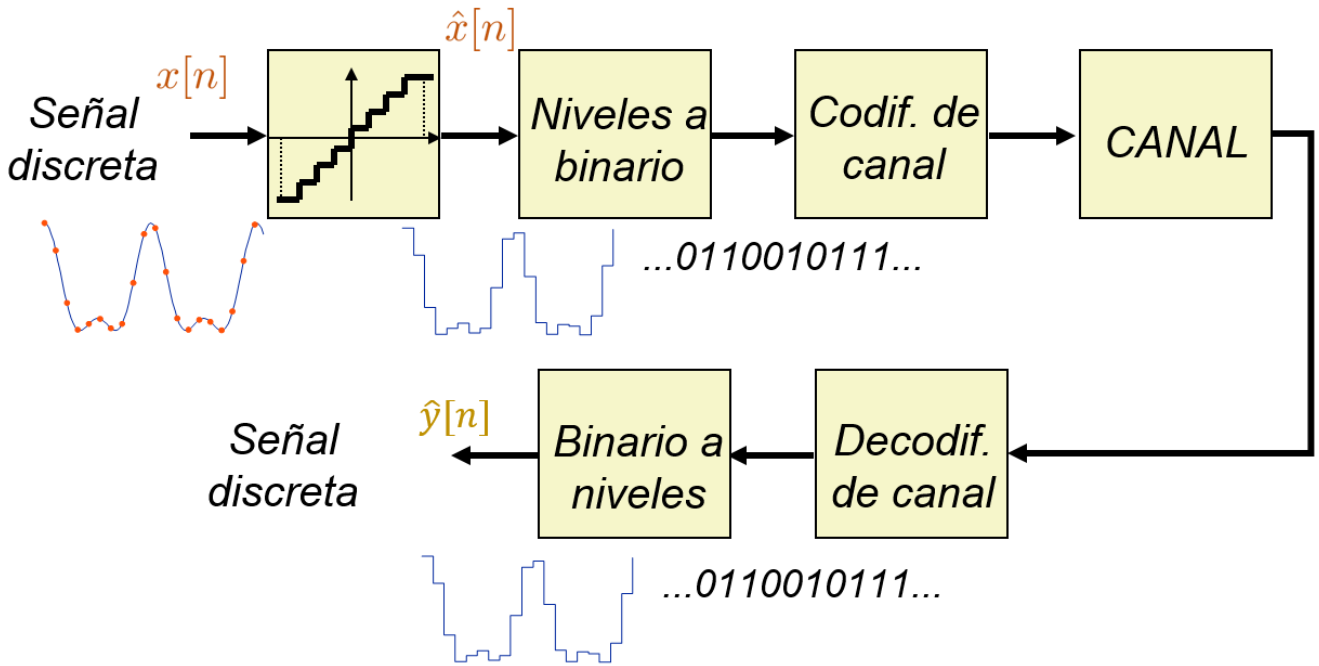
Introducción a la cuantificación escalar y vectorial

Procesado de sonido

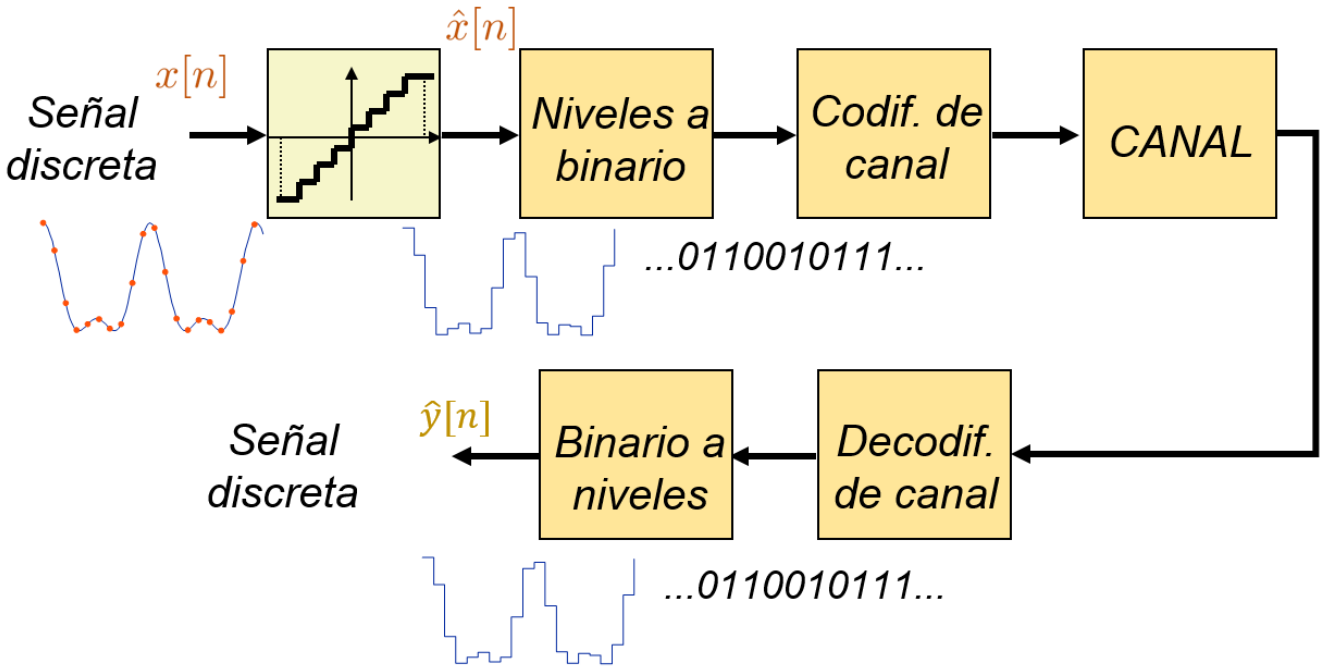
Universidad de Vigo

Régimen binario



- Cada muestra se codifica con n bits.
- Cada $T_s = 1/f_s$ segundos se envían n bits.
- El régimen binario en el canal se calcula como $R_b = n \cdot f_s$.
- Ejemplo: con $f_s = 8$ *kHz* y $n = 8$, $R_b = 8 \cdot 8000 = 64$ *kbps*.

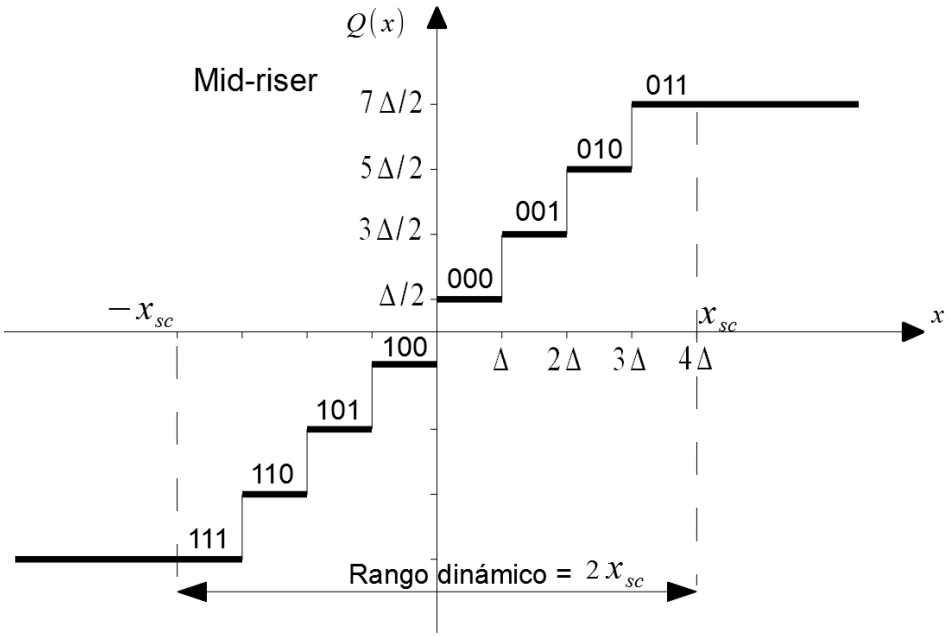
Régimen binario



- En ausencia de errores de canal $\hat{y}[n] = \hat{x}[n]$.
- En la simulación normalmente obviaremos los bloques en naranja.

Cuantificación uniforme (mid-riser)[1]

- Discretiza el eje de amplitudes.
- Caracterizado por:
  - n : número de bits (2^n niveles de cuantificación).
  - x_{sc} : valor de sobrecarga del cuantificador.
  - $\Delta = \frac{2x_{sc}}{2^n}$: escalón de cuantificación

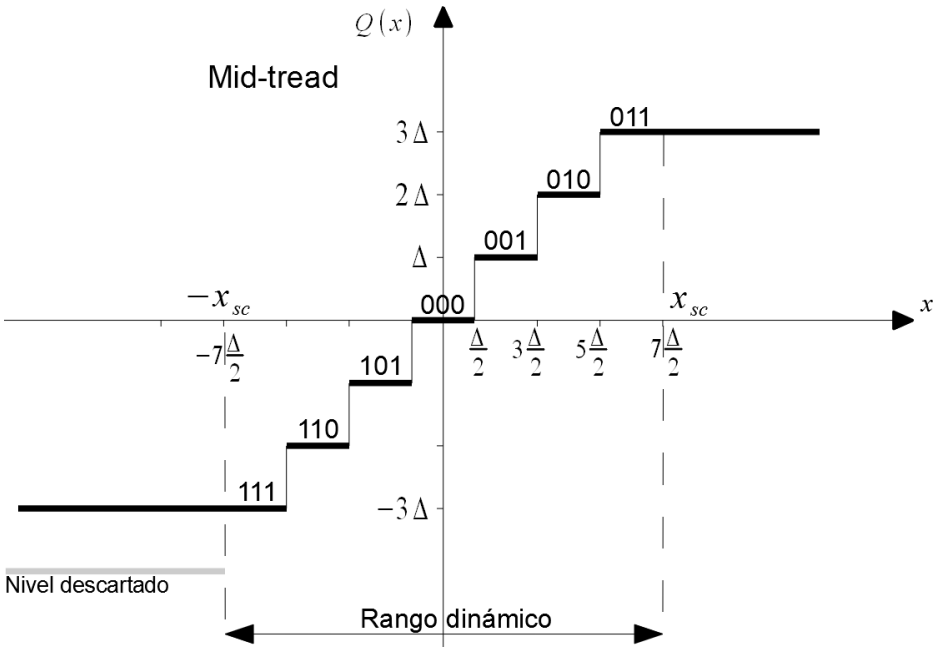


- Error de cuantificación: $e[n] = x[n] - \hat{x}[n]$
  - Menor cuanto mayor sea n
  - Menor cuanto menor sea el escalon de cuantificación.
  - En ausencia de error de sobrecarga, $e[n]$ puede aproximarse por una v.a. uniforme en el intervalo $(-\Delta/2, \Delta/2)$, siendo su potencia $\Delta^2/12$.

[1] Leer apuntes tema 3, apartado 5.1.1

Cuantificación uniforme (mid-tread)[2] [3]

- Discretiza el eje de amplitudes.
- Caracterizado por:
  - n : número de bits ($2^n - 1$ niveles de cuantificación)
  - x_{sc} : valor de sobrecarga del cuantificador.
  - $\Delta = \frac{2x_{sc}}{2^n - 1}$: escalón de cuantificación



- Error de cuantificación: $e[n] = x[n] - \hat{x}[n]$
  - Menor cuanto mayor sea n
  - Menor cuanto menor sea el escalon de cuantificación.
  - En ausencia de error de sobrecarga, $e[n]$ puede aproximarse por una v.a. uniforme en el intervalo $(-\Delta/2, \Delta/2)$, siendo su potencia $\Delta^2/12$.

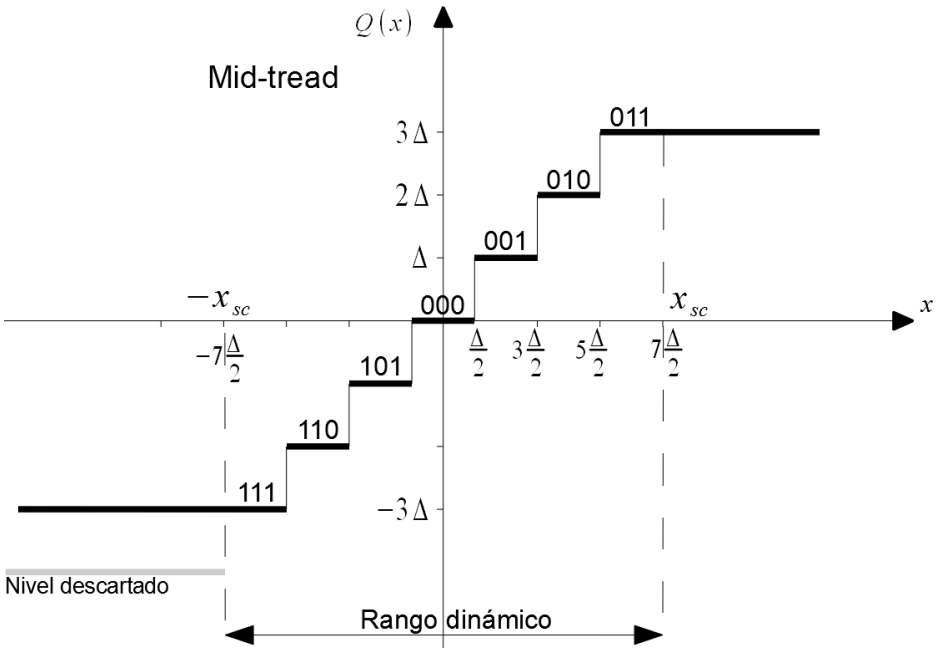
[2] Leer apuntes tema 3, apartado 5.1.1.

[3] En este trabajo, por sencillez, descartaremos un nivel de cuantificación. En caso contrario el cuantificador no sería simétrico.

Cuantificación uniforme (mid-tread)

Ejemplo mid-tread 3 bits

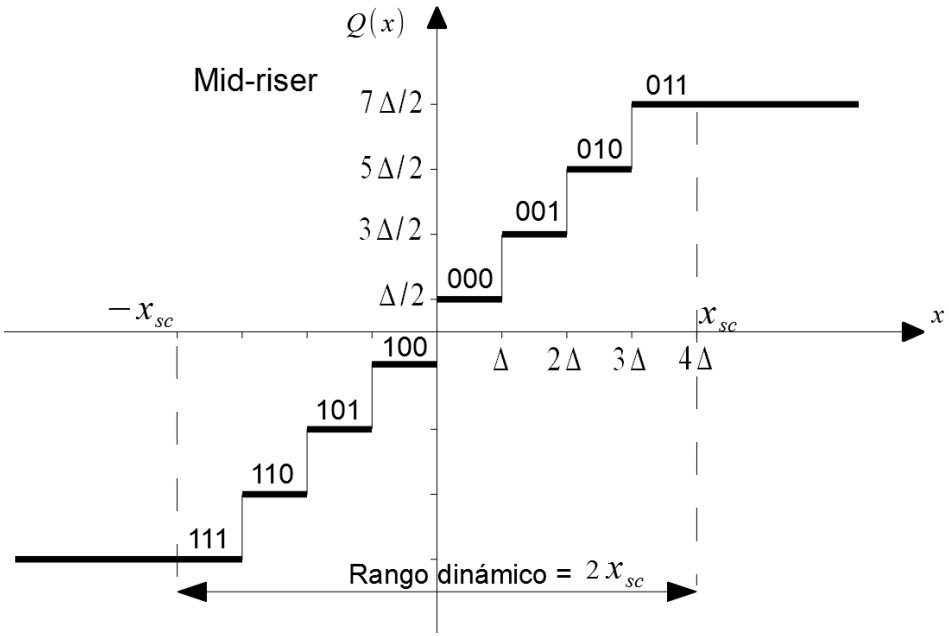
- Niveles de cuantificación: $L = 2^3 - 1 = 7$
- Si $x_{sc} = 1$, el escalón es: $\Delta = 2x_{sc}/7 \approx 0{,}29$
- El máximo valor cuantificado es $3\Delta \approx 0{,}857$
- El mínimo valor cuantificado es -3Δ
- Error de sobrecarga si $|x| > x_{sc}$.



Tarea 1: Cuantificación uniforme en Matlab

Comprender la función:
[y, e]=qmidriser(x,xsc,n)

- x : vector de muestras de entrada (longitud arbitraria)
- xsc : valor de sobrecarga del cuantificador
- n : número de bits
- y : vector con valores cuantificados
- e : error de cuantificación , x-y

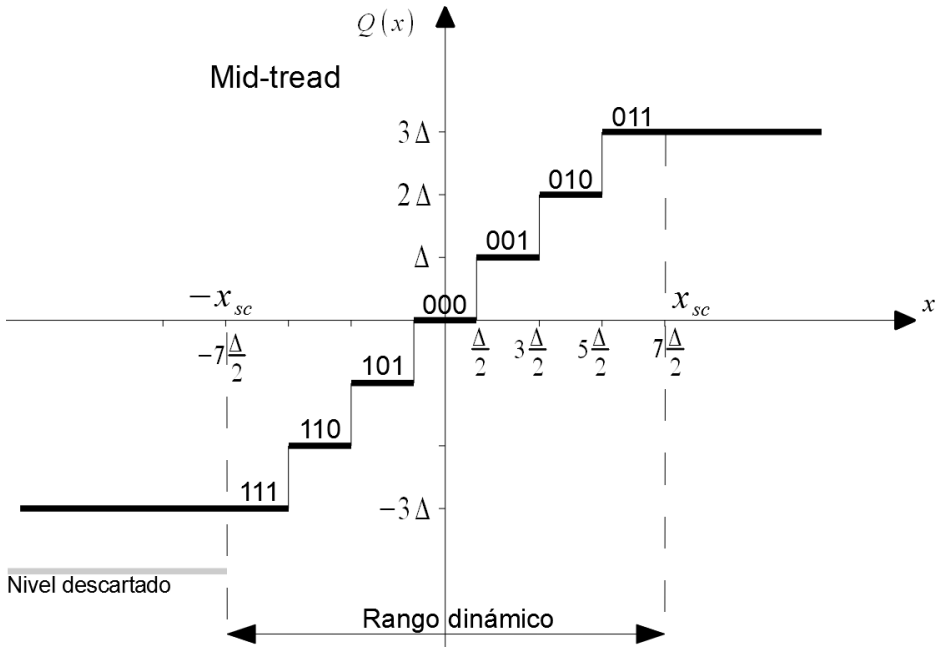


- $\Delta = 2x_{sc}/2^n$
- $k = \lfloor |x| / \Delta \rfloor$
- $y = \text{sign}(x) \cdot \Delta \cdot (k + 0{,}5)$
- La salida está limitada por los valores de salida correspondientes a los límites de sobrecarga (por ejemplo, si $x > x_{sc} \Rightarrow y = Q(x_{sc})$)

Tarea 1: Cuantificación uniforme en Matlab

Comprender la función:
[y, e]=qmidtread(x,xsc,n)

- x : vector de muestras de entrada (longitud arbitraria)
- xsc : valor de sobrecarga del cuantificador
- n : número de bits
- y : vector con valores cuantificados
- e : error de cuantificación , x-y



- $\Delta = 2x_{sc}/(2^n - 1)$
- $k = \text{round}(|x|/\Delta)$
- $y = \text{sign}(x) \cdot \Delta \cdot k$
- Hay que tener en cuenta los límites de la salida.

Tarea 1: Cuantificación uniforme en Matlab

leer apuntes

- Para x=-2:0.001:2 , xsc=1 y n=3
  - Representad la salida y el error de cuantificación de los dos cuantificadores respecto a la entrada x.
  - Indicad la zona de error granular y la zona de error de sobrecarga.
  - Ajustad adecuadamente los límites de las gráficas (funciones axis, xlim, ylim,...).
- Archivos Afonso_8kHz.wav y Gala_8kHz.wav, muestreados a 8 kHz y 16 bits (128 kbps).
  - Se pueden cargar con la función audioread()
  - Se pueden escuchar con sound() o soundsc()
- Cuantificación a 8, 4 y 2 bits, con qmidriser() y qmidtread(), xsc=1.
  - Escuchad y comentad los resultados. (Afonso y Gala)
  - Representad el error de cuantificación (Gala) y comentad los resultados.
- Repetir para 8 bits, xsc=0.5 y xsc=0.3, con ambos cuantificadores.
  - Representad el error de cuantificación (Gala). Comentad los resultados.

en función de n y unir mediante rectas

Relación señal-ruido de cuantificación

- En un cuantificador uniforme, en ausencia de error de sobrecarga, consideramos el error aprox. uniforme en el intervalo $(-\Delta/2, \Delta/2)$.
- En este caso la potencia de error (ruido de cuantificación) se aproxima a $\Delta^2/12$
- Entonces la relación señal-ruido (con P_s la potencia de la señal a cuantificar):

$$SNR(dB) = 10 \log \left(\frac{P_s}{\Delta^2/12} \right)$$

pot. de granular

- Mid-riser: $\Delta = 2x_{sc}/(2^n) \Rightarrow SNR(dB) = 6{,}02n - 20 \log \left(\frac{x_{sc}}{\sqrt{3P_s}} \right)$
  - Para una variable aleatoria uniforme entre $-x_{sc}$ y x_{sc}: $P_s = x_{sc}^2/3$

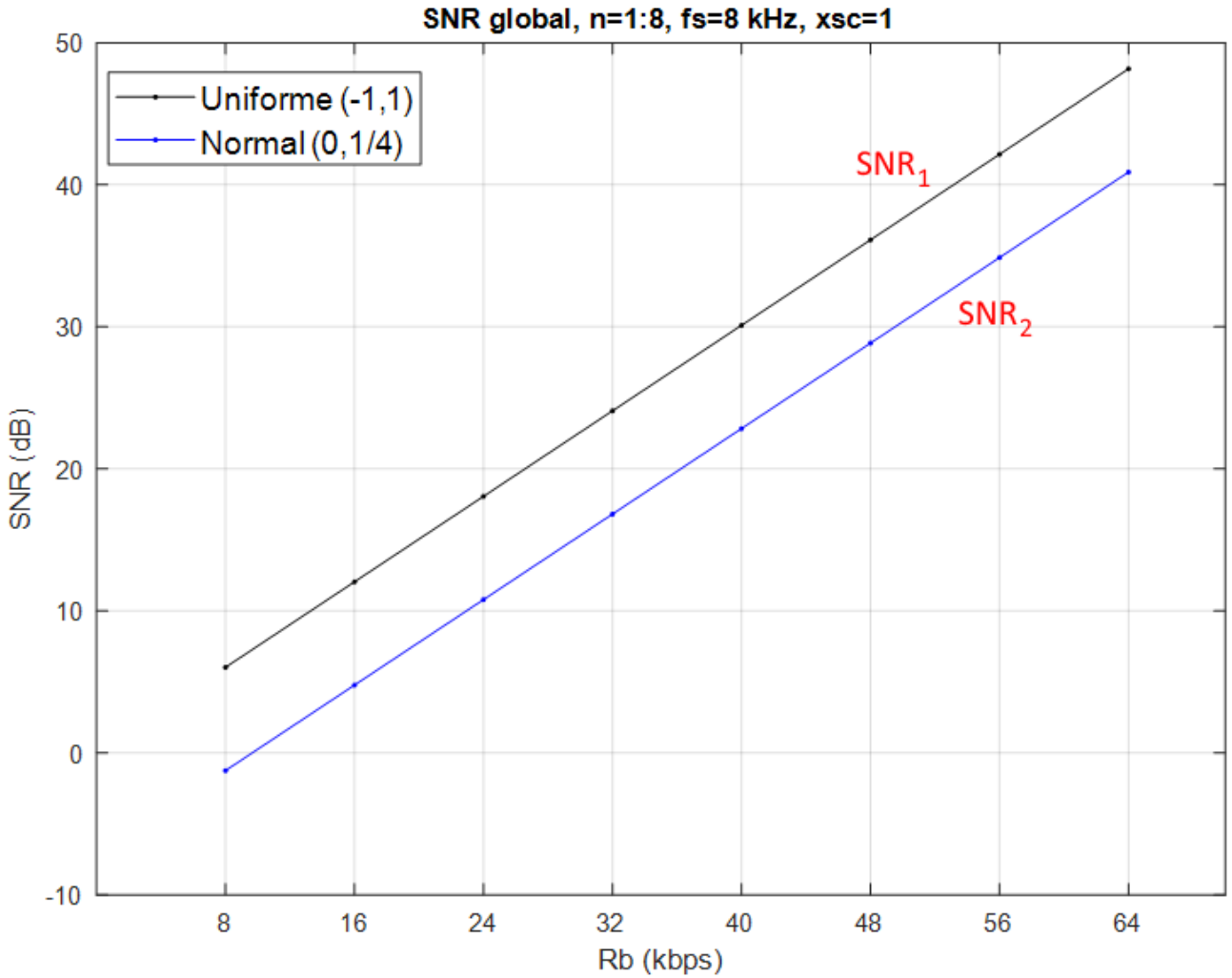
$$SNR(dB) = 6{,}02n \longrightarrow SNR_1$$

  - Para una variable aleatoria $N(0, \sqrt{P_s})$ y considerando $x_{sc} \approx 4\sqrt{P_s}$ ($P[|x|] > x_{sc}] \approx 6e-5$):

$$SNR(dB) = 6{,}02n - 7{,}27 \longrightarrow SNR_2$$

Relación señal-ruido de cuantificación

Ejemplo 1

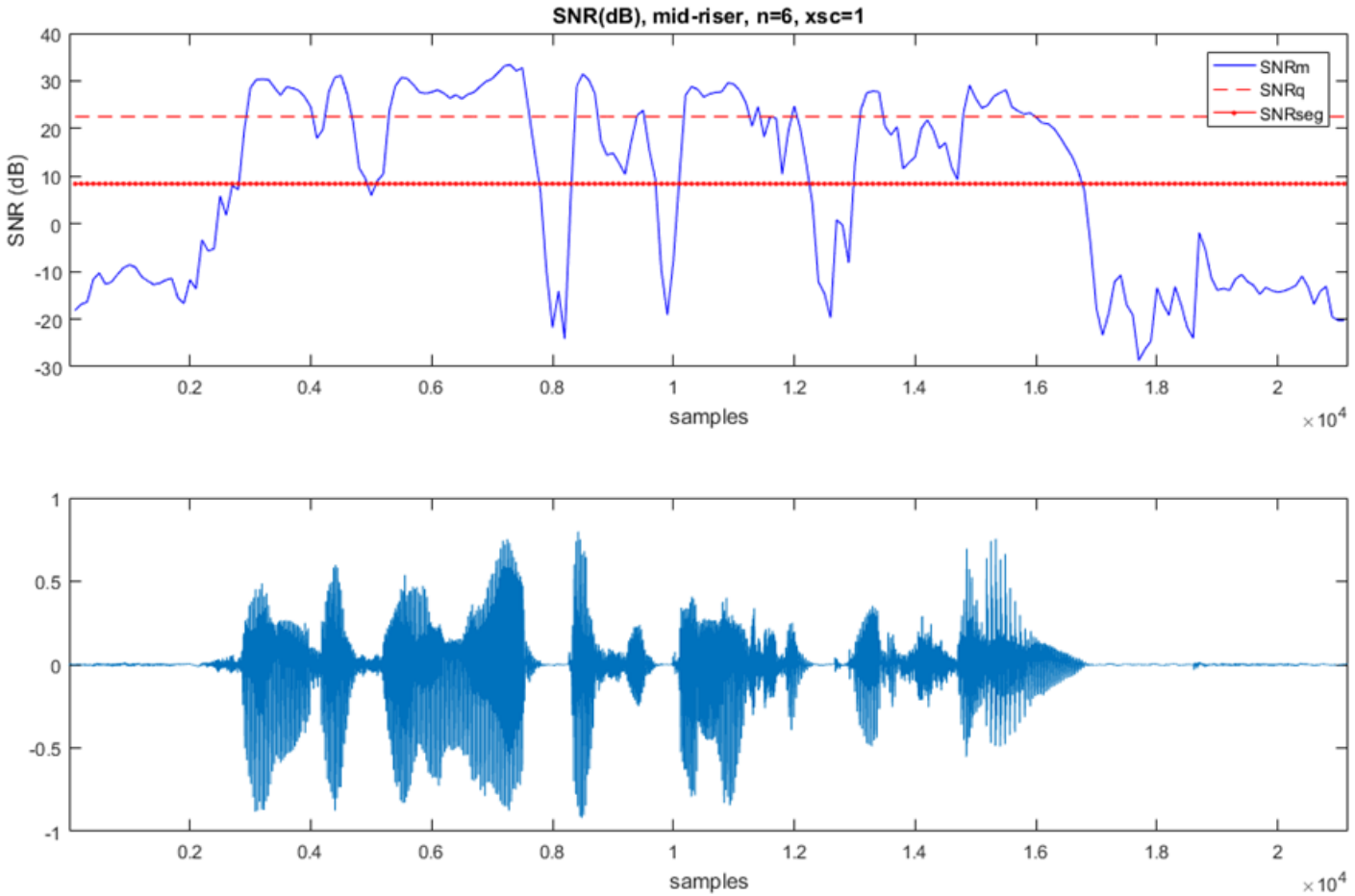


Relación señal-ruido de cuantificación

- En la práctica la SNR global se calcula: $SNR(dB) = 10\ log_{10}\left[\frac{\sum x^2(n)}{\sum(x[n]-\hat{x}[n])^2}\right]$
- Relación señal a ruido por tramas:
  - Se calcula dividiendo la señal en tramas o segmentos de longitud L.
  - Para cada trama se aplica la misma fórmula.
  - Si se divide x[n] en N tramas de L muestras, se obtienen N valores de la SNR que pueden representarse junto con la señal.
  - La SNR por tramas, SNRm , permite observar la evolución temporal de la SNR.
- Se define la SNR segmental, SNRseg, como el valor medio de la SNR por tramas. (El promedio se realiza directamente en dBs)

obtener error

Ejemplo 2



Tarea 2: Relación señal-ruido de cuantificación

Programad dos funciones para el cálculo de la relación señal ruido:

- Global: **SNRq=SNR(x,xq)** con x señal original, xq señal cuantificada, S relación señal ruido.
- Por segmentos: **[SNRseg, SNRm, m]** =SNRS(x,xq,L) con x señal original, xq señal cuantificada, L longitud del segmento, SNRm relación señal ruido por tramas (vector), SNRseg relación señal a ruido segmental, m vector de referencia en tiempo discreto para S
- El vector m debe contener unos índices temporales de referencia (el de la última muestra de cada segmento) de la SNRm de manera que se pueda representar con plot(m,SNRm)

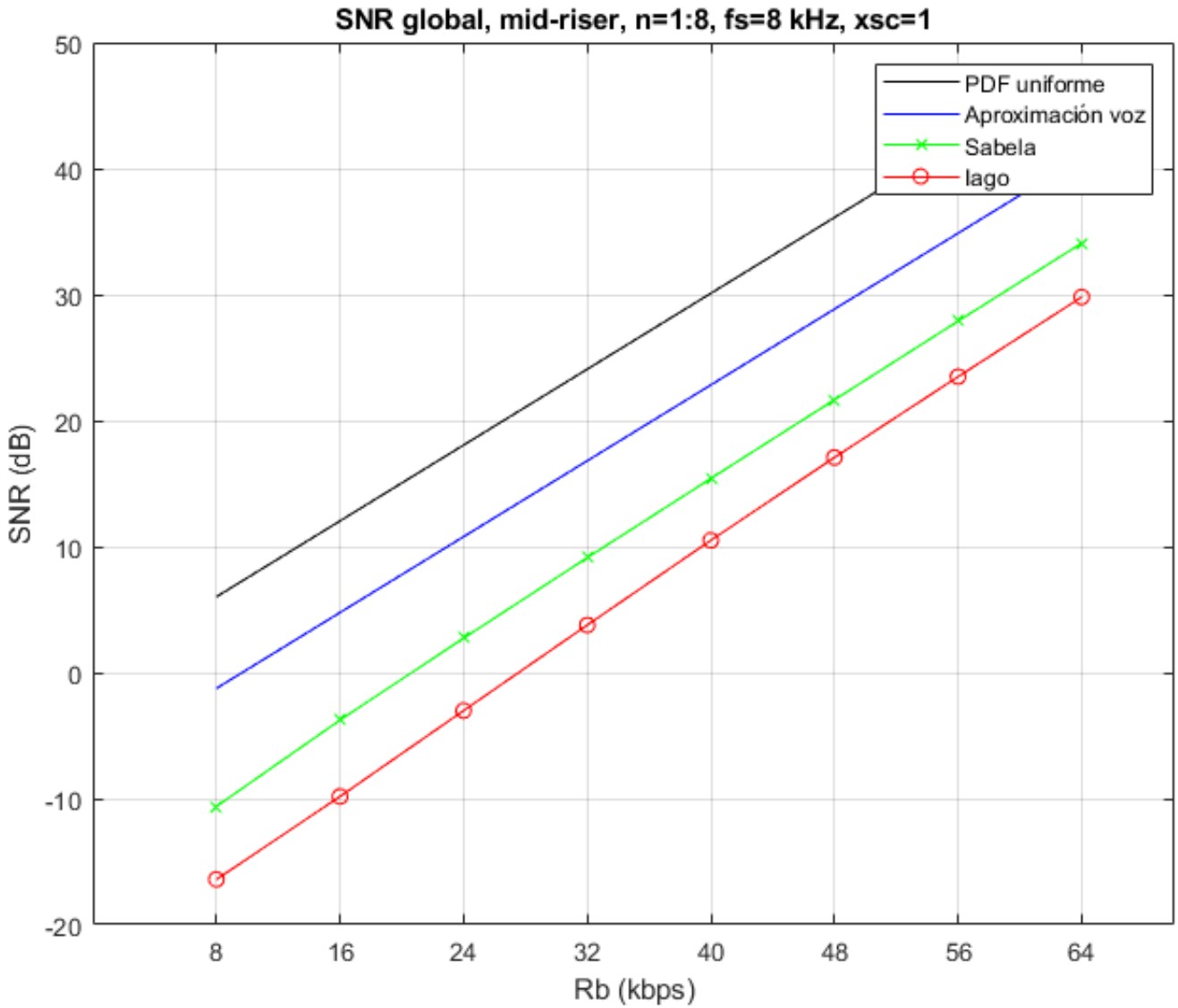
Tarea 2: Relación señal-ruido de cuantificación

- Para el fichero Gala_8kHz.wav, con qmidriser() y qmidtread() de la tarea 1 con xsc=1.
  - Obtened la SNR global, la SNR segmental y SNR por tramas. Representad junto a la señal original de forma análoga a "Ejemplo 2". Utilizad $n = 6$ y $L = 160$.
  - Explicad las variaciones de la SNR por tramas. ¿En qué tramos de la señal es mayor? ¿Es posible que en alguna trama sea cero?
- Repetid el apartado anterior para xsc=0.5 y xsc=0.3. Comparad y comentad los resultados.
- Para los ficheros Gala_8kHz.wav e Afonso_8kHz.wav, con n=1:8, xsc=1.
  - Obtened la SNR global en todos los casos, considerando qmidriser() y qmidtread().
  - Representad resultados como en la gráfica de la diapositiva "Ejemplo 3".
  - Repetid para n=1:8, xsc=0.5 y xsc=0.3. En este apartado sólo con qmidriser().
  - Explicad los resultados obtenidos.
- Requisitos de las gráficas. Todas las gráficas deben:
  - Estar referenciadas en el texto
  - Tener etiquetas en ejes x e y
  - Tener pie de figura explicativo
  - Es muy importante comentar todos los resultados e intentar extraer todas las conclusiones posibles.

aunque sean erróneas

Ejemplo 3

nos interesa SNR cuanto más alta mejor

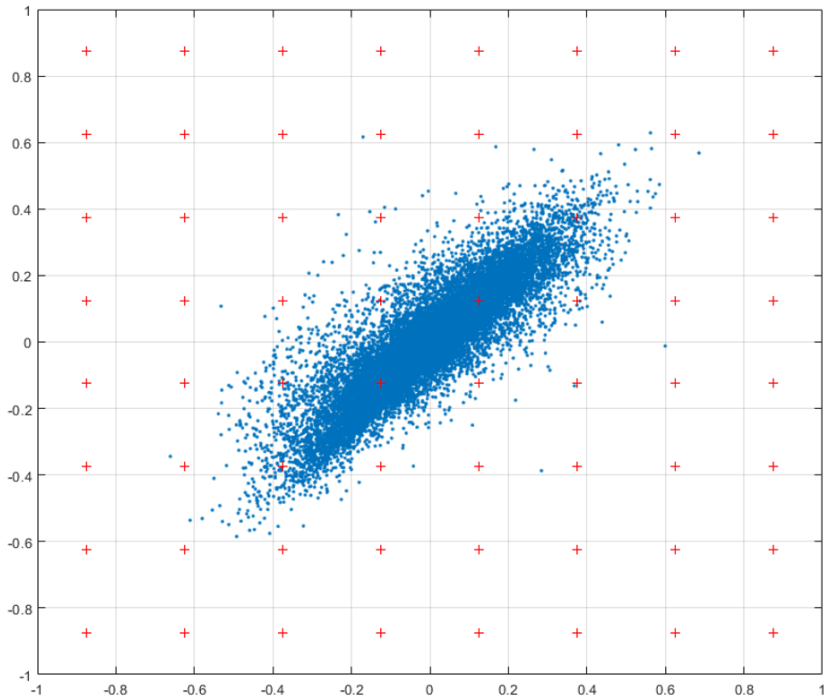


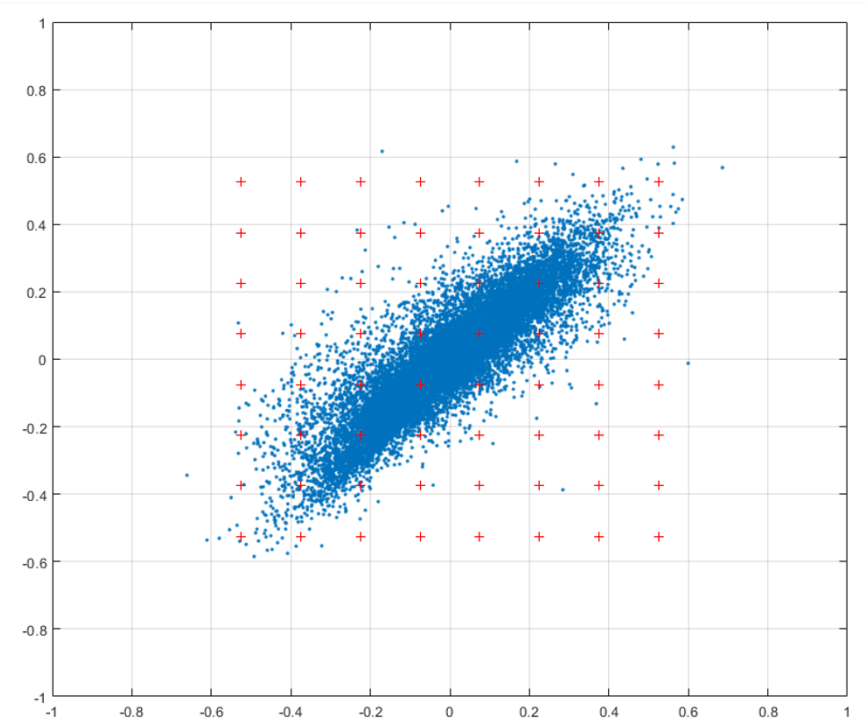
Para q se oigan la SNR a la Rb

plot(Rb,SNR1,'k',Rb,...); ax = gca; ax.XTick = Rb; grid

Introducción a la cuantificación vectorial (VQ)

- En vez de cuantificar valores aislados vamos a considerar vectores de muestras (o coeficientes).
- Formamos, por ejemplo, vectores de dos muestras consecutivas de voz y los representamos.



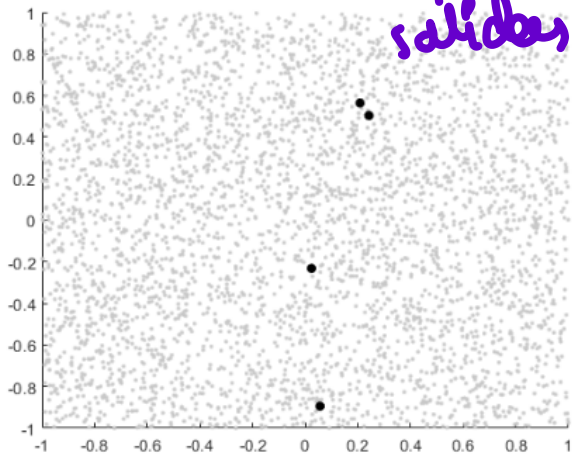


- ¡Gran parte de los niveles desaprovechados!
- ¿Cómo mejorar la cuantificación?

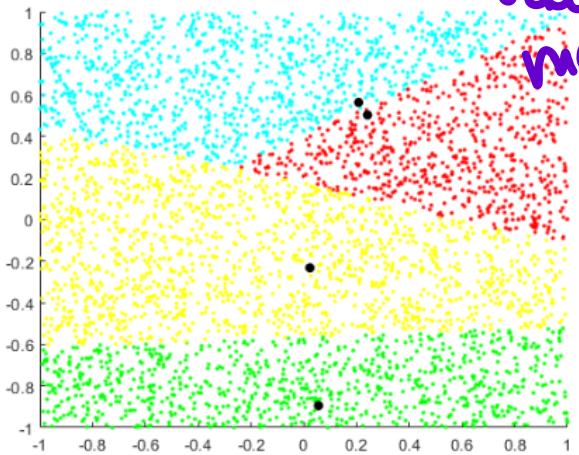
- Ajustar el valor de sobrecarga.
- ¿Es la mejor solución?

Ejemplo de diseño de un VQ de 2 bits

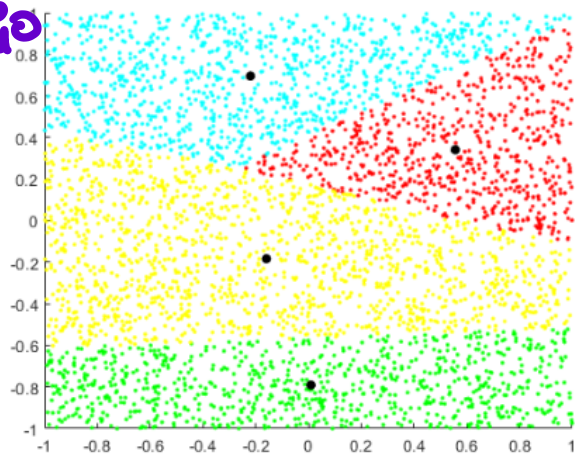
Inicialización de centroides



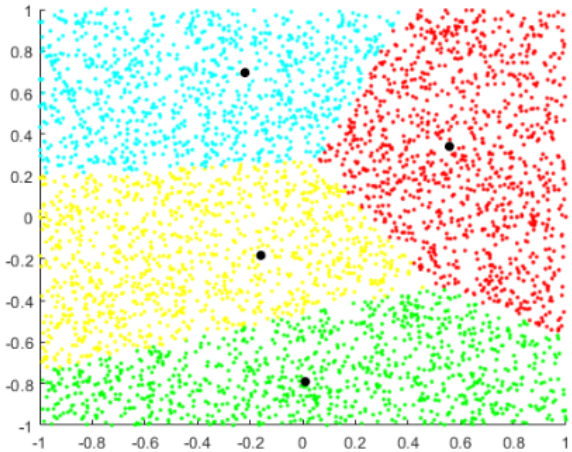
Asignación de vectores



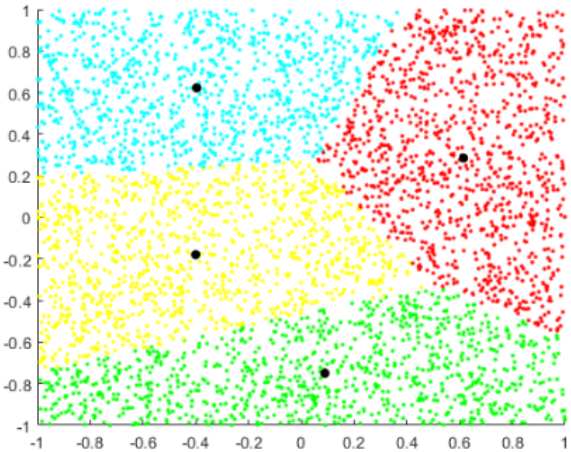
Reestimación de centroides



Resignación de vectores



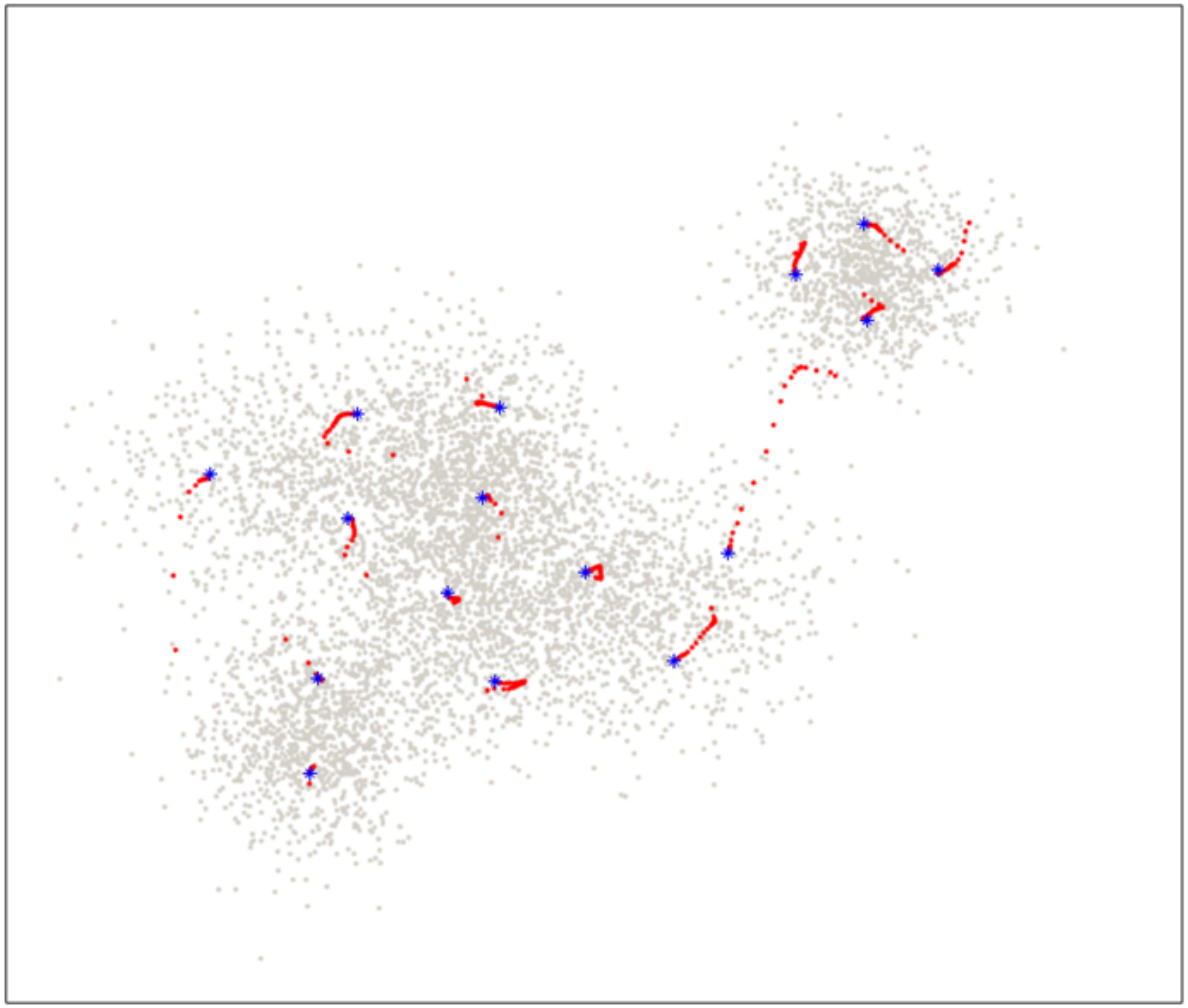
Reestimación de centroides



¿A qué converge?

...

Ejemplo de diseño de un VQ de 4 bits



- En este caso no son vectores de muestras de voz, pero el método de diseño es idéntico.
- ¿Qué hemos logrado?

Diseño de un VQ : algoritmo K-means

Necesita datos:

- Datos (vectores) de entrenamiento $\rightarrow$ Obtención del VQ.
- Datos de prueba (test) $\rightarrow$ Comprobar funcionamiento del VQ.

Algoritmo:

1. **Inicialización.** Selecciona arbitrariamente K vectores de los datos de entrenamiento, que serán los centroides iniciales del VQ.
2. **Asignación de centroides.** Para cada vector de entrenamiento calcula el centroide más próximo e incluye el vector de entrenamiento en su grupo (cluster) o celda.
3. **Actualización de los centroides.** Calcula el centroide de cada celda y actualiza el VQ. El centroide es el vector medio de cada celda (suma de todos los vectores de la celda dividida por el número de vectores en dicha celda).
4. **Iteración.** Repite los pasos 2 y 3 hasta que la medida de distorsión del cuantificador apenas sufra variación entre iteraciones.

Algoritmo K-means programado

function [VQ vDist no_asig] = Kmeans(data, nbits, VQini, umbral, display)

```
% VQ: Matriz con los centroides resultantes por filas.
% vDist: vector con la distorsión resultante en cada iteración
% del algoritmo
% no_asig: número de centroides no asignados (contienen NaN)
%
% data : matriz con vectores de entrenamiento por fila.
%        Dimensión VQ = número de columnas de data.
% nbits: numero de bits del VQ deseado.
% VQini: centroides iniciales por filas. Puede ser una matriz vacía.
% umbral: umbral para la variación relativa de la distorsión (MSE)
%         entre iteraciones utilizado como criterio de parada (ej. 1e-2).
% display: si vale 1 y la dimensión de los vectores de entrenamiento es 2,
%          muestra alguna gráfica ilustrativa.
% Posibles colores en gráficas: vectores de entrenamiento (verde), centroides
%          en cada iteración (rojo), centroides resultantes (azul).
%
% Ejemplo:  load traindata; [VQ,MSE]=Kmeans(traindata,4,[],1e-3,1);
```

Algunas funciones Matlab utilizadas: *randperm()*, *mean(A)*, siendo *A* una matriz.

Algoritmo K-means programado

vector centroides inicial

function [VQ vDist no_asig] = Kmeans(data, nbits, VQini, umbral, display)

- **Inicialización.** Si VQini vacío, inicializamos el cuantificador con vectores elegidos aleatoriamente de entre los datos de data. Es útil la función Matlab *randperm*.
- **Medida de distorsión:** Mean Square Error (MSE). A partir de los vectores de entrenamiento $\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_M} \in \mathbb{R}^N$ y de esos vectores cuantificados $\{\boldsymbol{y_i}\}_{i=1}^{M}$

M = nº centroides

$$MSE = \frac{1}{M}\sum_{i=1}^{M} \|\boldsymbol{x_i} - \boldsymbol{y_i}\|^2$$

- Normalizamos por la dimensión del vector (a efectos de comparación)

N

$$Dist = MSE/N$$

- **Criterio de parada:** varios posibles. Utilizaremos

$$\left| \frac{Dist(it) - Dist(it-1)}{Dist(it)} \right| < umbral$$

Tarea 3: entrenamiento del algoritmo K-means

Inicializar ejecución en paralelo (requiere Parallel Computing Toolbox):

```
if  isempty(gcp('nocreate'))
  parpool;
end
```

Con los datos en *traindata.mat*:

- Diseñar un VQ de 6 bits, utilizando un umbral de parada de 0,001 y $VQini = []$.
- display=1 para representar los datos de partida, la evolución de los centroides en cada iteración y los centroides resultantes.
- Representa cómo varía la distorsión obtenida, *vDist*, en función de la iteración.
- Ejecuta el algoritmo Kmeans 20 veces[4] . Guarda el tiempo de ejecución y la distorsión de la última iteración de cada ejecución. Represéntalos gráficamente. (Ayuda: funciones Matlab *clock* y *etime*)

Con el fichero *tvg_training_20s.wav*:

- Forma la matriz *training* que contenga pares, no solapados, de muestras consecutivas por filas. Cada fila será un vector de datos de dimensión 2.
- Diseñar un VQ de 6 bits (VQ6) con umbral=0.01.
- Sobre una misma gráfica representar los datos de partida y los centroides del VQ resultante.
- Representa cómo varía la distorsión obtenida en función de la iteración.

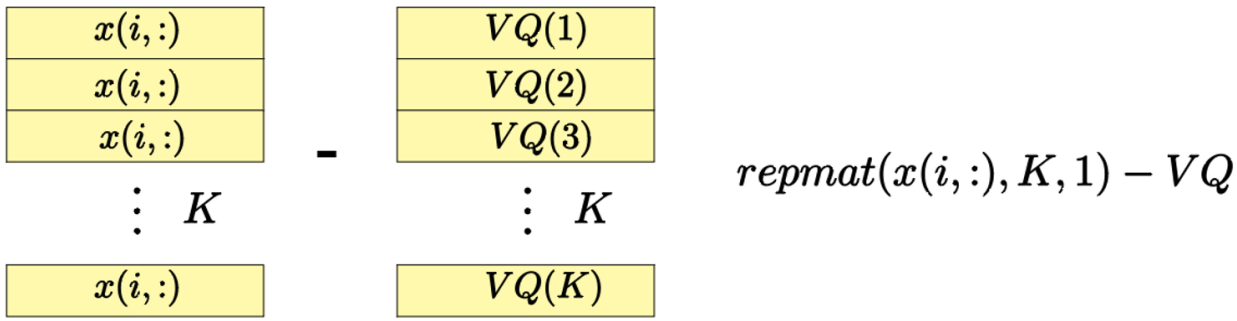
[4] Antes de la primera ejecución inicializa *parpool*, como se indica arriba, fuera de *Kmeans*.

Tarea 4: aplicación de un VQ previamente entrenado

function [y, Dist] = VQuantize(x,VQ)

```
% y: matriz con los vectores resultantes de la cuantificación
% Dist: Error cuadrático medio entre la dimensión del vector.
%
% x : matriz con vectores a cuantificar por fila.
% VQ: Matriz con los centroides del VQ por filas.
```

Cada fila de ***y*** es el centroide más próximo en distancia euclídea a cada fila de ***x***.



- Considera los ficheros *tvg_training_20s.wav* y *tvg_test_20s.wav*.
- Compara las distorsiones obtenidas al aplicar VQ6 sobre el conjunto de vectores de entrenamiento y de test.
- Para ambos conjuntos reordena las salidas del VQ de forma que se puedan escuchar las señales cuantificadas.
- En ambos casos estima la SNR de cuantificación global en dB.

Avance Sesión 2. Tarea 1: Comparación SNR algoritmo K-means

Los resultados de esta tarea formarán parte del segundo informe de progreso.

Vamos a comparar la SNR obtenida con cuantificación escalar y vectorial en función de R_b.

Tarea 1a. Utilizando como datos la matriz *training*:

- Obtén con *Kmeans* cuantificadores vectoriales de 1 a 16 bits (*umbral* $= 0{,}01$).
- Almacena los 16 VQs en una única matriz *VQt* de forma que las dos primeras filas se correspondan con los centroides del VQ de 1 bit, las cuatro siguientes con los centroides del VQ de 2 bits, etc.
- Registra el tiempo que tarda en estimarse cada uno de los 16 VQs y represéntalo en función del número de bits.
- Guarda la matriz *VQt* y los tiempos registrados en el fichero *VQt.mat*.

Tarea 1b. Considerando como señal de test *tvg_training_5s.wav* (fragmento de *tvg_training_20s.wav*):

- Representa la SNR global obtenida con los VQs en función del régimen binario. ¿Cuál es ahora el número de bits por muestra?
- Compárala con las gráficas que se obtienen con *qmidriser* y las aproximaciones SNR1 y SNR2.

Tarea 1c. Repite la tarea 1b considerando la señal en *tvg_test_5s.wav*. Compara los resultados de ambas tareas y justifica las diferencias encontradas (puede ser útil idear alguna gráfica adicional).