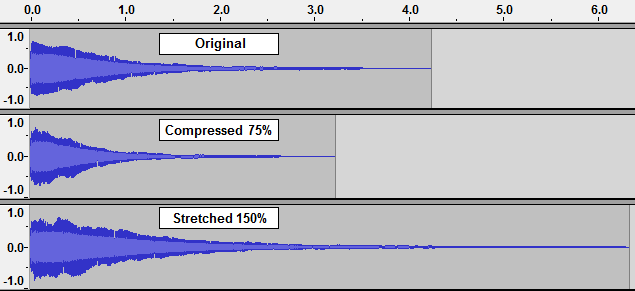
Il Phase Vocoder

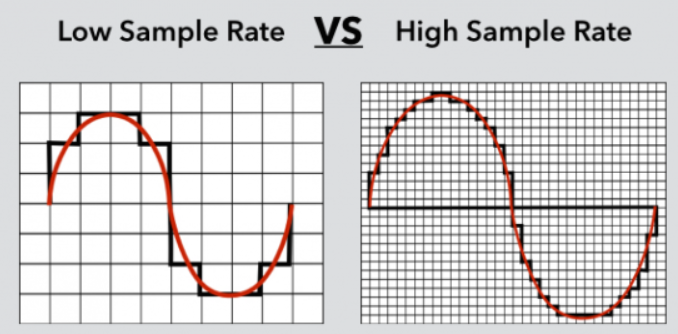


# Introduzione

In diversi campi di applicazione, successivamente alla digitalizzazione dei segnali analogici, si è avuta la necessità di sviluppare metodi e algoritmi per riuscire a modificare aspetti e variabili legati alla loro rappresentazione. In particolar modo nel campo musicale, la richiesta di poter modificare a piacimento le variabili legate ad una traccia audio digitale, è stata e continua ancora oggi ad essere oggetto di ricerca.

Uno dei principali studi, si è concentrato sulla possibilità di poter modificare la velocità effettiva di una traccia audio e di poterne incrementare o ridurre l’intonazione (pitch). Uno dei metodi per raggiungere i due scopi è attraverso un algoritmo chiamato **Phase Vocoder**.

La Teoria dei Segnali ha reso possibile un’analisi approfondita delle tracce audio digitali, a partire dallo studio delle loro caratteristiche frequenziali. Sappiamo infatti che una traccia audio, per essere digitalizzata, viene campionata ad una certa frequenza, quantizzata, e codificata attraverso dei bit di livello. La frequenza di campionamento definisce appunto il numero di campioni (in un secondo) presenti nel segnale digitale. La più comune nell’ambito musicale è 44,1 Khz: questo è dovuto al fatto che, per poter ricostruire il segnale senza errori e in maniera definita, si è dovuto scegliere circa il doppio delle bande di frequenza massime udibili dall’essere umano (22 Khz), tali da rispettare il **Criterio di Nyquist.**



In linea teorica dunque, una possibile soluzione al problema sulla modifica dell’intonazione e della velocità di una traccia, sarebbe quella di riprodurla, aumentandone o diminuendone la **frequenza di campionamento**. Questa soluzione però, non permette la scelta di modificare singolarmente uno dei due parametri, in quanto entrambi direttamente proporzionali. Si può verificare dall’esempio seguente:

**22050Hz(Originale**) **33050Hz**



Per questa ragione, vengono implementate una serie di operazioni che ci permettono di cambiare singolarmente i due parametri. Possono essere schematizzate in due fasi:

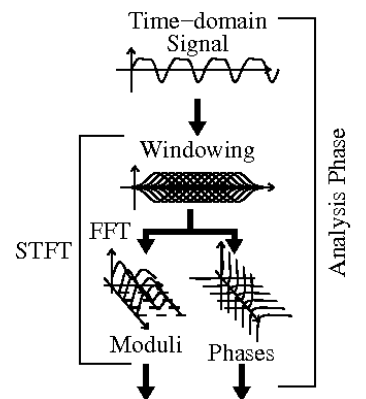
**-Fase di Analisi**

**-Fase di Sintesi**

Le operazioni da svolgere

Nella prima fase, la traccia audio viene analizzata attraverso un’operazione denominata come **Short Time FFT**:

A differenza della FFT standard, la trasformata non viene eseguita su tutto il segnale, ma singolarmente su sezioni locali della traccia audio, che vengono prodotte mediante una divisione in segmenti, chiamati finestre (windows). Successivamente alla segmentazione, viene eseguita una moltiplicazione dei segmenti con una funzione finestra, chiamata **Finestra di Hann**. Inoltre, prima di essere eseguita una FFT su ogni singola finestra, i segmenti vengono sovrapposti tra di loro sulla base di alcuni parametri che dobbiamo definire.

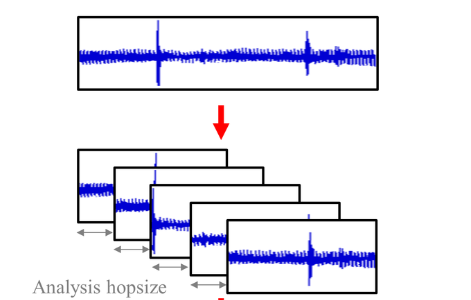


I parametri in questione sono:

**-WindowSize**: Che rappresenta la dimensione di ogni singola finestra, misurata in campioni (samples)

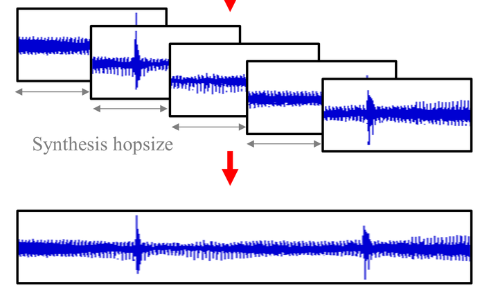
**-AnalysisHopSize**: Che è la distanza tra una finestra e un’altra nella fase di analisi del segnale audio, misurata in campioni

**-SynthesisHopSize**: Che è la distanza tra una finestra e un’altra nella fase di sintesi del segnale audio, misurata in campioni



Naturalmente, maggiore sarà la grandezza della finestra e maggiore sarà l’accuratezza dell’analisi in frequenza. Possiamo notare infatti, che la suddivisione in finestre nel dominio del tempo, produce una suddivisione in canali nel dominio delle frequenze, ognuno dei quali possiede una larghezza di banda che possiamo determinare dividendo la frequenza di campionamento per la grandezza della finestra, ottenendo cosi la larghezza di banda coperta da ogni singolo canale.

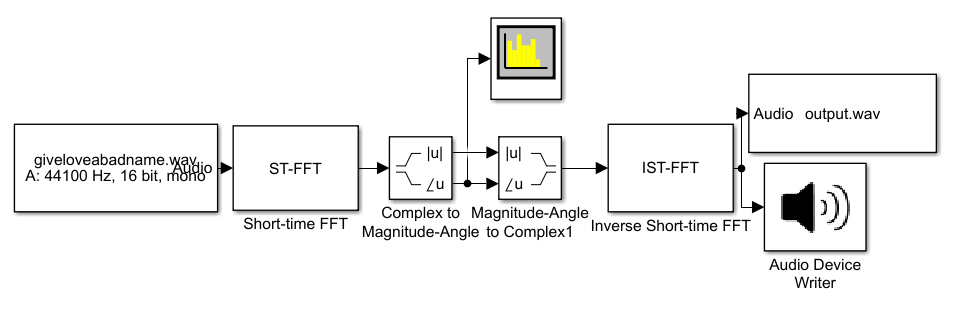
Nella seconda fase invece, viene eseguita una Short Time FFT inversa, attraverso il quale le singole finestre saranno riportate nel dominio del tempo, rifinestrate, sommate e sovrapposte in base al parametro SynthesisHopSize, in modo da ricostruire il segnale originale:



La differenza di intonazione rispetto alla traccia originale dunque, è determinata a partire dal rapporto tra il SynthesisHopSize e l’AnalysisHopSize: infatti scegliendo un numero maggiore di campioni in fase di ricostruzione rispetto alla fase di analisi, avremo conseguentemente un aumento della frequenza di campionamento nella traccia ricostruita, e dunque un aumento dell’intonazione. Viceversa se il numero di campioni in fase di sintesi sarà minore, ne risulterà una traccia audio con intonazione minore.

Se volessimo invece modificare solo la velocità, ci basterebbe riprodurre la traccia ricostruita alla frequenza di campionamento originale, ottenendo quindi un rallentamento se il numero di campioni in fase di sintesi è maggiore, e una velocizzazione se il numero è minore.

Setup di Simulink



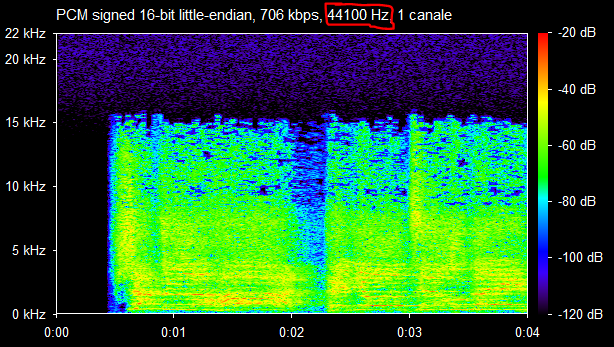
Nel progetto Simulink, ho utilizzato come ingresso una traccia audio mono di durata 4 secondi e con frequenza di campionamento originale a 44100Hz.

Per la modifica dell’intonazione, ho scelto come parametri, una WindowsSize di 2048 campioni, un’AnalysisHopSize di 512 campioni e un SynthesisHopSize di 1024 (volendo aumentare di un’ottava l’intonazione).

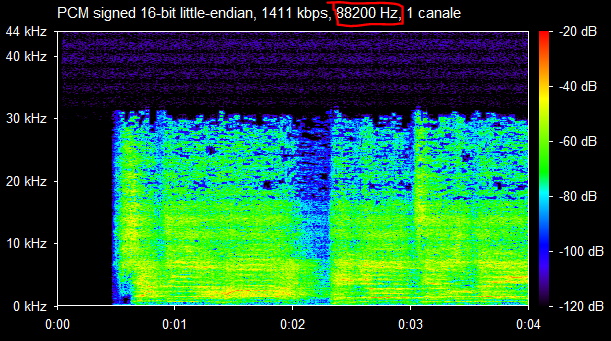
Per la modifica della velocità ho lasciato gli stessi parametri tranne il SynthesisHopSize, che l’ho scelto a 256 campioni. (volendo raddoppiarla)

Nel primo caso possiamo vedere che la traccia ricostruita ha esattamente il doppio della frequenza di campionamento di quella originale:

**-Originale**

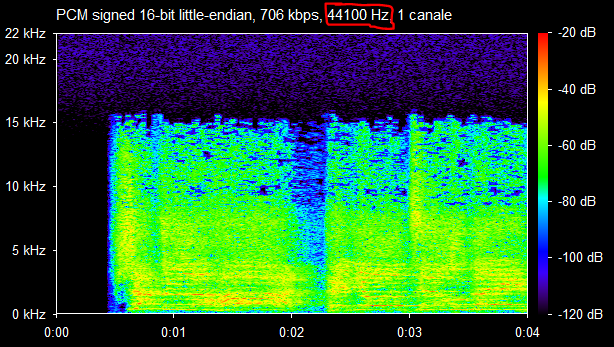


**-Ricostruita**



Nel secondo invece, la traccia ricostruita ha una frequenza di campionamento più piccola, che riprodotta alla sua frequenza originale, ci darà la traccia audio velocizzata:

**-Originale**



**-Ricostruita**

