la secuencia buscada, los cuales involucran cálculos del orden de $2T * N^T$ (siendo N la cantidad de estados en el modelo). Así, por ejemplo, para resolver un modelo con 5 estados y 100 observaciones, se requieren $2 * 100 * 5^{100}$ cálculos (aproximadamente 10^{72}).

Este problema fundamental es resuelto utilizando las técnicas y algoritmos de resolución de modelos ocultos de Markov (HMM) como base principal en la búsqueda de las secuencias y probabilidades en cuestión.

Para entender los HMM y como resuelven el presente problema de planteo general, por medio de modelos acotados que pueden ser entrenados para maximizar la probabilidad de reconocimiento, primero debemos entender de que se trata un proceso estocástico, más precisamente un modelo de Markov, a continuación se verá un resumen, que puede servir como inicio.

Procesos estocásticos

Un proceso estocástico es un modelo matemático que describe el comportamiento de un sistema dinámico sometido a un fenómeno de naturaleza aleatoria.

Todo proceso estocástico cumple con las siguientes características:

- *Describe el comportamiento* de un sistema variante respecto de un determinado parámetro (el cual es el objeto del modelo)
- Existe un *fenómeno aleatorio* que evoluciona según un parámetro t, normalmente es el tiempo.
- El sistema presenta *estados definidos y observables*, a los cuales se les puede asociar una variable aleatoria $X(t)$ que represente una característica mensurable de los mismos.
- El sistema *cambia probabilísticamente de estado*
- Todo estado del sistema tiene una *probabilidad de estado asociada*: $p_x(t)$, la cual indica la probabilidad de encontrarse en el estado x en el instante t.

De acuerdo a lo anterior, un proceso estocástico queda definido por el conjunto:

- $X(t)$
- $P_x(t)$
- t

Un ejemplo sencillo de un proceso estocástico estaría dado por un sistema de pronóstico del clima diario, en el cual:

$t = 1, 2, 3, \dots, n$ días

$X(t)$ = pronóstico del clima asociado al día requerido (nublado, lluvioso, soleado, etc).

$P_x(t)$ = probabilidad de estado asociada.

Clasificación de los procesos estocásticos

Procesos estocásticos según la memoria de historia de estados

En este tipo de clasificación se tiene en cuenta la memoria que mantiene el proceso de la historia de los estados anteriores por los cuales atravesó.

Para efectuar un correcto análisis de la clasificación es necesario definir lo que se conoce como probabilidad condicional o de transición entre estados, elemento que, junto con el resto de los componentes de un proceso estocástico, permitirán analizar la evolución del proceso.

Formalmente, se expresa la probabilidad condicional como:

$$P\{X(t + \Delta t) = x_{t+\Delta t} \mid X(t) = x_t, X(t - \Delta t_1) = x_{t-\Delta t_1}, X(t - \Delta t_2) = x_{t-\Delta t_2}, X(t - \Delta t_3) = x_{t-\Delta t_3}, \dots\}$$

Siendo:

$x_{t+\Delta t} \rightarrow$ un estado particular en el instante $t + \Delta t$

$x_t \rightarrow$ un estado particular en el instante t

$x_{t-\Delta t_1} \rightarrow$ un estado particular en el instante $t - \Delta t_1$

etc.

Esta expresión de probabilidad condicional nos dice que se quiere informar la probabilidad de que el sistema se encuentre en el estado $x_{t+\Delta t}$ en el instante $t + \Delta t$ dado que el sistema se encontraba en el estado x_t en el instante t y en el estado $x_{t-\Delta t_1}$ en el instante $t - \Delta t_1$, y así sucesivamente.

Teniendo en mente esta expresión surgen tres tipos de procesos estocásticos distintos, los cuales varían en la cantidad de estados que toman en cuenta (que guardan en su memoria) para analizar el comportamiento futuro del sistema modelado.

a. Procesos aleatorios puros.

Un proceso aleatorio puro, es aquel en el cual la probabilidad de que el sistema se encuentre en un estado cualquiera: $x_{t+\Delta t}$ en el instante $t + \Delta t$, puede ser calculada independientemente de cuales hayan sido los estados anteriores por los cuales atravesó el modelo ($x_t, x_{t-\Delta t_1}, x_{t-\Delta t_2}, \dots$ etc)

Por lo tanto un proceso aleatorio puro, es un proceso “sin memoria” de la historia de estados anteriores.

Formalmente, se expresa esta propiedad en la forma:

$$P\{X(t + \Delta t) = x_{t+\Delta t} \mid X(t) = x_t, X(t - \Delta t_1) = x_{t-\Delta t_1}, X(t - \Delta t_2) = x_{t-\Delta t_2}, \dots\} = P\{X(t + \Delta t) = x_{t+\Delta t}\}$$

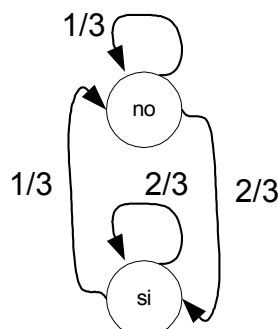
Ejemplo:

Dado un bolillero con tres bolillas: 1,2 y 3, se extraen bolillas con reposición y los resultados aleatorios definen los estados $X(t)$ del siguiente proceso:

$X(t) = \text{SI}$. Si la bolilla es 1 ó 2

$X(t) = \text{NO}$. Si la bolilla es 3

$T = 1, 2, 3, \dots$



b. Procesos sin memoria tipo Markov

Son procesos que cumplen la siguiente propiedad:

$$P\{X(t + \Delta t) = x_{t+\Delta t} \mid X(t) = x_t, X(t - \Delta t_1) = x_{t-\Delta t_1}, X(t - \Delta t_2) = x_{t-\Delta t_2}, \dots\} = P\{X(t + \Delta t) = x_{t+\Delta t} \mid X(t) = x_t\}$$

O sea, que la probabilidad de que el sistema se encuentre en un estado cualquiera: $x_{t+\Delta t}$ en el instante $t + \Delta t$, se puede calcular si se conoce cual ha sido el estado inmediatamente anterior x_t , siendo independiente de cuales hayan sido los restantes estados anteriores.

Otra forma de expresar la propiedad anterior es: dado el estado presente del modelo x_t , el futuro $x_{t+\Delta t}$ es independiente del pasado ($x_{t-\Delta t_1}, x_{t-\Delta t_2}$, etc).

Suele decirse, que un proceso de tipo Markov es un proceso sin memoria de la historia de estados anteriores excepto del inmediatamente anterior. También se los suele conocer como procesos estocásticos de “memoria uno”.

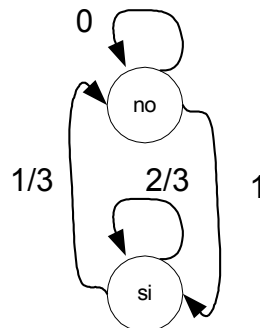
Ejemplo:

Al modelo planteado anteriormente se le hace el siguiente cambio:

$X(t) = \text{SI}$. Si la bolilla es 1 ó 2 y se reponen

$X(t) = \text{NO}$. Si la bolilla es 3 y no se repone

En este caso, dado un estado del proceso $X(t)$ se puede calcular las probabilidades de los estados $X(t+1)$, lo que convierte al experimento en un proceso de tipo Markov, según el siguiente diagrama de transiciones:

**c. Procesos con memoria**

Son todos los procesos restantes cuyas probabilidades condicionales de transición guardan historia del pasado, o de todos los estados anteriores por lo cuales pasaron. Para calcular la probabilidad de que el sistema se encuentre en un estado cualquiera $x_{t+\Delta t}$ en el instante $t + \Delta t$, un proceso con memoria requiere conocer todos los estados por los cuales atravesó el sistema.

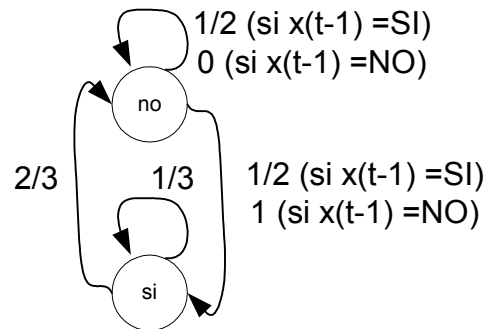
Ejemplo:

Continuando con el modelo inicial, se propone ahora el siguiente cambio:

$X(t) = \text{SI}$. Si la bolilla es 1 y se repone

$X(t) = \text{NO}$. Si la bolilla es 2 o 3 y no se repone

Ahora se tiene que, por ejemplo, para determinar la probabilidad de que el estado $X(t+1) = SI$, es necesario conocer los estados $X(t)$ y $X(t-1)$, definiendo por lo tanto un proceso con memoria, el cual es representado en el siguiente diagrama de transiciones:



Dentro de esta clasificación, son de mucha importancia para el estudio de varios de los procesos estocásticos reales (y en especial para los sistemas de reconocimiento de voz), los procesos de tipo Markov, por ello, se dedicará especial atención al análisis de los mismos.

Procesos estocásticos de tipo Markov

Un proceso estocástico, principalmente los de tipo Markov, tienen gran importancia en el análisis y estudio de sistemas con comportamiento dinámico. Este hecho se basa principalmente en que conocidos los elementos que caracterizan al sistema a modelar (que serán analizados en detalle en la siguiente sección), se pueden resolver los siguientes interrogantes:

- Cual es la probabilidad de que se produzca luego de una cantidad determinada de tiempo, una secuencia de observaciones particular?
- Cual es la probabilidad de que el sistema alcance un estado determinado luego de la evolución del parámetro t ?

Elementos de un proceso estocástico de tipo Markov

A partir de la definición de proceso estocástico y teniendo en cuenta las particularidades observadas para un proceso de tipo Markov, se determinan, para un modelo dado, los siguientes elementos:

- **Conjunto o espacio de estados del sistema:** son los diferentes estados por lo cuales el sistema transitará durante el ciclo dinámico del mismo.
- **Parámetro del sistema:** se refiere al elemento que mide la evolución del sistema. Se indica con la letra t y normalmente es el tiempo.
- **Probabilidad condicional de transición:** define la probabilidad de transición desde un estado particular i a otro estado j en un intervalo de tiempo Δt .

$$a_{ij}(\Delta t) = P\{X(t + \Delta t) = j \mid X(t) = i\}$$

El conjunto de estas probabilidades de transición, para todos los estados del modelo, definen lo que se conoce como *matriz de probabilidades de transición*, o matriz de transición, o matriz A .

- **Probabilidad incondicional de estado:** es la probabilidad de que el sistema se encuentre en el estado i en el instante t :

$$p_i(t) = p_{x=i}(t)$$

Al conjunto de probabilidades incondicionales de todos los estados de un modelo, se lo conoce con el nombre de vector de probabilidades de estado, o vector de estado.

En el caso particular en que $t=0$, se adopta el nombre de vector de estado inicial, o P_i (π):

$$\pi_i = p_{x=i}(0)$$

En todo proceso estocástico es necesario definir correctamente y conocer en forma particular los dos primeros elementos mencionados: los estados y el parámetro de evolución.

Procesos de tipo Markov según la naturaleza de las variables

Al analizar la naturaleza de las variables se hace referencia a la característica continua o discreta del espacio de estados de la variable aleatoria $X(t)$ y del parámetro t .

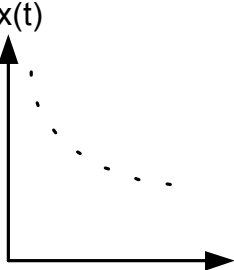
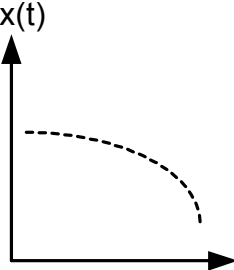
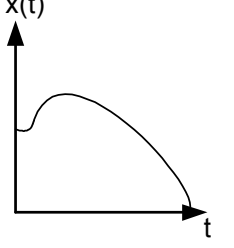
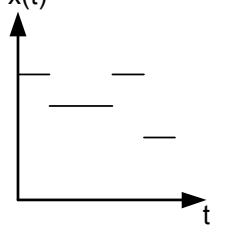
Según la naturaleza del espacio de estados, los procesos de tipo Markov pueden ser:

- Procesos de Markov:** cuando la variable $X(t)$ representa una magnitud continua (fuerza, energía, presión, etc). En este caso, el espacio de estados $X(t)$ debe ser un intervalo de números reales. Se los suele llamar también “procesos de tipo Markov con estados continuos”.
- Cadenas de Markov:** cuando la variable $X(t)$ representa una magnitud discreta (cantidad de clientes en un sistema de atención, número de líneas en servicio en un sistema de transmisión de energía eléctrica, etc). En este caso, el espacio de estados $X(t)$ es una secuencia finita o numéricamente infinita de enteros. También se los conoce como “procesos de tipo Markov con estados discretos”.

Según la naturaleza del parámetro t , los procesos de tipo Markov pueden ser:

- Procesos o Cadenas de Markov de parámetro continuo:** las observaciones al sistema se realizan en cualquier momento del continuo ($t \geq 0$).
- Procesos o Cadenas de Markov de parámetro discreto:** las observaciones al sistema se realizan en determinados instantes del parámetro t (por ejemplo: cada hora, cada minuto, cada día, etc).

En el siguiente cuadro, se resumen las clasificaciones anteriores junto con un ejemplo de las mismas:

		Naturaleza del espacio de estados $X(t)$	
		Discreto	Continuo
Naturaleza del parámetro t	Discreto	Cadenas de Markov de parámetro discreto 	Procesos de Markov de parámetro discreto 
	Continuo	Cadenas de Markov de parámetro continuo 	Procesos de Markov de parámetro continuo 

Representación de procesos de tipo Markov

Todo proceso estocástico puede ser representado gráficamente, en particular las cadenas de Markov admiten una modelización gráfica bastante intuitiva: se trata de un grafo dirigido en el cual se identifican:

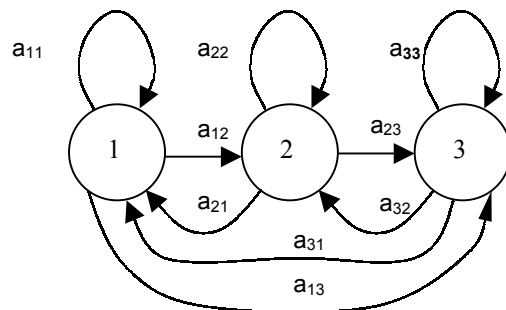
- los *estados* como los *nodos* del grafo (círculos con la etiqueta correspondiente al estado)
- las *transiciones* entre estados como las *asociaciones entre nodos* a las cuales se las etiqueta con la probabilidad de transición correspondiente.

Como ejemplo, supongamos el siguiente sistema:

3 estados diferentes y la probabilidad de transición entre cada uno de ellos es:

$$P(X_t = j \mid X_{t-1} = i) = a_{ij}$$

Tenemos entonces una máquina de estados que cambia de estado a cada determinado intervalo t . El modelo gráfico de esta situación es como el que se indica a continuación:



Propiedades y estructuras de matrices de estado

Sea A la matriz de transiciones:

$$A = \begin{bmatrix} a_{11} & \dots & a_{1N} \\ \vdots & & \vdots \\ a_{N1} & \dots & a_{NN} \end{bmatrix} \text{ con } \begin{cases} a_{ij} \geq 0 \quad \forall i, j \\ \sum_{j=1}^N a_{ij} = 1 \quad \forall i \end{cases}$$

Si A es de la forma:

$$A = \begin{bmatrix} B & 0 \\ 0 & D \end{bmatrix}$$

Con B y D submatrices cuadradas, si el sistema parte de un estado contenido en B no podrá encontrarse en uno contenido en D , y viceversa los conjuntos de estados forman dos redes desunidas, se encuentran aislados, se dice que A es *reducible* o *separable*.

Si A es de la forma:

$$A = \begin{bmatrix} B & 0 \\ C & D \end{bmatrix}$$

La probabilidad de que el sistema se encuentre en uno de los estados de D decrece monótonamente cuando aumenta el número de transiciones. El paso de D hacia B es posible pero el contrario no lo es. Se dice que los estados de D son *transitorios* y los de B *recurrentes*.
Si A es:

$$A = \begin{bmatrix} 0 & B \\ C & 0 \end{bmatrix}$$

Todas las potencias pares de A darán matrices (1) y las impares tipo (2), el sistema oscilará entre los dos subconjuntos de estados, y se dice que es un sistema *periódico*.

$$A = \begin{bmatrix} B & 0 \\ 0 & D \end{bmatrix} \quad (1)$$

$$A = \begin{bmatrix} 0 & B \\ C & 0 \end{bmatrix} \quad (2)$$

Propiedad ergódica en una cadena de Markov

Si se da que:

$$\lim_{n \rightarrow \infty} [A]^n = [\tilde{A}]$$

En la cual el segundo miembro es una matriz de transición sin elementos nulos se dice que el sistema es *ergódico* o estable en probabilidad o que posee un régimen permanente, se demuestra que si una matriz de transición no es periódica ni separable, será ergódica. En un sistema *ergódico* un estado cualquiera puede ser alcanzado por cada uno de los demás estados en un número finito de pasos.

Ejemplo de cadena de Markov

Supongamos que se quiere modelar un sistema de pronósticos del tiempo sencillo y limitado. Se considera que un día particular puede tener alguna de las siguientes características:

- puede estar nublado
- puede estar soleado
- puede llover

Si se conocen las probabilidades de que, por ejemplo, dado que un día está lluvioso, al otro día se encuentre soleado, y demás, podemos establecer un modelo de Markov para resolver preguntas como: “cual es la probabilidad de que dado que hoy está soleado, se de una observación de características climáticas particular durante el transcurso de T días”.

Según estos datos, podemos establecer los siguientes componentes del modelo:

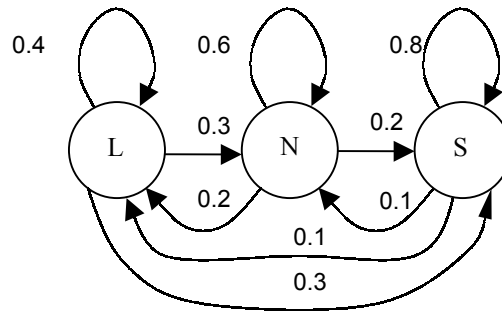
- **Estados del sistema:** { L: lluvioso; N: nublado; S: soleado }
- **Parámetro del sistema:** $t = \text{día} \rightarrow \Delta t = 1 \text{ día}$
- **Probabilidad condicional de transición:** $a_{ij}(\Delta t) =$ probabilidad de que un día particular se encuentre en estado j dado que el día anterior se encontraba en estado i . Por ejemplo, para un conjunto de datos obtenidos en forma estadística, la matriz de transiciones podría ser:

	L	N	S
L	0.4	0.3	0.3
N	0.2	0.6	0.2
S	0.1	0.1	0.8

- **Probabilidad incondicional de estado:** en el ejemplo indica la probabilidad de que un determinado día (el de comienzo de la prueba) el clima sea S, L o N:

	L	N	S
π	0.4	0.2	0.4

Gráficamente:



Para averiguar “la probabilidad de que dado que hoy está soleado, se de una observación como la siguiente: $O = \{ S, S, L, L, S, N, S \}$ ” se procede como se muestra a continuación.

$$\begin{aligned}
 P(\text{Secuencia} \setminus \text{Modelo}) &= P(S, S, L, L, S, N, S \setminus \text{Modelo}) \\
 &= \pi(S).P(S \setminus S).P(L \setminus S).P(L \setminus L).P(S \setminus L).P(N \setminus S).P(S \setminus N) \\
 &= 0.4 * 0.8 * 0.3 * 0.4 * 0.3 * 0.1 * 0.2 \\
 &= 0.0002304
 \end{aligned}$$

Si ahora deseamos calcular la probabilidad de que un estado i se presente exactamente durante d períodos tenemos:

$$\begin{aligned}
 P(\text{Secuencia} \setminus \text{Modelo}, X_1 = i) &\text{ con } \text{Secuencia} = \{X_2 = i, X_3 = i, \dots, X_d = i, X_{d+1} \neq i\} \\
 P_i(d) &= (a_{ii})^{d-1} \cdot (1 - a_{ii})
 \end{aligned}$$

Se trata de la densidad de probabilidades de duración, d en el estado i , y se distingue claramente que es del tipo exponencial, característico de una cadena de Markov.

Así, continuando con el ejemplo, si queremos calcular la probabilidad de que el clima se encuentre soleado durante 5 días, tenemos:

$$P_S(5) = (0.8)^4 \cdot (1 - 0.8) = 0.08192$$

Modelos ocultos de Markov (HMM)

Los modelos de Markov, presentados en la sección anterior son bastante restrictivos para ser aplicables a ciertos problemas de interés. Por ejemplo, en los modelos mostrados, la salida correspondiente a un determinado estado no es aleatoria.

Para salvar estas restricciones se extiende el campo de definición de los procesos aleatorios hacia los Modelos Ocultos de Markov, los cuales van un paso más allá y como principal diferencia a los modelos anteriores presentan que:

- los estados no son observables
- si bien las transiciones siguen siendo probabilísticas a_{ij} , se obtienen secuencias de salida (observaciones) que se generan en cada estado a partir de una determinada función de densidad de probabilidades $b_j(k)$.
- definen un proceso doblemente estocástico

A continuación se detallan dos modelos que ayudarán a definir y a entender los alcances de HMM:

Ejemplo 1 - Modelo del lanzamiento de las monedas

Dos personas se encuentran cada una en una habitación. Las habitaciones (A y B) son contiguas y se encuentran comunicadas por un tablero. La persona de la habitación B, arroja una o más monedas e informa a la persona de la habitación A el resultado del experimento.

De esta forma la persona en la habitación A, solo podrá ver una secuencia de “caras” –H- y “secas” –T- pero no podrá determinar en qué forma fue realizado el experimento.

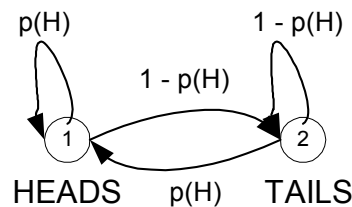
Para modelar un HMM que se corresponda con este escenario, debemos responder varias preguntas:

- Qué representarán los estados? Cuáles serán?
- Cuántos estados se tomarán?

Existen varias respuestas a estas preguntas, y cada una de ellas determina un modelo diferente.

Modelo 1

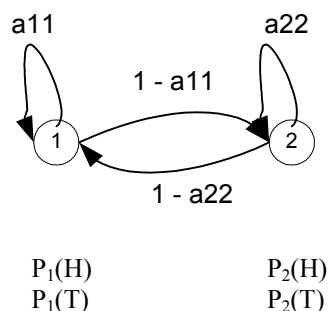
Se asume que se está lanzando una sola moneda, con lo cual se podrían modelar dos estados representando cada una de los posibles resultados de arrojarla (H, T). En este caso, el modelo descrito es una cadena de Markov, sus estados son observables y solo resta determinar cual es la probabilidad asociada a las transiciones. A continuación se muestra un diagrama de estados para la situación planteada:



En este modelo, una secuencia de observación queda determinada únicamente por las probabilidades asociadas a cada estado (o sea, la probabilidad de ocurrencia de cara o seca en un lanzamiento).

Modelo 2

Suponemos que se lanzan dos monedas, y modelamos cada una de ellas como un estado diferente del sistema. Como es lógico suponer, cada una de las monedas tendrá las probabilidades asociadas a los resultados posibles (H, T). Además, será necesario definir las probabilidades de arrojar una moneda o la otra en cada lanzamiento. Aquí estamos en presencia de un HMM con dos estados ($N=2$) y dos símbolos de observación posibles ($M=2$), que puede ser representado como:

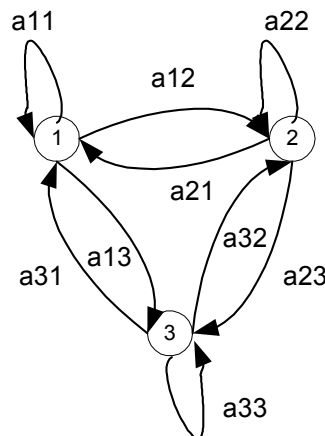


Para este modelo, una secuencia de observación particular queda determinada por:

- Cual es la moneda que se lanza (probabilidad de transición entre estados)
- La probabilidad de que el resultado obtenido para el lanzamiento de esa moneda sea cara o seca (probabilidad de observación asociada al estado).

Modelo 3

El modelo planteado anteriormente (modelo 2) puede ser extendido suponiendo que las monedas lanzadas son 3 en vez de dos, en este caso tendremos también un HMM, con variación de los parámetros respectivos: $N=3$ y $M=2$:



	1	2	3
H	$P_1(H)$	$P_2(H)$	$P_3(H)$
T	$P_1(T)$	$P_2(T)$	$P_3(T)$

En este modelo, crece la cantidad de parámetros de los que se depende para la modelización: son necesarios 6 parámetros contra 4 del modelo 2 y 1 del modelo 1.

Ejemplo 2 - Modelo de las N urnas y las M bolas de colores

Otro ejemplo práctico para comprender los HMM es el definido por el escenario siguiente:

Supongamos tener un número N de urnas en una habitación, las cuales contienen un número de bolas de colores. Asumimos que la cantidad de colores distintos es M .

Para esta situación se define un experimento con la siguiente metodología:

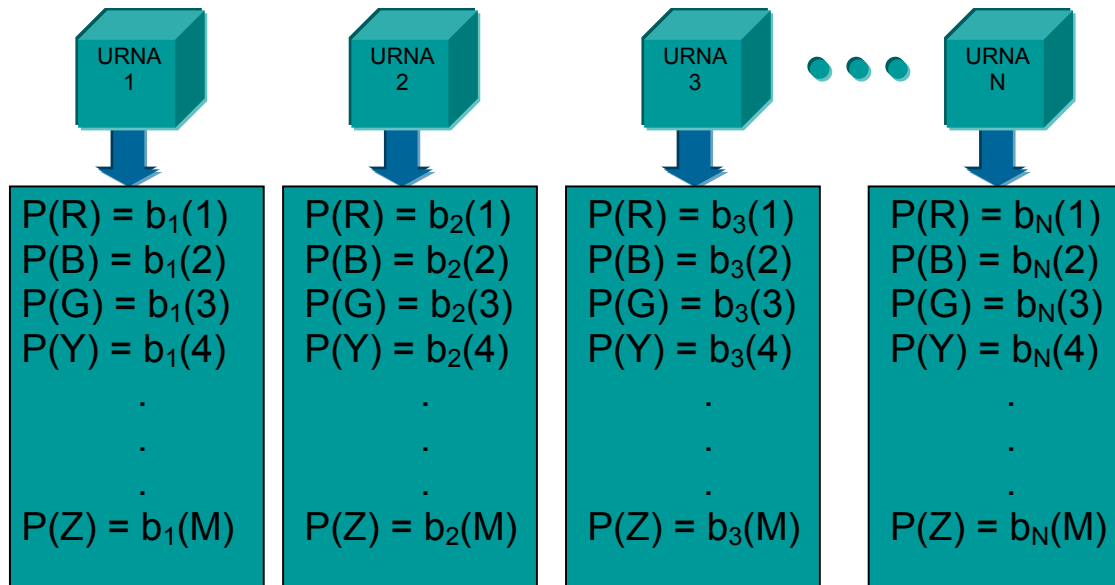
Una persona se encuentra en la habitación y en base a un proceso aleatorio selecciona una urna inicial. De esta urna elige una bola al azar y su color es grabado como una observación. La bola es colocada nuevamente en la urna. Acorde al proceso aleatorio asociado a la urna actual, se selecciona una nueva urna y se repite el proceso de selección y registración de bola.

Este proceso genera un conjunto de observaciones finito de colores, el cual es deseable modelar como la salida de un HMM.

Para la definición de este modelo se debe tener en cuenta que:

- Los estados estarán representados por las urnas.
- Existe por el proceso aleatorio de selección de urnas, probabilidades asociadas a las urnas y su selección.
- Existe por el proceso aleatorio de selección de bolas y por la cantidad de bolas de cada color existentes en cada urna, probabilidades de sacar de cada urna una bola de cada uno de los M colores existentes.

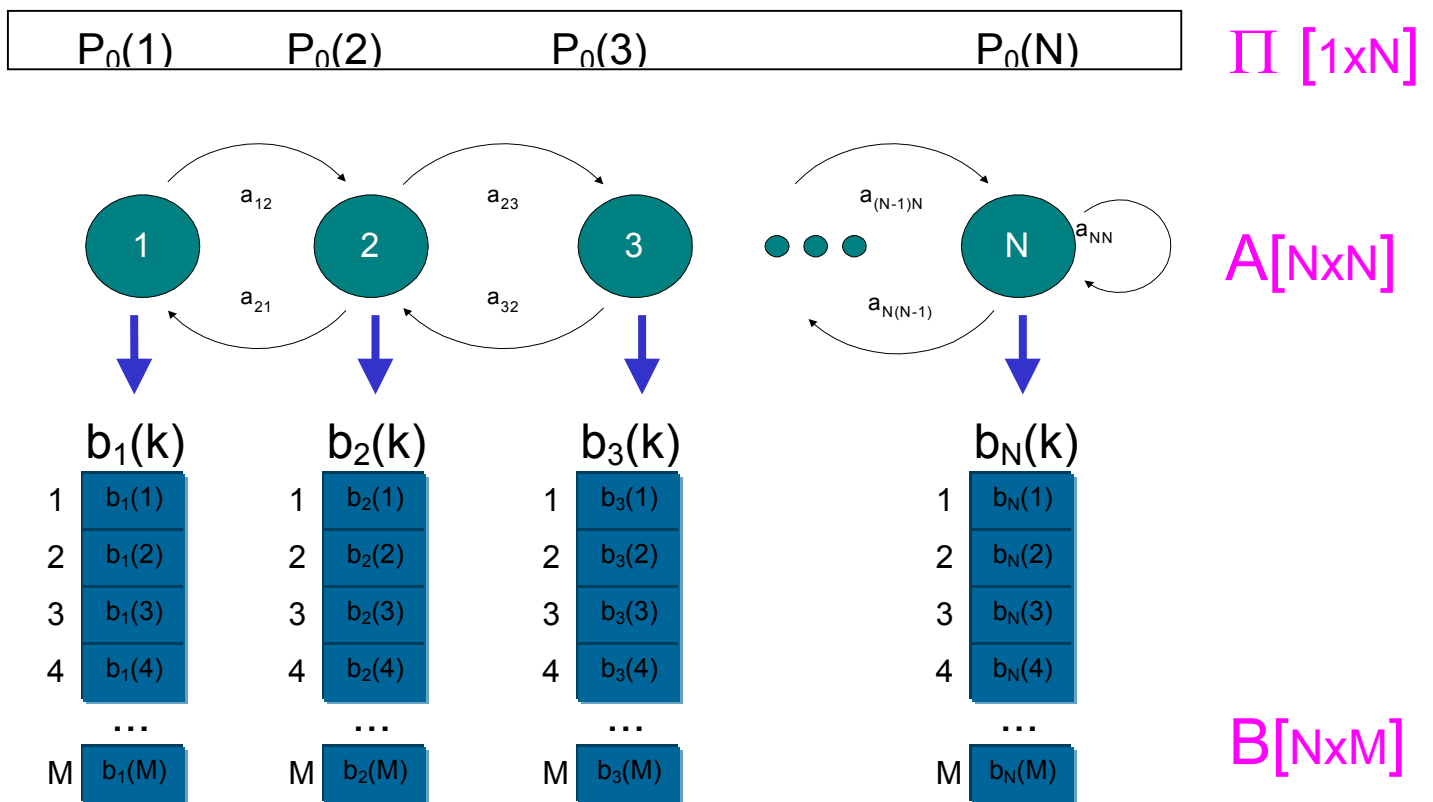
En forma gráfica el caso puede ser expresado de la siguiente forma:



$$O = \{ G, G, B, R, Y, R, \dots, Y \} / \text{Card}(O) = T$$

Elementos de un HMM

Dado un modelo HMM como el definido para el ejemplo 2, cuya representación gráfica es la siguiente:



Pueden definirse sobre el mismo los siguientes elementos, que caracterizan completamente al HMM:

- **N**: cantidad de estados del modelo. Si bien no son observables, para algunas aplicaciones suelen tener algún significado físico asociado.
- **M**: el número de símbolos de observación distintos por estado. Corresponde a la salida física del sistema modelado.

$$\lambda = \begin{cases} A = \{a_{ij}\} & 1 \leq i, j \leq N \\ a_{ij} = P(q_{t+1} = j \mid q_t = i) & \text{con } \sum_{j=1}^N a_{ij} = 1 \quad \forall i \\ B = \{b_j(k)\} & 1 \leq i \leq N, 1 \leq k \leq M \\ b_j(k) = P(o_t = k \mid q_t = j) & \text{con } \sum_{k=1}^M b_j(k) = 1 \quad \forall j \\ \pi = \{\pi_i\} & 1 \leq i \leq N \\ \pi_i = P(q_1 = i) & \text{prob. del estado inicial, con } \sum_{i=1}^N \pi_i = 1 \end{cases}$$

Siendo los elementos a_{ij} comúnmente expresados por medio de la matriz de transiciones A .

A partir de una secuencia de observaciones se puede inferir el modelo dinámico más probable λ , resultando un modelo para el proceso deseado.

En resumen, un modelo HMM, queda caracterizado en forma completa (modelo λ) al estar definidos:

$$\lambda = (A, B, \pi)$$

Por lo tanto, un HMM puede describirse como un **modelo generativo** que modela un proceso (en nuestro caso la voz) como una sucesión de estados que se conectan por transiciones.

Cada estado tiene asociada una salida de una observación, con su correspondiente distribución de probabilidades.

Cada transición está asociada a una probabilidad que la caracteriza. El modelo lleva el nombre de Markov debido a su restricción de que la probabilidad de un estado en el tiempo actual, sólo depende del estado previo, y *oculto* ya que los estados no son observables en forma directa, sino a través de sus *observaciones* que son los vectores correspondientes, característicos de la señal. La probabilidad de dicha salida modela la imposición acústica y la probabilidad de transición entre estados modela su duración.

Inconvenientes de los HMM

Como es lógico suponer, y a pesar de ser el método preferido en la actualidad por sus ventajas, los HMM tienen puntos débiles que necesitan ser destacados antes de proseguir:

- La asunción de primer orden de Markov, que dice que el proceso sólo depende de su estado actual, no es tan cierta para un proceso vocal. Como consecuencia de esto los HMM no modelan correctamente la coarticulación (Causada por las limitaciones en la velocidad de movimiento del tracto vocal). Las duraciones se modelan de forma inexacta por una exponencial en lugar de distribuciones más adecuadas como la de Poisson.
- La asunción de Independencia, que indica que no existe correlación entre frames adyacentes de entrada, también es falsa en sistemas de voz.
- Los modelos de densidad de probabilidades de salida, $b_j(k)$, tienen sus inexactitudes.
- El criterio de entrenamiento más comúnmente usado, maximización de la estimación, da una discriminación pobre entre distintos modelos acústicos, esto puede mejorarse con MMI (Maximun Mutual Information).

HMM y problemas asociados

Desde el punto de vista del modelado de segmentos o símbolos de voz, los HMM son muy versátiles, pudiéndose realizar el modelo de fonemas, palabras, y hasta frases enteras.

Provisto por el cuantificador de vectores el HMM tendrá una cantidad discreta de observaciones M y también una cantidad N de estados.

Los problemas que se deben enfrentar son:

- En primer lugar es necesario obtener el modelo $\lambda(A, B, \pi)$ para un determinado grupo de observaciones que haga óptimo el cálculo de $P(O|\lambda)$, es decir, que haga máxima la probabilidad de haber generado dichas observaciones, proceso conocido como entrenamiento del HMM.
- En segundo término calcular las probabilidades $P(O|\lambda)$, es decir que la secuencia de observaciones O , haya sido generada por un dado modelo λ .
- Por último si se tiene un grupo de observaciones O , y el modelo λ , calcular la secuencia de estados óptima, la que hace máxima $P(\text{secuencia}|O, \lambda)$.

Si se quiere encarar el problema de reconocimiento de palabras aisladas y su número no hace que computacionalmente sea irrealizable, puede calcularse el modelo (λ) de cada una de ellas, con entrenamiento adecuado, y luego al llegar al sistema un conjunto de observaciones desconocido, calcular las probabilidades $P(O|\lambda_i)$ y decidir por el modelo de mayor probabilidad.

Si el sistema es de reconocimiento continuo o el número de palabras es grande, lo anterior es imposible, ya sea el hecho de tener modelos de todas las palabras posibles como el de calcular tal número de probabilidades. Una alternativa a esto es tratar de obtener la secuencia de estados óptima, hecho que permite el ahorro de tiempo de cálculo.

Tipos de HMMs

a) Ergódico y de izquierda a derecha

Según su topología tenemos dos modelos útiles en reconocimiento de voz:

- *Ergódico*
- *de izquierda a derecha (o Bakis model)*.

El primero (*modelo ergódico*) se trata del analizado anteriormente, en el que se da el caso particular de una matriz de transiciones *ergódica* para la cual cada estado puede alcanzarse desde cualquier otro estado en una cantidad finita de pasos.

La topología *de izquierda a derecha*, tiene la característica que a medida que avanza el índice de tiempo t , el índice de estados también avanza (nunca retrocede), o queda en el mismo estado, lo que permite modelar adecuadamente señales cuyas propiedades varían con el tiempo, como la voz. De esta manera, se considera al HMM progresando de izquierda a derecha ya que la voz siempre adelanta en el tiempo, pudiendo también quedar en un loop para permitir mayor duración a cada estado individual. Las modificaciones que resultan son pocas, y tienen que ver con los estados iniciales y algunas probabilidades de transición.

Las probabilidades, para esta topología, quedan restringidas con la siguiente expresión:

$$a_{ij} = 0 \quad \forall j < i$$

Esto indica que el sistema no puede retroceder o producir una transición hacia un estado con un índice menor al actual.

Con respecto a la distribución inicial de estados, se reduce a comenzar en el primero de ellos y a finalizar en el último N , matemáticamente, tenemos:

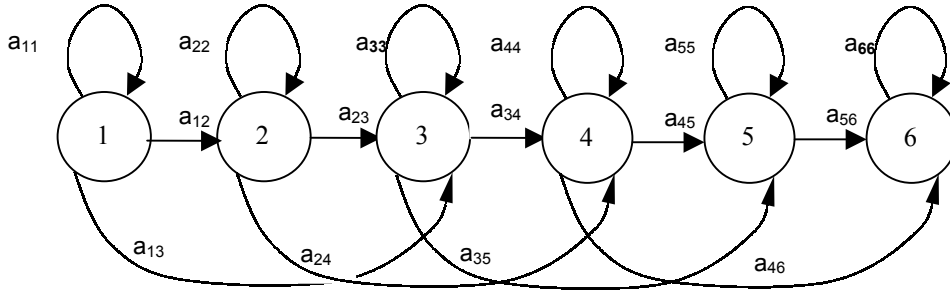
$$\pi_i = \begin{cases} 0, & i \neq 1 \\ 1, & i = 1 \end{cases}$$

Normalmente se incluye otra restricción para que no ocurran grandes saltos durante el avance de estados, y es la siguiente:

$$a_{ij} = 0 \quad j > i + \Delta, \quad \text{con } \Delta = 2$$

$$a_{NN} = 1 \text{ para su estado final}$$

De esta manera no se darán saltos mayores a 2 estados en el sistema, un gráfico característico para un HMM de 6 estados es:



Su matriz de transiciones es:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 & 0 & 0 \\ 0 & a_{22} & a_{23} & a_{24} & 0 & 0 \\ 0 & 0 & a_{33} & a_{34} & a_{35} & 0 \\ 0 & 0 & 0 & a_{44} & a_{45} & a_{46} \\ 0 & 0 & 0 & 0 & a_{55} & a_{56} \\ 0 & 0 & 0 & 0 & 0 & a_{66} \end{bmatrix}$$

Todo lo descrito para el modelo de *izquierda a derecha*, no modifica en nada lo asumido para resolver un modelo *ergódico*, ya que todas las restricciones realizadas a cualquier tipo de modelo son automáticamente respetadas en la reestimación de parámetros. Todo lo dicho hace que éste modelo sea muy utilizado en reconocimiento de voz de palabras aisladas ya que de esta forma se puede asociar el tiempo y los estados de manera más clara para poder imaginar el concepto físico de los estados por los que pasa el modelo, y luego mejorarlo.

Una ventaja de la topología de izquierda a derecha radica en que permite reducir la cantidad de transiciones entre estados, hecho que trae como consecuencia el aumento de información estadística asociada a los elementos del sistema.

b) Densidad de emisión discreta y continua

Del punto de vista de la función de densidad de probabilidades de observación, también existen variantes, en la vista hasta aquí mediante un cuantificador de vectores, se hace que dicha función sea discreta, pero una mejora significativa es tener densidades de emisión continuas ya que el proceso real del habla también es continuo.

Para esto una función de distribución muy utilizada es la suma de densidades Gaussianas multivariantes las cuales se entrenan y optimizan a través de sus diferentes parámetros de forma similar a la descripta.

$$b_j(o) = \sum_{k=1}^K c_{jk} \cdot G(o, \mu_{jk}, R_{jk})$$

En la cual:

K es el número de Gaussianas que se considerarán, (una distribución de este tipo puede aproximar arbitrariamente cerca cualquier densidad de distribución), c_{jk} es el factor de "peso" de cada Gaussiana G , que a su vez tiene medias μ_{jk} , y una matriz de covarianzas R_{jk} , debiendo ser:

$$\sum_{k=1}^K c_{jk} = 1 \quad \text{para } 1 \leq j \leq N$$

$$c_{jk} \geq 0 \quad \forall j, k$$

En este modelo de emisión continua es necesario calcular, mediante algoritmos y técnicas varias: c_{jk} , μ_{jk} y R_{jk} de forma tal que permitan hacer máxima $P(O|\lambda)$, representando un gran avance al modelo discreto, pero su inconveniente está en que cada estado debe modelar su densidad de salidas en forma independiente, lo que significa gran cantidad de parámetros a optimizar, aumentando significativamente el tiempo de entrenamiento si la cantidad de estados del sistema es grande. Por otro lado, al tratarse de funciones continuas ya no es necesaria una sección de cuantificación de vectores.

Aplicaciones de HMM

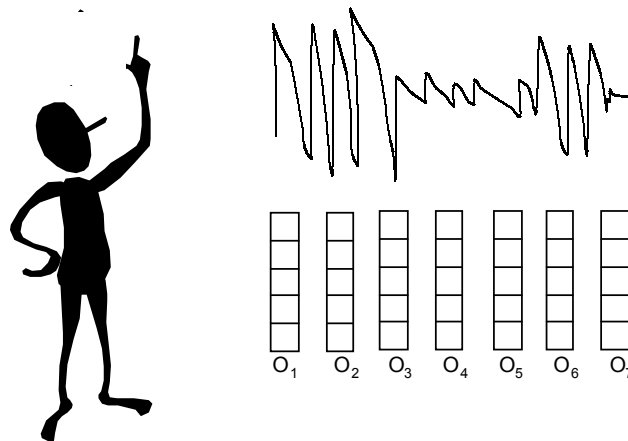
Como se dijo anteriormente, los modelos ocultos de Markov son efectivos en la caracterización de las señales de voz, en cuanto permiten modelar:

- **La duración:** mediante las probabilidades de transición entre estados
- **La variación acústica:** mediante las probabilidades de emisión asociadas a cada estado.

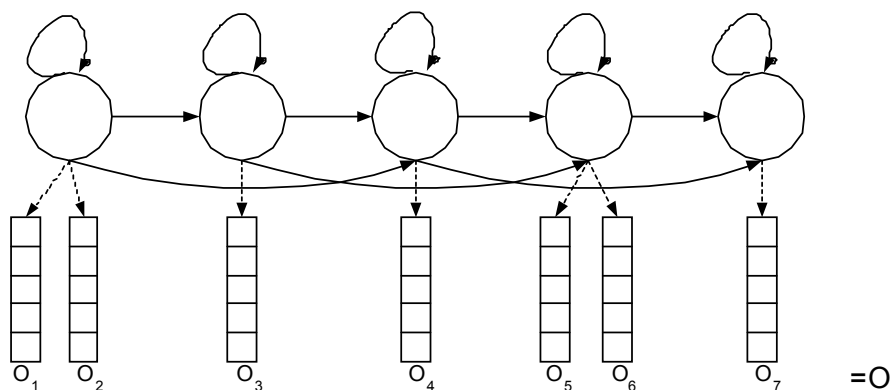
Como se dijo también anteriormente, los HMM deben ser vistos como modelos generativos de las características vocales.

Podemos decir que por cada sonido emitido por una persona, se puede construir un HMM que sea capaz de generar los mismos sonidos. Si bien este proceso no es sencillo, podemos describirlo como se indica a continuación:

- una persona emite una palabra, la cual es caracterizada por una señal de voz que puede ser descompuesta en una secuencia de observaciones características de la señal emitida.



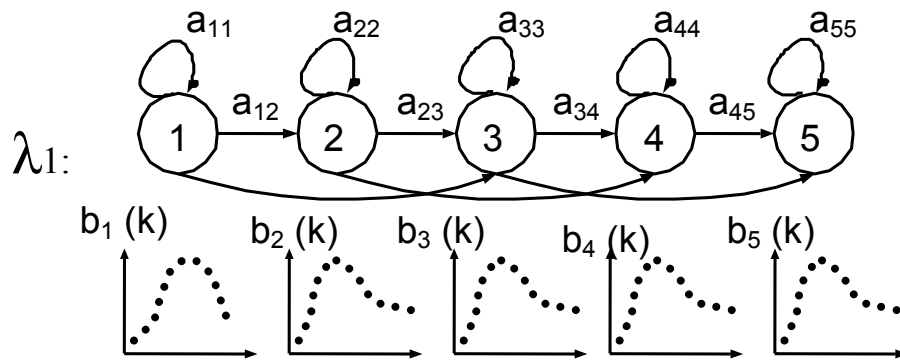
- Por analogía, se puede obtener un modelo oculto de Markov, que genere tras un ciclo determinado de tiempo, una secuencia de observaciones idéntica a la emitida por el ser humano.



Ahora bien, esta analogía puede ser generada efectivamente, pero recordando las definiciones del modelo descritas en las secciones anteriores, se concluye que, para obtener dicho modelo es necesario caracterizarlo completamente, es decir, se deben obtener:

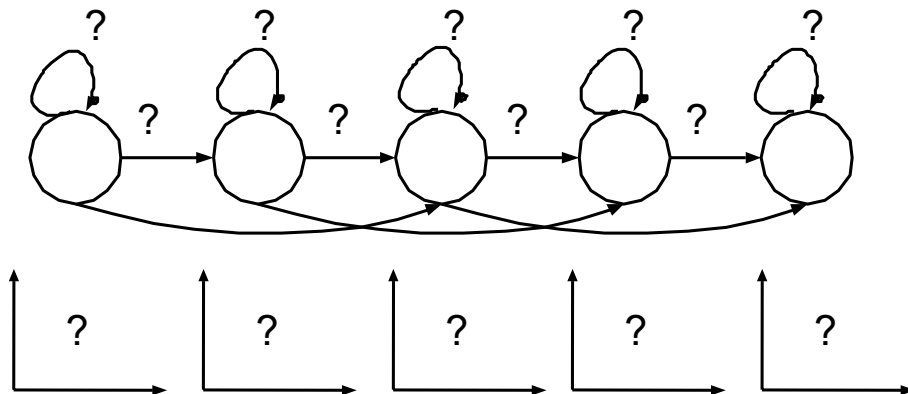
- A
- B
- π

Para definir el modelo λ_1



Es justamente la obtención de este modelo, el principal problema que se debe enfrentar al intentar implementar un modelo HMM generativo de una secuencia de observaciones.

Para realizar esta tarea se debe tener en cuenta que, inicialmente tendremos un modelo HMM completamente indefinido, el cual se vería gráficamente como:



Y se debería completar cada uno de los signos de pregunta. El interrogante es cómo?

Se debe partir de una secuencia de observaciones inicial y mediante algoritmos determinados generar los valores que cubran los signos de pregunta anteriores, o sea, que será utilizada para la obtención del modelo más probable para generarlas. Esto se logra mediante el proceso de entrenamiento del modelo, el cual se describe en la siguiente sección.

Entrenamiento de HMM

El proceso de entrenamiento de un modelo HMM, utiliza los algoritmos necesarios para realizar el cálculo o estimación de los parámetros que definen al modelo.

Se trata básicamente de un proceso iterativo que maximiza en forma local la probabilidad de que una secuencia de observación haya sido generada por un modelo particular [$P(O|\lambda)$] y que garantiza en cierta forma la convergencia del proceso a partir de un modelo inicial aleatorio.

Uno de los métodos más conocidos para realizar esta tarea es la técnica de **Maximización de la Estimación**, y como una especialización de la misma, el **algoritmo de Baum – Welch**. Este último define los parámetros de un HMM como se muestra a continuación:

- Probabilidades de transición de estados (Matriz A)

$$a_{ij} = \frac{\text{Número esperado de transiciones de } i \text{ a } j}{\text{Nro esperado de transiciones desde } i}$$

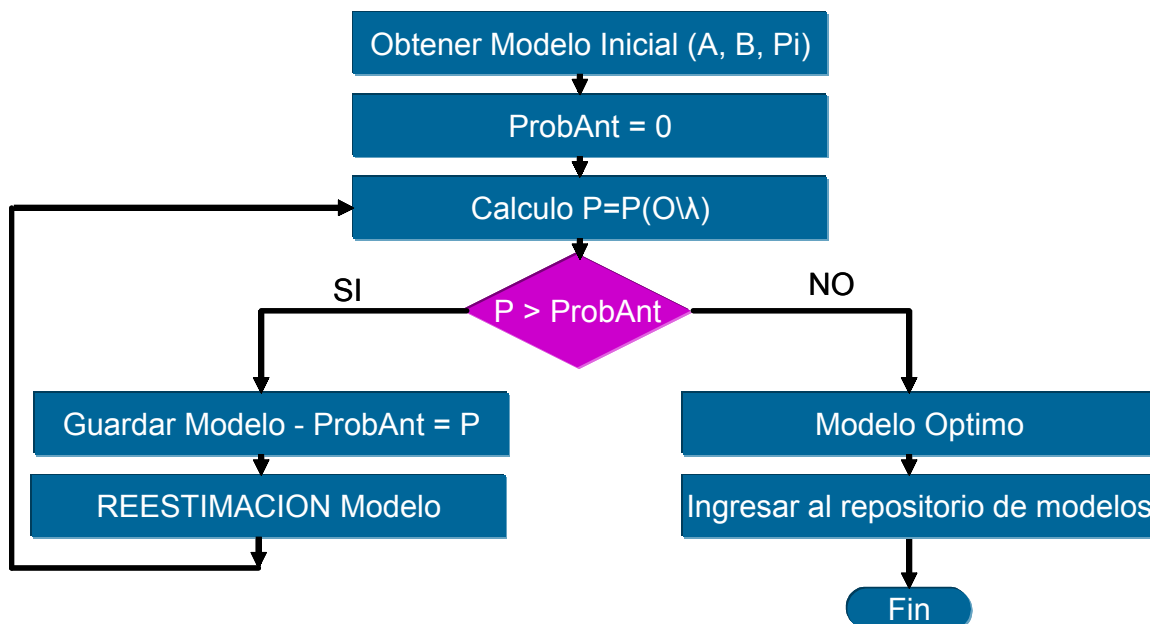
- Probabilidades de emisión (Matriz B)

$$b_{j(k)} = \frac{\text{Número esperado de veces en el estado } j \text{ con el símbolo } v_k \text{ observado}}{\text{Nro de veces en el estado } j}$$

- Vector de probabilidad inicial

$$\Pi_i = \text{probabilidad de iniciar en estado } i$$

Entonces, dada una secuencia muestral, se busca, mediante el método de maximización de la estimación, obtener el modelo HMM que tenga más probabilidad de generar la secuencia indicada utilizando las fórmulas expuestas y el algoritmo general que se indica a continuación:



- **Obtención del modelo inicial:** se obtiene en forma totalmente aleatoria, sujeto como es de suponer a las restricciones de probabilidades comunes. Existe la posibilidad de implementar mejoras que ayuden a obtener parámetros iniciales más exactos. Cabe aclarar que cuanto más exacto o cercano al máximo global se encuentre el modelo inicial, más exacto será el modelo final obtenido.
- **Cálculo de P:** la probabilidad de que la observación haya sido generada por el modelo obtenido es calculada con la ayuda de algoritmos intermedios auxiliares que permiten reducir la complejidad computacional del proceso (algoritmos forward y backward). Esta probabilidad es calculada por cada uno de los modelos obtenidos hasta verificar que la misma es máxima.

- **Condiciona $P > Prob_{Ant}$** : define el ciclo iterativo del algoritmo, esto es, se recalcularán los parámetros del modelo hasta que la probabilidad obtenida para uno de ellos sea menor que la del modelo anterior. Esto indica que se ha alcanzado un máximo local para el modelo inmediatamente anterior al actual.
- **Reestimación del modelo**: se trata de recalculer los parámetros del modelo utilizando las fórmulas anteriores, basándose para ello en el modelo obtenido en la iteración anterior.
- **Modelo óptimo**: una vez alcanzada la máxima probabilidad, se está en presencia del modelo óptimo el cual debe ser guardado para su utilización posterior en lo que se da a llamar el **repositorio de modelos**

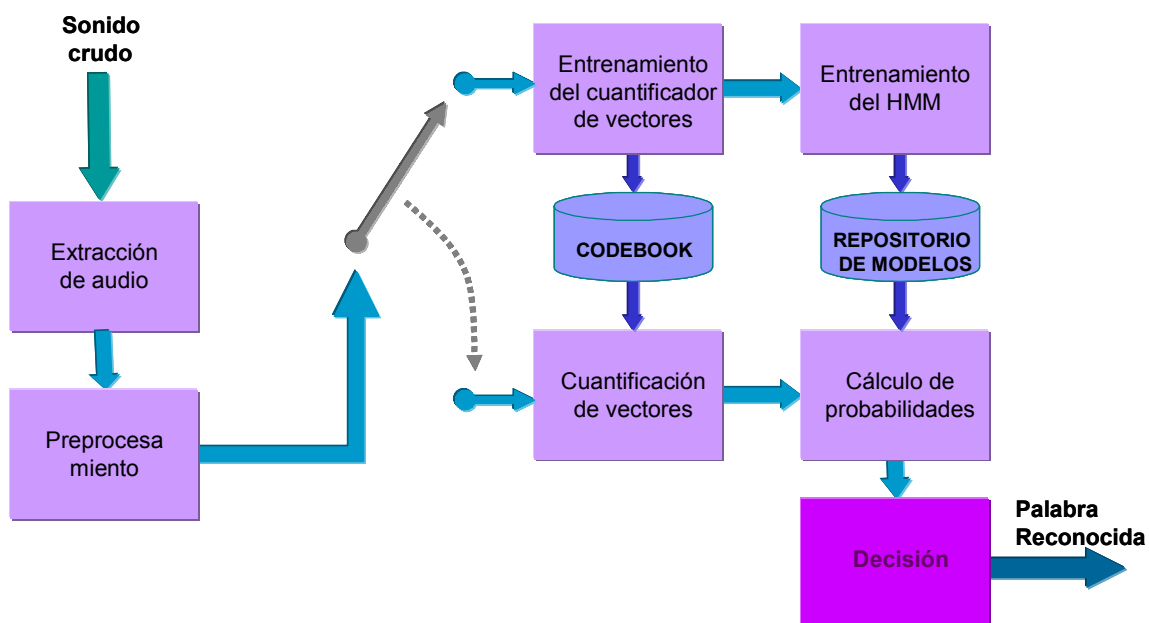
En resumen, el algoritmo de maximización de la estimación propone obtener en forma iterativa modelos λ_n para una secuencia de observaciones muestra, e ir comparando las probabilidades de generación de los mismos (las cuales, como es de suponer, son crecientes hasta alcanzar el máximo) hasta detectar que se llegó a un máximo local para dicha probabilidad. Alcanzado este máximo (paso garantizado por la convergencia del método) se toma al modelo final como el que más probablemente pueda generar la secuencia de observaciones caracterizada por la muestra.

Habiendo generado un modelo óptimo para una secuencia de observaciones dada, solo resta determinar cómo interactuar para obtener un reconocedor de palabras aisladas a partir de lo expresado.

Visión Global del proceso de entrenamiento/reconocimiento utilizando HMM

Interacción entre módulos

Para obtener una idea clara del proceso de entrenamiento/reconocimiento de palabras aisladas utilizando HMM se debe tener presente como interactúan y comunican los módulos que componen el modelo y que aplican los conceptos vistos previamente en este apunte. A continuación se muestra un diagrama en bloques del proceso general de reconocimiento y entrenamiento de palabras.



Como entrada, tanto el proceso de entrenamiento como el de reconocimiento, recibirán una señal de audio. La diferencia entre la fase de entrenamiento y la de reconocimiento en cuanto a la entrada se encuentra dada por la cantidad de señales que se recibirán:

- En el entrenamiento se ingresan una o más muestras de una palabra (varias señales) de entrada, que serán utilizadas para construir los modelos necesarios para su posterior reconocimiento.
- En el reconocimiento se ingresa una palabra (una señal), la que es evaluada para determinar el modelo que más probablemente haya podido generarla.

Para poder trabajar con la señal de audio pura (cruda), primero es necesario ejecutar dos procesos sobre la misma para lograr caracterizarla. Estos dos procesos son comunes tanto para el reconocimiento como para el entrenamiento.

- **Módulo extracción de audio:** Realiza la separación o extracción de la señal propia de audio, de la señal de ruido general que no pertenece a la misma, puede adaptarse dinámicamente a variaciones de los niveles de ruido.
- **Módulo Preprocesamiento:** Obtiene las mejores características descriptivas del proceso de la señal de voz humana y trata de reducir la cantidad de información a manipular.

Una vez preprocesada la/s señal/es de entrada, se puede utilizar la misma para entrenar o para reconocer, siendo distintos los procesos que se ejecutan sobre los vectores de observación correspondientes dependiendo de la acción requerida. A continuación se indican los módulos necesarios para ambas acciones:

Entrenamiento

- **Módulo de entrenamiento del cuantificador de vectores:** Entrena el cuantificador de vectores de manera no supervisada.
- **Módulo de entrenamiento del HMM:** determina el modelo $\lambda (A, B, P_i)$ que caracteriza a la palabra entrenada en base a ciertos parámetros N y M preestablecidos. Para obtener el modelo utiliza el algoritmo de maximización de la estimación, mediante el método de Baun-Welch. Una vez obtenido el modelo, lo ingresa en un repositorio común del sistema, junto con la palabra entrenada (establece la relación modelo-palabra).

Reconocimiento

- **Módulo Cuantificador de vectores:** Se encarga, una vez entrenado de agrupar, reducir y discretizar la cantidad de información, logrando asociar grupos de vectores a un sector determinado que será representativo.
- **Módulo Calculador de probabilidades:** obtiene del repositorio común cada uno de los modelos λ , calculando la probabilidad de que cada uno de los ellos haya podido generar la palabra de entrada: $P(\text{palabra} | \lambda_x)$.
- **Módulo de decisión:** compara las probabilidades obtenidas por el módulo calculador y obtiene la máxima. De esta forma asocia el modelo a la palabra a reconocer. Finalmente obtiene del repositorio la palabra asociada al modelo de mayor probabilidad e informa dicha palabra como palabra reconocida (asociando de esta forma la palabra a reconocer –de entrada- con la palabra reconocida -del repositorio-).

Como se puede apreciar, el conjunto de módulos anteriores definen en forma íntegra los procesos de entrenamiento y reconocimiento de palabras aisladas mediante la utilización de los algoritmos, técnicas y procedimientos enunciados en las secciones anteriores del apunte.

Operación general de un reconocedor de palabras aisladas

Ahora que se tienen definidos los módulos que componen al entrenador/reconocedor de voz, se explicará en grandes rasgos la forma de operar del mismo para un conjunto limitado de palabras aisladas.

En forma previa al reconocimiento de una palabra determinada, se deberá **entrenar** al reconocedor para que aprenda o caracterice a dicha palabra. Para ello, se deberá realizar el proceso de entrenamiento para cada una de las palabras que conforme el conjunto de vocablos que se deseen reconocer (previamente las mismas deben ser preprocesadas para caracterizarlas correctamente). Por ejemplo, si se quiere reconocer el conjunto de palabras de control = { atrás, adelante, izquierda, derecha } se deberá realizar el proceso de entrenamiento para cada una de estas palabras.

Es importante aclarar que el entrenamiento puede requerir una muestra por cada palabra o varias, definiendo lo que se conoce como entrenamiento simple observación o entrenamiento multiobservación respectivamente.

Este proceso de entrenamiento en forma general, determina el modelo de Markov óptimo (mediante el proceso de maximización de la estimación) para cada una de las palabras. Por lo tanto se obtendrá un modelo λ para cada una de las palabras entrenadas, los cuales serán guardados en forma permanente en un repositorio de modelos para su posterior

utilización en el proceso de reconocimiento. Volviendo al ejemplo de las palabras de control, el entrenamiento de las mismas generaría cuatro modelos λ :

- ✓ $\lambda(\text{adelante})$
- ✓ $\lambda(\text{atrás})$
- ✓ $\lambda(\text{izquierda})$
- ✓ $\lambda(\text{derecha})$

En el repositorio de modelos se generarían cada uno de estos modelos asociados a las palabras correspondientes.

Además de los modelos λ (correspondientes a HMM), el entrenamiento genera el codebook necesario para las palabras a reconocer, el cual será guardado en forma permanente en el repositorio de modelos. Para nuestro ejemplo se generará:

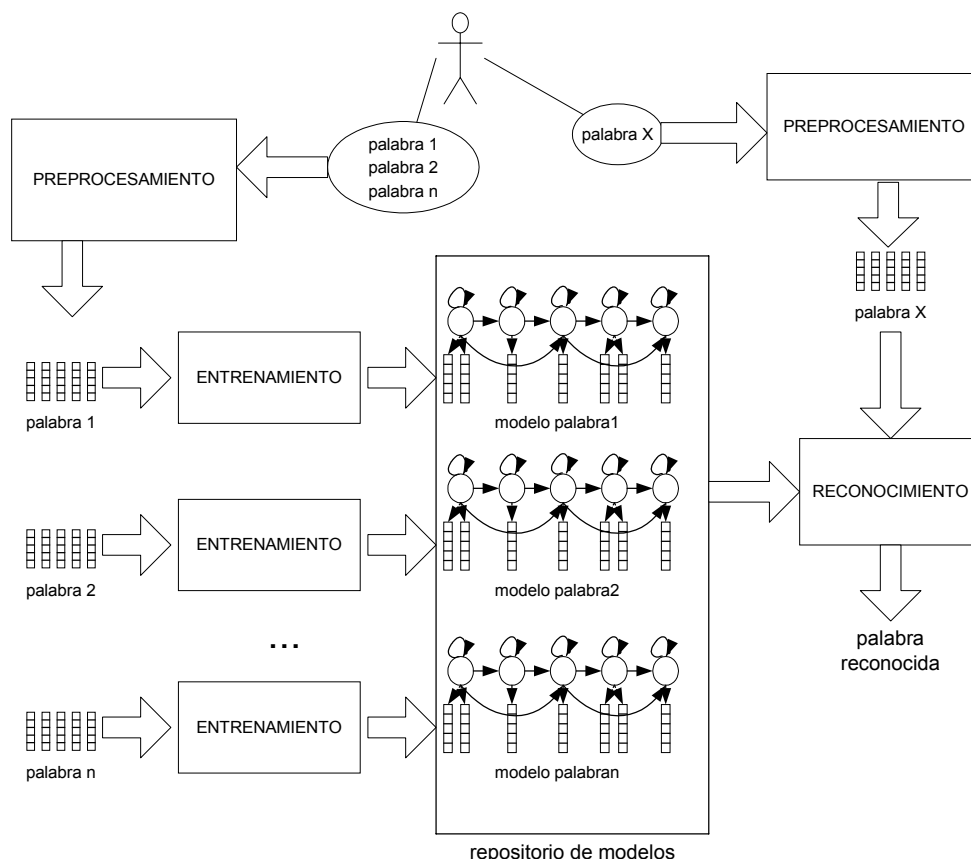
- ✓ **codebook(adelante, atrás, izquierda, derecha)**

De esta forma, habiendo obtenido ambos modelos (codebook y HMM) finaliza el proceso de entrenamiento de palabras y es posible que las mismas sean reconocidas.

El proceso de reconocimiento recibe como entrada una palabra (la que se quiere reconocer), la cual es preprocesada con el objetivo de caracterizarla en forma adecuada. Una vez hecho esto, se está en condiciones de reconocerla, para ello el proceso recuperará los modelos HMM existentes en el repositorio común y calculará cual es la probabilidad de que cada uno de los modelos haya generado esa palabra: $P(\text{palabra} | \lambda_x)$. Para el ejemplo planteado, el reconocedor obtendrá:

- ✓ $P(\text{palabra} | \lambda(\text{adelante}))$
- ✓ $P(\text{palabra} | \lambda(\text{atrás}))$
- ✓ $P(\text{palabra} | \lambda(\text{izquierda}))$
- ✓ $P(\text{palabra} | \lambda(\text{derecha}))$

Finalmente, una vez obtenidas todas las probabilidades asociadas se toma la decisión de cual es el modelo más probable, y esto se hace obteniendo la máxima probabilidad calculada. Habiendo detectado cual es el modelo que más probablemente haya generado la palabra de entrada y sabiendo cual es la palabra asociada a dicho modelo, el sistema puede determinar cual es la palabra reconocida. Siguiendo con el ejemplo, si la máxima probabilidad obtenida es: $P(\text{palabra} | \lambda(\text{derecha}))$, el reconocedor determinaría a partir del repositorio que la palabra de entrada es **derecha**. Se muestra a continuación un diagrama general que define el proceso de entrenamiento/reconocimiento descrito:



Para que este proceso sea adecuado y se obtenga un error de reconocimiento mínimo se deben tener en cuenta varios puntos:

Es muy importante la estimación inicial de los parámetros de los modelos (existen técnicas varias que ayudan a una mejor estimación de A, B y P_i).

Es conveniente realizar entrenamientos multiobservación para extraer mayor información de caracterización de cada palabra.

Se deben implementar técnicas de escalamiento para evitar que los cálculos produzcan un underflow (dado que se trabaja por lo general con números muy pequeños cercanos a cero).

Se debe entrenar con conjuntos de datos / palabras representativos y suficientes para la correcta estimación de parámetros.

Se deben seleccionar determinados parámetros (N y M sobre todo) en base a un compromiso entre porcentual o margen de error y tiempos de entrenamiento, dado que ambos factores suelen tener características opuestas.

Es recomendable definir un modelo de “ruido” para caracterizar palabras no reconocidas o en su defecto establecer un margen de probabilidad de reconocimiento el cual si no es alcanzado por ningún modelo del repositorio para una palabra de entrada dada, indica que la misma no puede ser reconocida por los modelos existentes.

Bibliografía

- Métodos y Modelos de Investigación de operaciones. Vol. 2 Dr. Juan Prawda Witemberg. Editorial México Limusa. 1993
- Elements of the theory of Markov processes and their applications. A.T. Barucha Reid. McGraw Hill. 1960.
- Investigación Operativa – Apuntes de la materia – Marta Poiassina
- Procesos estocásticos- Cadenas de Markov - Rojo
- Speech Analysis course notes. Dr. Tony Robinson of the Engineering Dept of Cambridge University. <http://svr-www.eng.cam.ac.uk/~ajr/SA95/>.
- The LBG-U method for vector quantization an improvement over LBG inspired from neural networks. Bernd Fritzke, Ruhr-Universität Bochum - Institut für Neuroinformatik Internal Report January 1997. Kluwer Academic Publishers.
- Redes de Computadoras. Tercera Edición. Andrew S. Tanenbaum. Prentice Hall. 1997
- Markov Models for Secuential Data. Yoshua Bengio, Neural Computing Surveys 2, 129-162, 1999.
- A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. L.R. Rabiner, Proceedings of the IEEE, Vol 77, N° 2 February 1989: pp. 257 - 286.
- Fundamentals of Speech Recognition L.R. Rabiner, Biing-Hwang Juang, Publisher: Pearson Education POD; 1 edition (April 12, 1993) ISBN: 0130151572