Gabriele Petrillo

Fondamenti dell'audio digitale 2

Ciò che segue è stato sviluppato a partire dalla dispensa di composizione del 09/12/2019 del prof. N.Bernardini presso il conservatorio S.Cecilia di Roma che possono essere consultati presso https://github.com/SMERM/TR-2019-2020/tree/master/A.A.2019-2020/COME-02/20191209. L'argomento trattato sono i Decibel e la quantizzazione di un segnale audio con dimostrazioni ed esempi in Octave.

Decibel

Il Decibel è un'unità di misura logaritmica del rapporto tra due grandezze omogenee. Viene generalmente usato per misurare i fenomeni acustici ed esprime varie unità di misura, il pascal, il volt, il watt, a seconda del tipo di dB preso in considerazione. La caratteristica del decibel è quella di rappresentare i dati in modalità logaritmica e questo permette di descrivere grandi variazioni numeriche, come la pressione sonora, con numeri di poche cifre. L'espressione matematica è:

$$dB = 10 * log_{10}(liv./liv.\,di\,riferimento)$$

Il dBSPL (sound pressure level) è una scala usata per misurare il livello di pressione sonora. È la grandezza utilizzata per esprimere il livello di pressione del suono rispetto la soglia di udibilità. La scala parte da 0 dB, che rappresenta la soglia di udibilità, ed arriva a circa 125 dB che rappresenta la soglia del dolore - ovvero la massima pressione sonora udibile in modo continuato senza creare danni all'apparato uditivo[1].

Bisogna tener presente che essendo una grandezza logaritmica ogni raddoppio della pressione acustica è rappresentata da soli 6dB - quindi il doppio di 4o dBSPL sarà 46 dBSPL.

Il Range Dinamico

il range dinamico è espresso in decibel e rappresenta la differenza tra il livello più alto e quello più basso di un segnale. Nel caso di un sistema digitale rappresenta il più alto livello sonoro rappresentabile senza distorsioni e il rumore di fondo dei componenti elettronici.

Una grandezza associata al range dinamico è il rapporto segnale/rumore (SNR), ovvero il rapporto tra il segnale più forte possibile ed il rumore di fondo. Più il rapporto è ampio e più il suono risulterà chiaro. In condizioni ottimali il range dinamico è uguale al SNR.

Quantizzazione

Un altro passaggio del processo di conversione è la quantizzazione, ovvero l'accuratezza con cui è misurato il valore numerico di ogni campione. I valori di un segnale campionato possono assumere solo valori che possono essere rappresentati dalla memoria del computer, in genere numeri interi. I valori dei campioni vengono quindi arrotondati per difetto al valore più vicino rappresentabile. Per esempio, supponendo che il convertitore D/A è capace di rappresentare un segnale ad intervalli di 0.001V. Se un campione è campionato a 0.01227 V, sarà convertito a 0.012 V - con un errore di 0.00027 V.

Rumore di quantizzazione

Il valore di ogni campione differisce leggermente dal valore del segnale originale e questo problema è conosciuto come errore o rumore di quantizzazione. La quantità e l'effetto udibile dell'errore dipende dalla risoluzione del con-

[1] In realtà la scala arriva a circa 194 dB che rappresenta il limite fisico, ma rappresentano livelli di pressione sonora che non sono utili alla nostra trattazione.

vertitore e dal tipo di segnale in ingresso. Per esempio, se un convertitore A/D lavora con 12 bit si avrà in uscita di $2^{12} = 4096$ possibili valori. Supponendo che il segnale audio in ingresso può variare tra -10 e +10V - per un range totale di 20V, la grandezza dell'intervallo di quantizzazione corrispondente al valore di un bit sarà $20/4096 = 0.004883$ V/bit. Questo errore si mostra generalmente come un fruscio (white noise), tuttavia può manifestarsi in maniera diversa a seconda del tipo di segnale in ingresso al convertitore.

Se il segnale è molto basso di livello si possono verificare dei troncamenti che creano delle armoniche non presenti nel segnale originali che potrebbero superare la frequenza di Nyquist e creare delle frequenze di alias.

Per eliminare il problema è possibile inserire del rumore bianco a bassissimo livello (la quantità di rumore aggiunto in genere è dell'ordine di 3 dB) in modo da mascherare le frequenze spurie che si vengono a creare in seguito al troncamento del sengnale, questo rumore è chiamato *dither*. È possibile anche aumentare la risoluzione dei bit per avere un range dinamico più grande e far ricadere gli artefatti ad un livello non udibile. Aumentando la risoluzione in bit almeno a 20 avremo infatti almeno 120 dB di dinamica in modo da eccedere il range dinamico percepibile dall'uomo in qualsiasi ambiente.

Il range dinamico in un sistema digitale

Per calcolare il massimo range dinamico di un sistema digitale, possiamo usare la seguente formula:

$$Max.\,range\,dinamico\,in\,dB = n.bit * 6$$

Il numero 6 deriva dal fatto che ogni bit di risoluzione rappresenta 6 dB di dinamica. Quindi, se registriamo un suono con un sistema ad 8 bit avremo un range dinamico di 48 dB, se registriamo a 16 bit, il range dinamico aumenta a 96 dB. Un convertitore a 20 bit offre un range dinamico di 120 dB, che corrispondo all'incirca a quello dell'uomo.

Ex.1 Simulazione di segnale quantizzato a 2 3 4 8 12 16 bit

```
[snd fc] = audioread ("babay.wav");
nbits = [2 3 4 8 12 16];
out = zeros (length(snd) *
lenght(nbits),1);
dur = length(snd)/fc;

for k = 1:length(nbits)
curbits = 2^nbits(k);
curstep = 2/curbits;
globstart = (length(snd)*(k-1)) + 1;
tmpout = round(snd / curstep)*
curstep;
out(globstart : globstart +
length(tmpout)-1) = tmpout;
end
audiowrite("babaynew.wav",out,fc)
```

Cerchiamo di leggere il codice velocemente senza soffermarci sul funzionamento degli opcode già descritti nella lezione precedente.

Per prima cosa dobbiamo leggere il file audio da ricampionare attraverso la funzione *audioread*. Scriviamo un array *nbits* con la profondità di bit che vogliamo utilizzare ad ogni ciclo. Costruiamo una matrice *out* grande quanto *snd* * il numero di elementi di *nbits* - il prodotto tra il file campionato ed il numero di volte che verrà riquantizzato - che sarà la matrice di output con conterrà i nuovi campioni. Ed infine stabiliamo la durata in secondi del file con la variabile *dur*.

Ora dobbiamo creare il ciclo che leggerà il file con i diversi valori di quantizzazione. Possiamo leggerlo così per K uguale a 1 arriva al numero di elementi presente in *nbits* - che rappresenta il numero di volte che dobbiamo rileggere il file.

Ora dobbiamo specificare per ogni ciclo - quindi per ogni valore di k - la quantità di valori di quantizzazione secondo la formula 2^{nbits} il valore di ogni step $2/quantità\,stepquindi$:

```
curbits = 2^nbits(k);
curstep = 2/curbits;
```

Con k stabiliamo la posizione dell'elemento nell'array *nbits*.

Con *globalstart*[2] identifichiamo il campione di inizio di ogni ciclo.

Ora dobbiamo riquantizzare tutti i campioni, la formula è:

$$x_{qnt} = [x/q] * q$$

dove x_{qnt} è il segnale quantizzato, x sono i campioni originali - nel nostro caso *snd* e q è il valore degli step di campionamento, quindi *curstep*.

quindi:

```
tmpout = round(snd/curstep)*curstep;
```

Per finire dobbiamo scrivere i valori nella nostra matrice di uscit *out* copiando i valori ti tmpout nelle posizioni che vanno da globalstart - il campione di inizio - alla lunghezza di *tmpout* - tutta la lunghezza dei nuovi campioni, successivamente *audiowrite* scriverà il nuovo file audio:

```
out(globstart : globstart +
length(tmpout)-1) = tmpout;
end
audiowrite("babaynew.wav",out,fc)
```

Lo script produce il seguente plot:

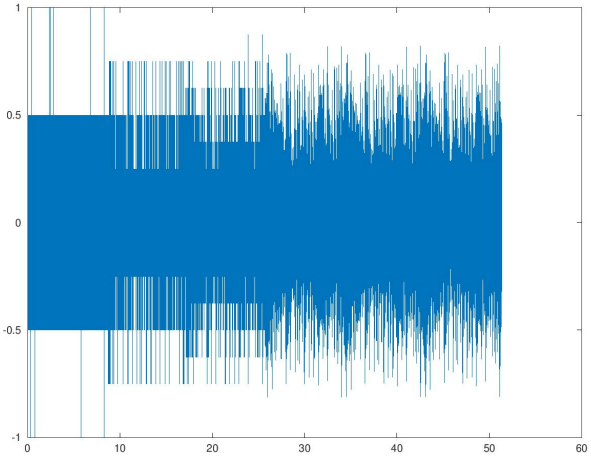

fig.1

[2]Per la spiegazione di globalstart guardare lez$_1 Ex_4$

Ex.2 Calcolo dell'errore di quantizzazione

Cerchiamo di calcolare l'errore di quantizzazione di un segnale. Per calcolare l'errore di quantizzazione dobbiamo applicare la formula:

$$z_{errore} = x_{sig} - x_{qnt}$$

dove z - l'errore di quantizzazione - è uguale alla differenza tra segnale in ingresso x_{sig} e il segnale quantizzato x_{qnt}.

```
[snd fc] = audioread ("babay.wav");
nbits = [2 3 4 8 12 16];
out = zeros (length(snd) *
lenght(nbits),1);
dur = length(snd)/fc;

for k = 1:length(nbits)
curbits = 2^nbits(k);
curstep = 2/curbits;
globstart = (length(snd)*(k-1)) + 1;
tmpout = round(snd / curstep)*
curstep;
dif = snd - tmpout;
out(globstart : globstart +
length(tmpout)-1) = dif;
end
audiowrite("babaydif.wav",out,fc)
```

Questo codice è uguale al precedente in cui però è stato implementato il calcolo dell'errore di quantizzazione e la matrice di output è stata uguagliata all'errore, così possiamo stampare l'errore:

```
dif = snd - tmpout; out(globstart :
globstart +
length(tmpout)-1) = dif;
```

lo script produce il seguente plot:

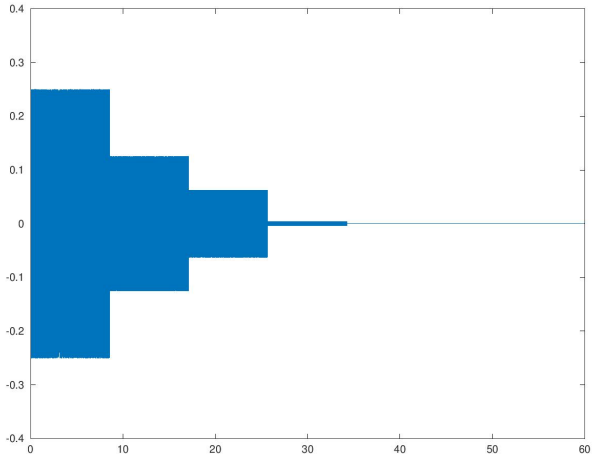

fig.2

Da questo plot possiamo vedere che alla fine non abbiamo nessun errore di quantizzazione, questo avviene perchè il file di origine non è un segnale analogico, ma digitali campionato a 16 bit.

BIBLIOGRAFIA

•Dodge, Charles and Jerse, Thomas A, *Computer music: synthesis, composition and performance*, Macmillan Library Reference 1997

•Roads, Curtis and Strawn, John and others, *The computer music tutorial*, MIT press, Editor 1996

•Massimi, Marco, *Mastering in the box*, Contemponet 2018