POLITECNICO DI MILANO
Facoltà di Ingegneria
Dipartimento di Elettronica e Informazione
Corso di Laurea in Ingegneria delle Telecomunicazioni

Studio e valutazione di algoritmi per la rilevazione dell'attività vocale e per la cancellazione d'eco acustico nei segnali telefonici codificati con tecniche a predizione lineare (ACELP)

Relatore: Prof. Sergio Brofferio
Correlatori: Ing. Luca Prati, Ing. Danilo Neri

Tesi di Laurea Specialistica di:
Daniele Giacobello
matricola 667066

Anno Accademico 2005-2006

Conosco a memoria i discorsi con cui ci hanno martellato il cranio
dalla culla alla scuola e poi dal pulpito:
sii povero di spirito,
sii umile di mente,
rinuncia alla ragione,
spegni quella luce abbagliante dell'intelligenza che ti infiamma e ti consuma,
perché più saprai, più sarai destinato a soffrire,
rinuncia ai tuoi sensi,
sii prigioniero della Santa Fede,
vivi nel tuo stato asinino.
Ma vivere in questo modo è vivere da morti.

dalla sceneggiatura originale
del film Giordano Bruno di G. Montaldo

Indice

Lista tabelle

Lista figure

Ringraziamenti

In primo luogo, un doveroso ringraziamento va al relatore di questo lavoro di tesi, il Professor Brofferio, per l'aiuto e la grande libertà concessami, ma soprattutto per la cordialità e la stima mostrata nei miei confronti.
Un sincera riconoscenza va inoltre ai correlatori, l'Ingegner Prati e l'Ingegner Neri, per avermi concesso l'opportunità di svolgere questo lavoro e la costante disponibilità dimostratami in questi mesi, dallo studio preliminare all'impaginazione finale. Un grazie va anche al mio papa-boy preferito, Matteo, per aver collaborato alla realizzazione della prima parte della tesi e per l'amicizia dimostratami.
Il mio debito verso i miei genitori è più grande che verso chiunque altro. Essi sono stati le colonne su cui ho potuto poggiare in ogni momento. Mi hanno guidato, ispirato, incoraggiato e sostenuto. Soprattutto hanno sempre creduto in me. Questo lavoro di tesi è dedicato a loro.
La mia gratitudine va inoltre a tutta la mia famiglia per i piccoli e grandi finanziamenti alla mia dissolutezza ☺ e per l'affetto sempre dimostratomi.
Molte sono le persone che in questi anni mi sono state vicine fisicamente e moralmente e, direttamente o indirettamente, hanno contribuito al raggiungimento di questo importante obbiettivo della mia vita, specialmente (in ordine rigorosamente alfabetico):
Alessio, per aver contribuito a purgare un numero considerevole di supponenti superstar di reti (da dove cominciamo?) e per le tue letture culturali neo-reazionarie (vedi "Il cavaliere d'inverno");
Andrea, per le tue bestemmie sempre a proposito e le grigliate alcoliche;
Emilia, per il tuo corpo spettacolare e per tutte le volte che non ce l'abbiamo fatta ad arrivare al cinema (o che non ci abbiamo neanche provato);
Francesca, per essere sempre una convinta giansenista, nonostante il passare degli anni, per le nostre seratine etno-radical-chic (…e comunque insisto che la tipina bionda era proprio una fighetta) e per i tuoi uomini rudi (leggi bifolchi) che vuoi trasformare in dandy;

Giulia, per il tuo sublime intelletto e la tua prosa tagliente (vedrai che ce la faremo un giorno ad entrare nell'intellighenzia…) e quas'm'dimenticaf e pe' averm coret li congiuntife che i' no saccio di parlaro beine;

Jay, for being always there for me, you'll always be my big brother! (ps: you scared to f*ck with us cause we got abramo in the back ah! bite yo balls off ah!);

Luca, io e te saremo sempre parte della meglio gioventù: belli e proletari! (o alto alto borghesi che va bene uguale);

Lucia, per le lunghe chiacchierate, le cene all'eritreo a brindare alla nostra indipendenza emotiva e per l'incredibile energia che sprigioni (vedi i tuoi diari… specialmente lo spogliarello in chiesa);

Marco, per le salsicce, per le nostre luuuuuunghe giornate di studio, per le centinaia di sigarette fumate e per -come tralasciarla?- la tua insana adulazione verso i professori (Matricciani TVB, ce la fai?);

Mario, per tutte le belle serate passate insieme, per avermi lasciato girare documentari verità con la tua fotocamera (??), per avermi offerto supporto psicologico nei momenti brutti e per avermi tenuto buono nelle mie serate da marcio (vedremo stasera come va, casomai mi fermo da te per qualche giorno…);

Marta, per i nostri momenti di serietà, per i film di Eisenstein e per i nostri discorsi "di sinistra";

il professor Spalvieri, per essere stato il mio mentore in questi anni di università ultra-spersonalizzante;

Stella, per conoscermi fin troppo bene e per i nostri piccoli segreti inconfessabili;

Virginia, per aver sempre sopportato silenziosamente le mie molestie sessuali ☺ durante le tediose lezioni di trasmissione numerica.

Un non-ringraziamento infine va agli stupidi pedanti e ai bigotti che inquinano con la loro presenza il Politecnico di Milano, rendendolo più simile a una bottega dell'ignoranza piuttosto che un luogo di scienza e apprendimento, imponendo la loro volontà di sopraffazione culturale e cercando di soffocare ogni manifestazione dello spirito. Speriamo che un giorno possano essere scacciati da questa e da qualsiasi altra università del mondo.

Time to roll now… batter's up!

Introduzione

Nel contesto dei sistemi di comunicazione radiomobili hanno assunto importanza crescente le tecniche di Voice Quality Enhancement, VQE, volte a migliorare la qualità del segnale audio. Durante le conversazioni, infatti, la qualità del segnale risulta spesso degradata, causa il frequente utilizzo dei terminali mobili in ambienti rumorosi e la comparsa di echi acustici dovuti ad accoppiamenti indesiderati nei terminali. Da qui è nata l'esigenza di implementare all'interno delle reti radiomobili blocchi funzionali adibiti a svolgere queste importanti funzioni. Il primo passo fondamentale nei processi di miglioramento della qualità audio è rivestito dall'algoritmo di rilevazione di attività vocale (*Voice Activity Detection*, VAD), il cui scopo è segnalare la presenza o meno di parlato nel segnale, al fine, in un secondo passo, di operare in maniera efficace la riduzione dei disturbi. Inoltre l'algoritmo migliora l'efficienza del sistema occupando meno risorse durante le pause normalmente presenti in una conversazione. Nella situazione attuale le operazioni di VQE nelle reti radiomobili vengono svolte all'interno dell'unità di transcodifica (TRAU) dopo la conversione dal segnale compresso secondo tecniche di codifica basate sulla predizione lineare a quello PCM utilizzato sulla rete fissa PSTN. Pertanto le tecniche note in letteratura sono prevalentemente focalizzate sull'analisi di segnali rappresentati secondo codifiche a forma d'onda. Il carattere innovativo di questo studio è stato quello di indagare la problematica di discriminazione tra voce e rumore e di riduzione del rumore nell'ambito dei segnali codificati secondo tecniche a predizione lineare (ACELP).

Le tecniche VAD e di cancellazione d'eco acustico su segnali compressi mediante codifica ACELP trovano infatti applicazione nei moderni sistemi di comunicazioni radiomobili GSM e UMTS dove si rende necessario limitare l'introduzione di transcodifiche supplementari al fine di ottimizzare l'occupazione della banda, aumentare l'efficienza delle risorse computazionali, diminuire il ritardo complessivo nel trasporto dei segnali ed infine mitigare il degrado introdotto dalla presenza di transcodifiche multiple. Le figure 1 e 2 mostrano rispettivamente gli elementi

del terminale mobile e della rete radiomobile coinvolti nella situazione attuale in cui gli algoritmi di miglioramento del parlato avvengono nel dominio lineare dopo la transcodifica e nella situazione futura in cui gli algoritmi di miglioramento del parlato avverranno nel dominio codificato agendo direttamente sui parametri in uscita dal codificatore.

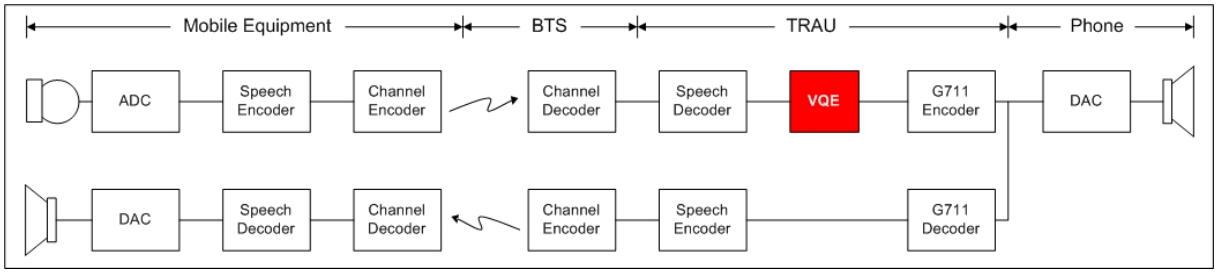


Figura 1 Situazione attuale degli elementi della rete radiomobile

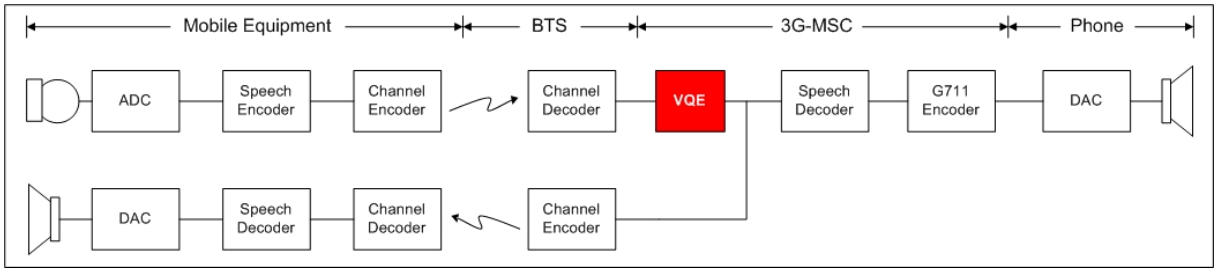


Figura 2 Situazione futura degli elementi della rete radiomobile, caratterizzata da VQE operante sul segnale codificato

In questa tesi, svolta in collaborazione con i laboratori di ricerca di *Siemens Mobile Communications*, sono stati sviluppati algoritmi di VAD e di cancellazione dell'eco acustico basati esclusivamente sui parametri del codificatore, che viaggiano nella rete radiomobile; di conseguenza il loro utilizzo non introduce nessuna decodifica supplementare e presenta allo stesso tempo un costo computazionale decisamente contenuto, rispetto ad algoritmi che effettuano una transcodifica multipla, i quali necessitano una decodifica e una ricodifica, oltre al costo computazionale necessario agli algoritmi per effettuare le decisioni di VAD e cancellazione d'eco.

Capitolo 1

La codifica vocale

Da quando Homer Dudley, più di 60 anni fa, iniziando un pionieristico lavoro sulla trasmissione della voce, riuscì a dimostrare la ridondanza del segnale vocale [11], molta ricerca è stata svolta nell'ambito dello *speech coding*. In particolare, verso la fine degli anni sessanta, il progresso delle tecniche di elaborazione numerica dei segnali iniziava a fornire sempre nuovi impulsi per continuare a migliorare e a cercare nuove soluzioni al problema. In seguito, l'avvento delle comunicazioni su fibra ottica, verso gli anni ottanta, sembrava dovesse porre fine ad una ricerca sempre continua verso la minimizzazione dell'utilizzo delle risorse disponibili, in quanto questo nuovo tipo di servizio avrebbe reso estremamente poco costose le trasmissioni numeriche e quindi anche quelle riguardanti la voce. Tuttavia, la crescita vertiginosa, soprattutto negli ultimi dieci anni, dei sistemi di comunicazione radiomobile, ha ridato nuovi impulsi e riportato grandi risorse di tipo economico e finanziario verso la ricerca di sistemi di codifica vocale sempre più innovativi, con l'obiettivo di renderli capaci di "spremere" la massima informazione possibile dal segnale vocale e a trasportarla nell'etere con il minor numero di bit possibili, portando così ad un uso ottimale della banda, risorsa estremamente preziosa per le comunicazioni radio.

Ovviamente la telefonia mobile non è l'unico mercato a richiedere codificatori vocali sempre migliori, si pensi solo al veloce sviluppo che stanno avendo la telefonia *Voice over IP* o i sistemi per la videoconferenza, tutti accomunati, insieme alla telefonia mobile, dal medesimo obiettivo: la migliore qualità con il minor uso di banda possibile.

Nel seguente capitolo si cercherà innanzitutto di chiarire il parallelismo che intercorre tra la natura della voce umana e la progettazione dei codificatori vocali [14], con la trasposizione dal modello fisico del parlato al modello matematico; saranno inoltre introdotti alcuni basilari concetti di fonetica, indispensabili per apprezzare a pieno il lavoro svolto nei successivi capitoli [12].

Una breve introduzione ai metodi di codifica a *forma d'onda* verrà svolta, soprattutto per introdurre la *Pulse Code Modulation* (PCM) [36], alla base della telefonia fissa moderna, nonché primo passo per il passaggio da dominio analogico a dominio numerico per tutti i segnali codificati in seguito con tecniche più complesse.
Il cuore del capitolo, infine, sarà l'introduzione alle tecniche più recenti di codifica vocale, in particolare, la codifica di tipo *Algebraic Code Excited Linear Prediction* (ACELP) [4] [44], stato dell'arte in fatto di codificatori vocali, alla base del codec *Adaptive Multi Rate* (AMR) [57] di cui si parlerà nel capitolo successivo. Prima dell'introduzione all'ACELP si farà un breve cenno al *Linear Predictive Coding* (LPC) [3], ormai alla base di tutti i codificatori a parametri [42] [40].
L'estrema sensibilità al rumore, agli errori e alla quantizzazione dei parametri in uscita dall'analisi LPC ha portato a modelli per la quantizzazione sempre più raffinati. Uno dei più recenti è la trasformazione dei coefficienti LPC in linee spettrali di frequenza o *Line Spectral Frequencies* (LSF) introdotte da Itakura ormai 20 anni fa [20]. Essendo questo metodo ormai largamente apprezzato per le sue qualità, nonché utilizzato dal codec AMR [58], verrà anch'esso introdotto.

1.1 La voce: dal modello fisico al modello matematico

1.1.1 Il modello fisico

Il segnale vocale è il suono generato dalle vibrazioni delle corde vocali per effetto del fiato e modulato timbricamente dal canale vocale. La figura (1.1) mostra i principali organi atti alla produzione della voce.

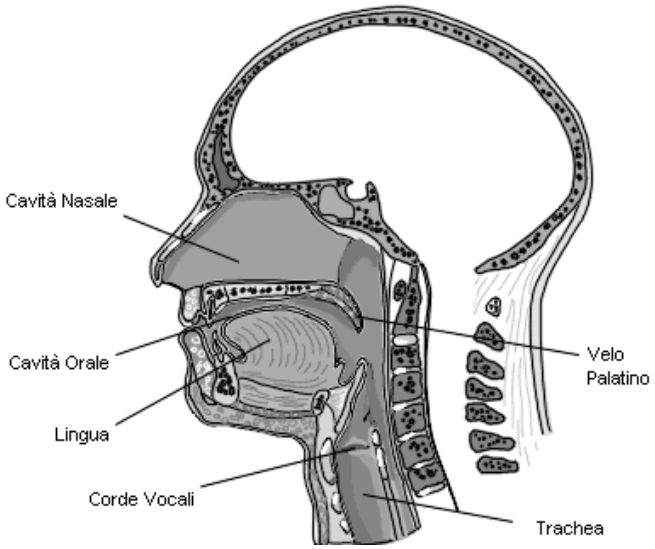


Figura 1.1 Apparato vocale umano

L'articolazione per la creazione di suoni avviene partendo dal diaframma; questo si espande e si contrae assistendo i polmoni nel forzare l'aria attraverso la trachea, penetrando così nelle corde vocali e ancora più su fino alle cavità orale e nasale dove l'interazione con la lingua, i denti e le labbra crea infine il fonema.

Gli organi dell'apparato vocale umano svolgono numerose e diverse operazioni a seconda del segnale che deve essere prodotto [51]. In particolare, le corde vocali se non sono tese faranno uscire un suono caratterizzato da uno spettro tendenzialmente uniforme, detto sordo o *unvoiced*; nell'altro caso invece, se le corde vocali sono tese, l'aria non riesce a fuoriuscire se non quando l'accresciuta pressione dei polmoni vince l'azione di occlusione delle corde vocali, le quali lasceranno uscire l'aria a "sbuffi", creando un segnale all'incirca periodico, per un suono detto sonoro o *voiced*. La frequenza di questo segnale periodico corrisponde all'inverso del tempo con il quale si ripete il fenomeno di occlusione-divaricazione delle corde vocali, questa viene detta frequenza fondamentale f_0 o *pitch frequency*, così chiamata impropriamente (pitch = tono; tone = timbro), in quanto direttamente associabile al timbro della voce e non al tono. L'intorno nel quale si muove la frequenza di pitch, durante il parlato, varia da persona a persona, ad esempio, questa sarà attorno ai 80-100 Hz per un uomo adulto con voce particolarmente roca, oppure attorno ai 250-300 Hz per voci di bambini; la frequenza può variare anche a seconda del tono che il singolo imprime alla voce, ad esempio, se si esprime un interrogazione f_0 sarà crescente col tempo, mentre, nel caso dichiarativo, decrescente.

Il modello fisico generale sarà quindi quello di un sistema riverberante, il quale sarà aperto da un lato, le labbra e le narici, ed eccitato dall'altro lato da una sorgente di aria, introdotta dai polmoni e trattata dalle corde vocali (Figura 1.2).

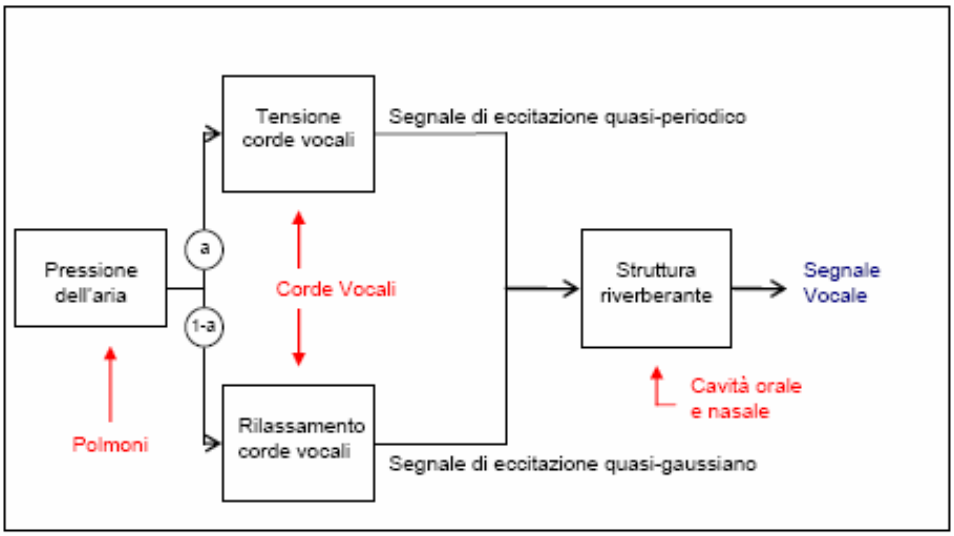


Figura 1.2 Modello fisico per la creazione del segnale vocale

Nella figura 1.2 si è tenuto conto di come le due possibili configurazioni delle corde vocali non sono nette e distinte, ma piuttosto, il segnale di eccitazione totale è una combinazione di queste. Oltre a questo, la struttura riverberante presenterà alcune frequenze di risonanza, nelle quali alcune armoniche prodotte dalle corde vocali verranno rinforzate. Queste frequenze di risonanza vengono chiamate formanti, e la loro posizione caratterizza sufficientemente le *vocali*. Per le varie lingue esistono fino a cinque frequenze di risonanza, in realtà poi in italiano, non se ne usano più di tre o quattro per ogni vocale. Nella tabella (1.1) vengono mostrate le frequenze delle prime tre formanti per i suoni vocalici della lingua italiana [43].

Vocale		Frequenza	Frequenza	Frequenza
rappresentazione	fonema	I formante	II formante	III formante
[a]	c**a**va	520	1190	2380
[é]	s**é**ta	530	1840	2480
[è]	s**è**rpe	660	1720	2410
[i]	l**i**no	270	2290	3010
[o]	s**o**le	730	1090	2440
[ò]	t**ò**ro	570	840	2410
[u]	l**u**po	440	1020	2240

Tabella 1.1 Frequenze delle formanti dei suoni vocalici italiani

L'emissione dei suoni vocalici può essere preceduta, interrotta o seguita da occlusioni o restringimenti del canale vocale, determinati dai movimenti articolatori. Questi suoni non vocalizzati, vengono detti *consonanti*. Questa seconda classificazione risulta avere numerose suddivisioni al suo interno; esisteranno infatti consonanti sibilanti, generate dal fruscio delle corde vocali quando non emettono impulsi, consonanti di carattere plosivo (ad esempio b, p, t), corrispondenti a bruschi transitori del tratto vocale, ecc.

1.1.2 Il modello matematico

Per introdurre un modello matematico valido, si può innanzitutto schematizzare la pressione dell'aria semplicemente come un valore energetico che avrà il significato di fattore moltiplicativo. Le due diverse configurazioni in cui si possono trovare le corde vocali verranno ben distinte per semplicità, l'eccitazione quasi-periodica verrà schematizzata dalla creazione di un treno di impulsi con spaziatura $1/f_0 = T_0$, ovvero l'inverso della frequenza di pitch, corrispondente ad un suono voiced; mentre nel caso di suono unvoiced, l'eccitazione delle corde vocali, verrà schematizzata da un generatore di rumore gaussiano bianco a spettro piatto [49].
L'eccitazione ora creata, sia essa corrispondente a un suono vocalizzato o sordo, verrà sagomata spettralmente da un filtro, variabile a seconda della posizione delle articolazioni, corrispondente alla struttura riverberante, modello per le cavità orale e nasale. Una buona approssimazione matematica del filtro formatore, o *shaping filter*, risulterà essere, per motivi che risulteranno chiari quando si introdurrà il metodo di analisi della predizione lineare, un filtro tutti poli. Una rappresentazione del modello matematico è mostrata in figura (1.3).

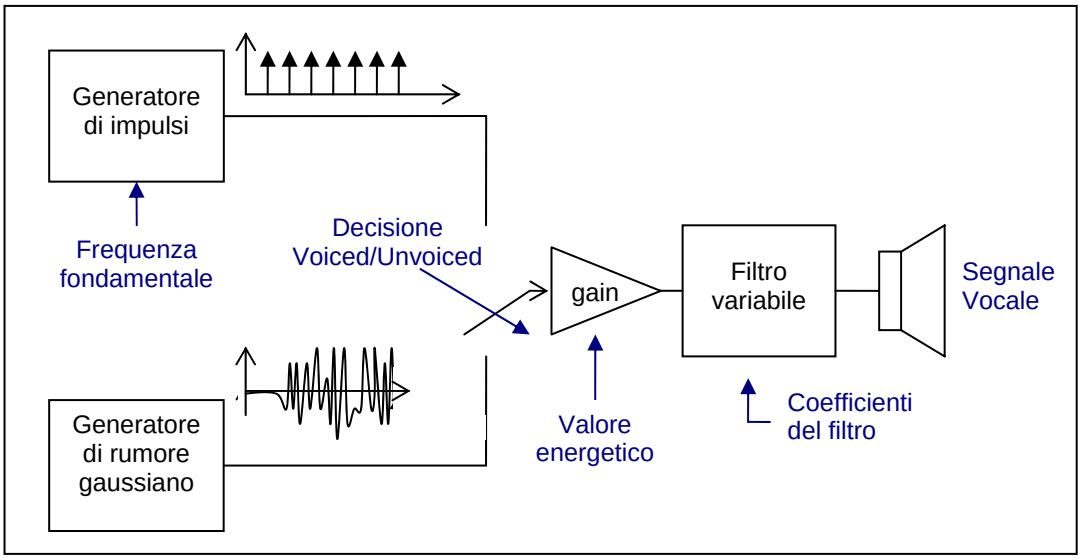


Figura 1.3 Modello matematico per la creazione del parlato

1.2 Codificatori a forma d'onda

Accantonando per il momento codificatori più complessi, che tengono conto della struttura di generazione del parlato, meritano un cenno i codificatori a forma d'onda, che si limitano a una rappresentazione discreta e quantizzata del parlato. Per questo lavoro saranno interessanti in quanto rappresentano il primo stadio per passare dal dominio analogico a quello discreto.

1.2.1 La codifica Pulse Code Modulation (PCM)

Un importante codifica audio dove il parlato è rappresentato da un numero fisso di campioni al secondo è il **PCM**. Nonostante sia stata brevettata attorno agli anni '40, è ancora il principale supporto per le reti telefoniche tradizionali. Essa applica al segnale in ingresso un filtro anti-aliasing con frequenza di taglio a 4 KHz, questo per evitare problemi con il successivo campionamento a 8 KHz. La quantizzazione viene effettuata poi con 8 bit/campione, creando un flusso di dati di 64 $Kbit/s$. In figura (1.4) è mostrato lo schema di codifica del PCM.

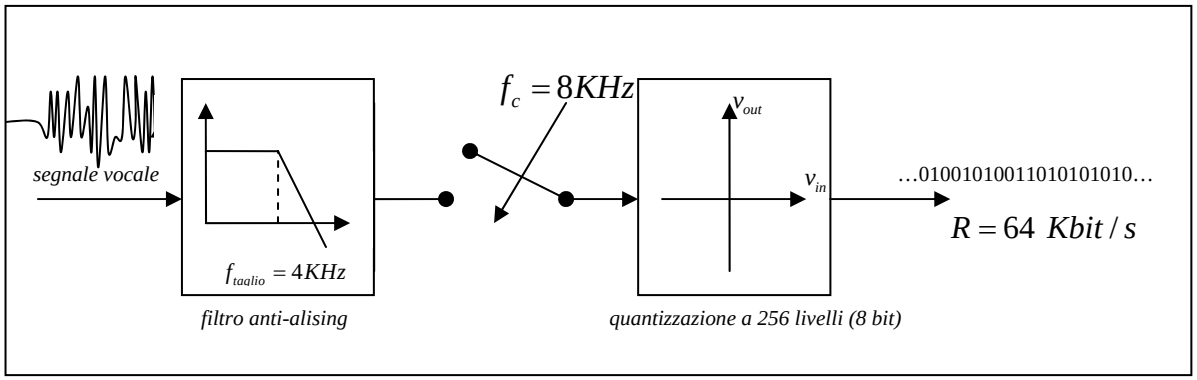


Figura 1.4 Schema di codifica PCM

La codifica PCM offre una buona qualità del parlato, con una complessità computazionale bassissima, tuttavia il rate che questo tipo di schema utilizza risulta decisamente sproporzionato.

Alcuni schemi di codifica realizzati sempre con il PCM come riferimento, studiati poco dopo l'introduzione di quest'ultimo, risultano in un uso più parsimonioso dell'informazione e quindi in una diminuzione del rate. Ad esempio, nel **ADPCM** (Adaptive Differential Pulse Code Modulation) [9] si invia la differenza tra il campione predetto al trasmettitore, e il campione attuale, utilizzando 3 o 4 bit per codificarlo. Inoltre il quantizzatore non è uniforme ma i suoi livelli sono disposti in modo da minimizzare il massimo errore possibile. La differenza tra campione attuale e campione predetto viene detto *errore di predizione*, risulterà chiaro in seguito come l'ADPCM sia stato un primo approccio verso la codifica LPC.

1.3 Linear Predictive Coding

Prima di presentare il codificatore ACELP, è necessario introdurre lo schema di analisi Linear Predictive Coding, sul quale questo si basa, ovvero la codifica della voce basata sulla predizione lineare.

L'idea alla base dell'analisi predittiva è che un campione di parlato è rappresentabile come combinazione lineare dei campioni ad esso precedenti, ovvero:

$$s(n) \approx \hat{s}(n) = a_1 s(n-1) + a_2 s(n-2) + ... + a_p s(n-p) = \sum_{k=1}^{p} a_k s(n-k) \qquad (1.3.1)$$

dove $s(n)$ rappresenta il campione vero all'n-esima posizione, mentre $\hat{s}(n)$ rappresenta la sua stima, combinazione lineare dei campioni precedenti.

Indicheremo ora con $\varepsilon_p(n)$ l'*errore di predizione* corrispondente ad un predittore di ordine P e cioè realizzato con P coefficienti:

$$\varepsilon_p(n) = s(n) - \hat{s}(n) = s(n) - \sum_{k=1}^{p} a_k s(n-k) \tag{1.3.2}$$

La determinazione dei coefficienti a_k viene fatta in modo ottimo, minimizzando il valore atteso del quadrato dell'errore:

$$E\left[\left|\varepsilon_p(n)\right|^2\right] = E\left[\left(s(n) - \sum_{k=1}^{p} a_k s(n-k)\right)\left(s(n) - \sum_{k=1}^{p} a_k s(n-k)\right)^*\right] \tag{1.3.3}$$

quindi basterà minimizzare rispetto ad essi il valore dell'errore quadratico medio e quindi imporre che:

$$\begin{aligned}\frac{\partial E\left[\left|\varepsilon_p(n)\right|^2\right]}{\partial a_k} &= 2E\left[\left(s(n) - \sum_{k=1}^{p} a_k s(n-k)\right)s^*(n-i)\right] = \\ &= 2E\left[\varepsilon_p(n)s^*(n-i)\right] = 0, \qquad \forall i = 1,..., p\end{aligned} \tag{1.3.4}$$

Queste equazioni corrispondono alla **condizione di ortogonalità**. Se l'errore di predizione è minimo, deve essere incorrelato con i dati utilizzati per calcolarlo; infatti, se ci fosse correlazione, si contraddirebbe l'ipotesi che l'errore sia minimo, in quanto i dati potrebbero essere utilizzati meglio per ridurre ancora di più l'errore. Le equazioni (1.3.4) vengono dette equazioni di Yule-Walker, dai ricercatori che per primi le utilizzarono, potremo poi riscriverle in questo modo:

$$E\left[\varepsilon_p(n)s^*(n-i)\right] = r_i + \sum_{k=1}^{p} a_k r_{i-k} = 0, \qquad \forall i = 1,..., p \tag{1.3.5}$$

dove, ignorando l'operatore di coniugio essendo $s(n)$ reale, abbiamo che:

$$r_i = E\left[s(n)s(n-i)\right] \quad \text{con} \quad i = 0,..., p-1 \tag{1.3.6}$$

è l'autocorrelazione del segnale. Possiamo vedere le equazioni di Yule-Walker anche in forma matriciale, considerando la simmetria dell'autocorrelazione di un segnale reale:

$$\begin{bmatrix} r_0 & r_1 & r_2 & ... & r_{p-1} \\ r_1 & r_0 & r_1 & ... & r_{p-2} \\ r_2 & r_1 & r_0 & ... & r_{p-3} \\ ... & ... & ... & ... & ... \\ r_{p-1} & r_{p-2} & r_{p-3} & ... & r_0 \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ ... \\ a_p \end{bmatrix} = - \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ ... \\ r_p \end{bmatrix} \tag{1.3.7}$$

Per risolvere questo sistema, si usa solitamente un algoritmo dai costi computazionali molto contenuti, l'algoritmo di Levinson-Durbin [29], che verrà spiegato al capitolo successivo in quanto usato per l'implementazione del codec AMR.

L'analisi spettrale autoregressiva, ci fornisce un metodo per calcolare un filtro tutti poli che applicato ad un rumore bianco, ne sagomi lo spettro in modo da riprodurre lo spettro della sequenza esaminata. La predizione lineare, a quanto osservato finora, non fa altro che applicare gli stessi metodi per il procedimento inverso, ovvero, sbiancare il segnale tramite un filtro a media mobile. Quindi i parametri a_k trovati conterranno l'informazione spettrale del segnale in ingresso e riprodotti in un filtro IIR alimentato da rumore gaussiano bianco, ci riporteranno al segnale precedente, a patto che l'ordine della predizione sia sufficientemente alto. In figura (1.5) viene mostrato questo ragionamento, si può dimostrare che, nel caso in cui $P \geq N$, e_n sarà statisticamente uguale w_n e quindi l'errore di predizione sarà bianco. Possiamo quindi scrivere:

$$s(n) = \sum_{k=1}^{p} a_k s(n-k) + u(n) \tag{1.3.8}$$

dove $u(n)$ è un segnale gaussiano bianco.

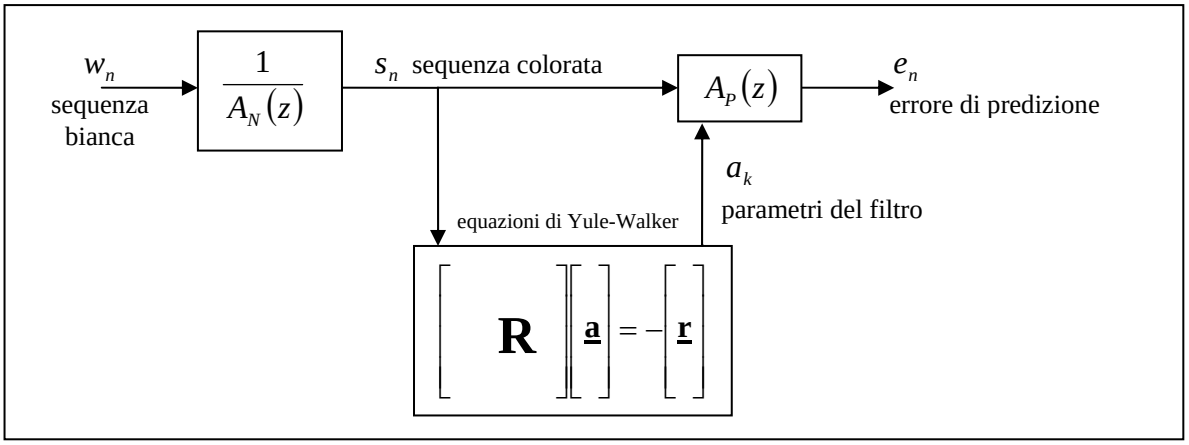


Figura 1.5 Schema di predizione

Abbiamo rivisto brevemente la teoria che sta dietro al Linear Predictive Coding, vediamo ora come si può implementare la codifica vera e propria. Dall'equazione (1.3.2) possiamo vedere l'errore di predizione come l'uscita da un sistema la cui funzione di trasferimento è:

$$A(z) = 1 - \sum_{k=1}^{p} a_k z^{-k} \tag{1.3.9}$$

un polinomio monico. Comparando questa equazione con la (1.3.8), possiamo dire che se il segnale vocale ubbidisce al modello dell'equazione (1.3.8) esattamente, allora $e(n) = u(n)$. Cosi, il filtro di predizione, $A(z)$, sarà un filtro inverso per il sistema di produzione del parlato, ad esempio:

$$H(z) = \frac{g}{A(z)} \tag{1.3.10}$$

dove g è una costante. A questo punto basterà inviare al ricevitore solo i coefficienti a_k del filtro di predizione $A(z)$ e un valore g che rappresenta una costante moltiplicativa. In realtà poi, pur sapendo che al ricevitore la sequenza che alimenterà il filtro $H(z)$ sarà gaussiana bianca, si cercherà di codificare anche il residuo di predizione, per non buttare via informazioni importanti (del resto il processo ha un numero infinito di realizzazioni, si cercherà di identificare almeno quale realizzazione si avvicina di più a $e(n)$).

L'ipotesi principale per applicare l'analisi LPC al parlato è il fatto che questo si mantenga all'incirca stazionario per un periodo di tempo; si può dimostrare che avendo in ingresso un segnale campionato a 8 KHz, la stazionarietà sarà mantenuta per circa 80-130 campioni, ovvero 10-16 ms. Quindi, una volta segmentato il segnale in intervalli di 80-130 campioni, verrà applicata a ciascun vettore un'analisi di predizione lineare, con un numero di coefficienti a_k solitamente variabile tra 8 e 14, (lo standard LPC-10 ne utilizza 10 così come il codec AMR).

1.4 La codifica ACELP

Come si diceva nell'introduzione a questo capitolo, molta strada è stata fatta nella ricerca sui codificatori vocali, la codifica ACELP non rappresenterà il punto di arrivo, tuttavia al presente simboleggia lo stato dell'arte in fatto di codificatori vocali, permettendo di creare segnali di buona qualità con bit rate dell'ordine dei 4 $Kbit/s$. L'acronimo sta Algebraic Code Excited Linear Prediction ed è basato, come dice il nome, su algoritmi di predizione lineare. La particolarità di questo metodo sta nel come viene trattato il residuo di predizione, o errore di predizione $e(n)$, come chiamato nella sezione precedente. Infatti, nella codifica predittiva, filtrare ciascun segnale con il filtro LP, porta ad un segnale residuo. Se questo residuo fosse usato come eccitazione del filtro in ricezione, l'uscita sarebbe identica all'originale segnale finestrato. Se l'eccitazione si avvicinasse di molto al residuo di predizione, allora il segnale ricostruito avrebbe comunque una qualità alta. L'innovazione della codifica ACELP è avere già a disposizione delle possibili eccitazioni e inviare quella che più si avvicina. Quindi, non sarà più necessario spedire il residuo, ma solo un indirizzo ad una parola in un elenco o *codebook*, ovviamente noto anche al ricevitore. Prima di introdurre l'ACELP, introdurremo i

principi dei codificatori *Analysis-by-Synthesis* [42], una famiglia di codificatori (della quale fa parte anche l'ACELP) che applica strumenti più raffinati della semplice analisi LPC, pur basandosi su questa.

1.4.1 Codificatori Analysis-by-Synthesis

La codifica ACELP si pone in un panorama più ampio di codificatori Analysis-by-Synthesis, questi codificatori si appoggiano tutti sullo schema LPC, tuttavia in essi si pone il problema di una buona qualità del parlato, quindi useranno alcune operazioni per trattare anche il residuo di predizione. Infatti la codifica LPC, vede solo due tipi di eccitazione possibili, un treno di impulsi distanziati di T_0 secondi (tempo di pitch) nel caso di suono vocalico, oppure un segnale gaussiano bianco nel caso di suono sordo, cioè in presenza di consonanti (vedi figura 1.3). Non si tiene conto del fatto che questa distinzione non è netta bensì mostra numerose sfumature. A questo proposito, i codificatori Analysis-by-Synthesis usano due filtri di predizione, uno a breve termine, per eliminare la prima ridondanza tra campioni vicini, generata dalle tipiche variazioni dello spettro del segnale vocale (le formanti), e uno a lungo termine per trovare una possibile ridondanza tra campioni lontani dovuta alla presenza di un segnale di pitch [33] (figura (1.6)).

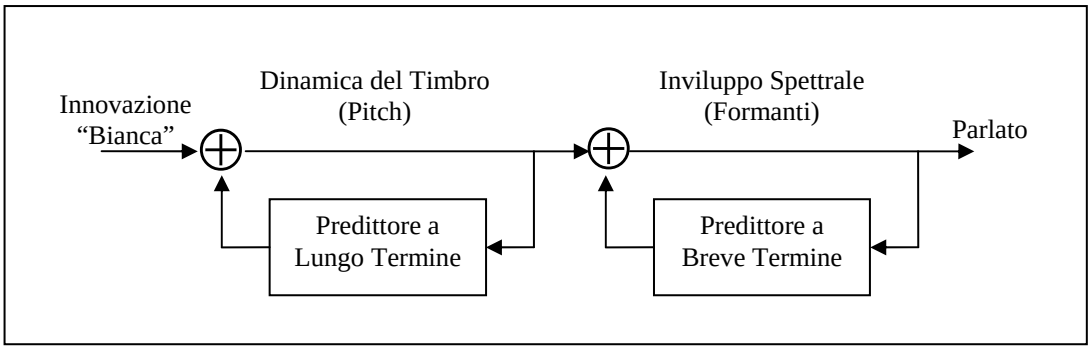


Figura 1.6 Filtri LPC in cascata per lo schema Analysis-by-Synthesis

L'azione combinata dei due predittori permette di avere come risultato un residuo di predizione idealmente gaussiano bianco che può essere rappresentato e trasmesso in modo efficiente. In figura (1.7) si mostra un esempio del codificatore Analysis-by-Synthesis. Si noti che la pesatura dell'errore viene effettuata per motivi psicoacustici, infatti, si cercherà di distribuire l'errore dove è meno percettibile per l'udito umano.

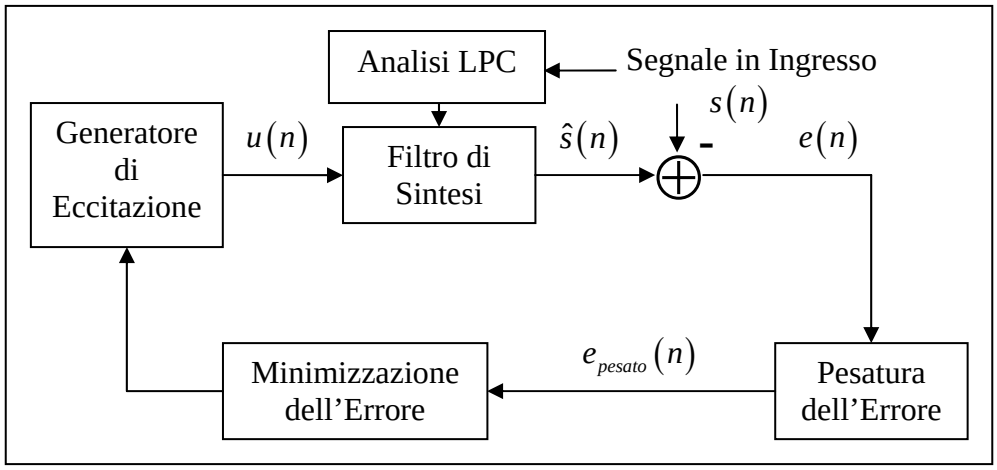


Figura 1.7 Schema a blocchi di un codificatore Analysis-by-Synthesis

1.4.2 La predizione a lungo termine

Avendo introdotto il predittore a lungo termine nella sezione precedente, ne daremo qui di seguito una giustificazione matematica. Come abbiamo detto, il predittore a lungo termine, serve ad eliminare la correlazione di lungo termine relativa alla dinamica del timbro: viene infatti chiamato anche *predittore di pitch*. Posto in cascata al predittore a breve termine, rende il segnale residuo un processo gaussiano bianco, anche se il tipo di suono è vocalico e quindi originato da un segnale quasi-periodico, a differenza del solo sbiancamento LPC.
La forma generale di un predittore a lungo termine è:

$$T(z)=\frac{1}{P(z)}=\frac{1}{1-P_L(z)}=\frac{1}{1-\sum_{k=m_1}^{m_2} g_k z^{-(\alpha+k)}} \tag{1.4.1}$$

Dove α è un valore nell'intorno del ritardo di pitch. I parametri α e g_k vengono determinati minimizzando l'errore residuo (ad esempio con MMSE) dopo i due predittori su un periodo di N campioni; è per questo che, di solito, per semplificare la ricerca, l'analisi si divide in due stadi: il primo trova un intorno possibile, il secondo il risultato ottimo.

Il segnale predetto da $P_L(z)=\sum_{k=m_1}^{m_2} g_k z^{-(\alpha+k)}$ sarà $\hat{s}(n)=\sum_{k=m_1}^{m_2} g_k s(n-\alpha-k)$, che definirà l'errore di predizione:

$$e(n)=s(n)-\hat{s}(n)=s(n)-\sum_{k=m_1}^{m_2} g_k s(n-\alpha-k) \tag{1.4.2}$$

Il filtro inverso $T(z)=1/\left(1-P_L(z)\right)$, relativo all'errore di predizione a lungo termine, con ingresso $w(n)$ gaussiano bianco, viene detto modello AR a lungo termine della serie; questo genererà il segnale:

$$s(n)=\sum_{k=m_1}^{m_2} g_k s(n-\alpha-k)+w(n). \tag{1.4.3}$$

La soluzione del problema è del tutto analoga al caso LPC di breve termine del paragrafo precedente, dato che si basa sul metodo dell'auto-correlazione (assunta la stazionarietà locale della serie osservata); la sola differenza formale consiste nel fatto che l'indice i varia tra $(\alpha+m_1)$ e $(\alpha+m_2)$.

Essendo in genere $m_2=-m_1\geq 0$, il sistema di m_2-m_1+1 equazioni in m_2-m_1+1 incognite potrà essere scritto come:

$$\sum_{j=\alpha+m_1}^{\alpha+m_2} g_{j-\alpha} r_{i-j}=-r_i \qquad \text{per} \quad i=\alpha+m_1,...,\alpha,...,\alpha+m_2 \tag{1.4.4}$$

ovvero, in forma matriciale:

$$\begin{bmatrix} r_0 & r_1 & r_2 & ... & r_{m_2} \\ r_1 & r_0 & r_1 & ... & r_{m_2-1} \\ r_2 & r_1 & r_0 & ... & r_{m_2-2} \\ ... & ... & ... & ... & ... \\ r_{m_2-1} & r_{m_2-2} & r_{m_2-3} & ... & r_0 \end{bmatrix} \cdot \begin{bmatrix} g_{m_1} \\ ... \\ ... \\ ... \\ g_{m_2} \end{bmatrix} = - \begin{bmatrix} r_{\alpha+m_1} \\ ... \\ ... \\ ... \\ r_{\alpha+m_2} \end{bmatrix} \tag{1.4.5}$$

La matrice è facilmente invertibile, e quindi il sistema è facilmente risolvibile, per lunghezza ridotta del predittore a lungo termine. In genere sarà così, infatti α varia tra 30 e 100 (valore in campioni, corrispondenti a frequenze fondamentali comprese tra $270\,Hz$ e $80\,Hz$), mentre l'ordine k è solitamente 1 o 2, ma ancora più frequentemente 0 (situazione con un solo coefficiente!).

Il predittore a lungo termine è fondamentale nei codificatori a basso bit-rate, come ACELP, dove il segnale di eccitazione è modellato da un processo gaussiano; infatti, la presenza del predittore di pitch assicurerà che la predizione residua sia molto simile a un processo gaussiano.

1.4.3 ACELP: funzionamento

Il codificatore ACELP si basa su uno schema di funzionamento CELP, *Code Excited Linear Prediction*, e differisce da quest'ultimo solo nel modo in cui viene scelta l'eccitazione relativa

al segnale uscente dai due blocchi di analisi predittiva (LPC e LTP). La figura 1.8 mostra il funzionamento del codificatore ACELP.

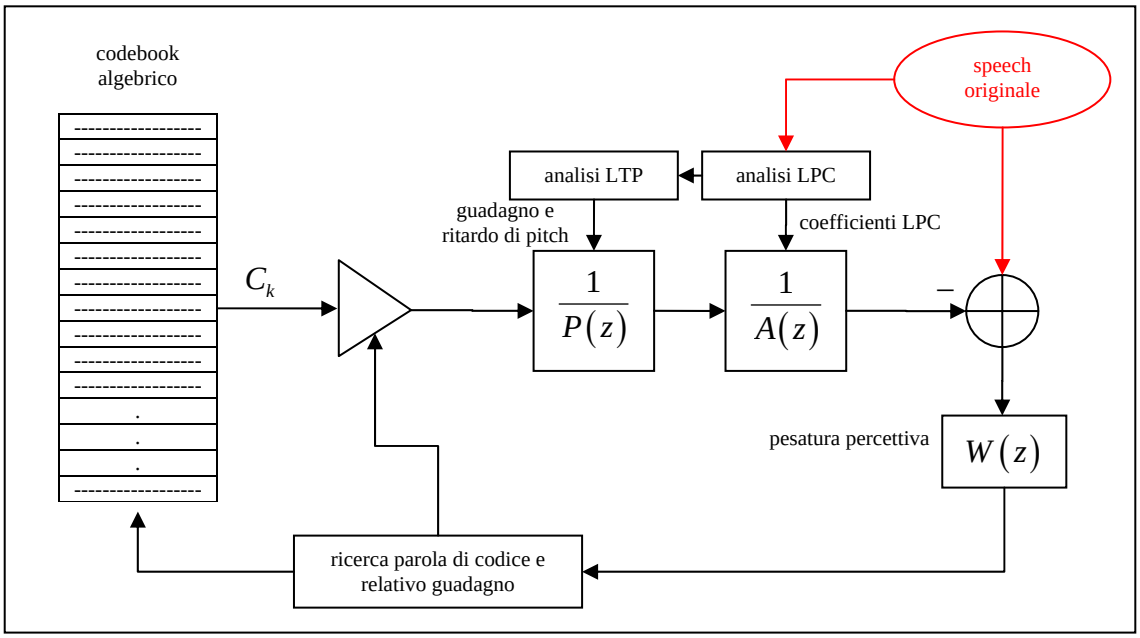


Figura 1.8 Funzionamento del codificatore ACELP

Il sistema solitamente riceve in ingresso segmenti temporali su cui svolgere l'analisi di durata variabile, campionati con frequenza variabile e quantizzati con un numero di livelli variabile; per semplicità, ci riferiremo ai medesimi valori utilizzati nel codec AMR.

Il sistema riceve in ingresso un segnale a $128\,Kbit/s$, risultante da una frequenza di campionamento di 8 KHz e una quantizzazione a $2^{16} = 65536$ livelli, ovvero con 16 bit. Il codec ACELP lavorerà su segmenti temporali di 5 ms, corrispondenti a 40 campioni, alla frequenza di campionamento considerata.

Il primo passo sarà l'analisi LPC, affrontata largamente nella sezione 1.3, il numero di parametri LPC utilizzato sarà 10, creando così un filtro formatore del tipo:

$$H(z) = \frac{1}{1 - a_1 z^{-1} - a_2 z^{-2} - \ldots - a_{10} z^{-10}} = \frac{1}{1 - \sum_{k=1}^{10} a_k z^{-k}} = \frac{1}{A(z)} \tag{1.4.6}$$

Il secondo passo sarà togliere dal segnale residuo di predizione $e(n)$ la correlazione di pitch, si sceglie un filtro con una sola presa, lavorando sul segmento di residuo corrente di 40 campioni e su 103 campioni relativi all'eccitazione passata. Lavorando come visto alla sezione 1.4.2, si otterrà il seguente filtro di predizione a lungo termine:

$$T(z) = \frac{1}{1 - g_p z^{-T_0}} = \frac{1}{P(z)} \tag{1.4.7}$$

Si noti che la procedura diventa estremamente onerosa a livello di calcolo per un'analisi così dettagliata: semplificazioni saranno effettuate, come risulterà chiaro dall'analisi di pitch del codec AMR.

Una volta decorrelato il segnale dal pitch, avremo come segnale residuo $e'(n)$ una realizzazione praticamente bianca anche per segnali vocalici. Ovviamente il segnale $e'(n)$ dipenderà anche dalle realizzazioni dell'eccitazioni di pitch precedenti, cadendo queste anche nel segmento temporale corrente.

L'ultimo passo, che contraddistingue la codifica ACELP, è la ricerca nel codebook algebrico, questa avviene basandosi sul segnale residuo che sarà pari alla differenza tra il segnale sintetizzato $\hat{s}(n)$ e il segnale naturale $s(n)$ analizzato, nel segmento temporale considerato, $n = k,...,k+39$. La differenza tra questi due segnali viene filtrata con un filtro $W(z)$ che deriva direttamente dalla modifica dei parametri LPC; questo viene fatto per mascherare l'errore, ponendolo dove è meno percettibile per l'orecchio umano.

Ora, il segnale residuo di 40 campioni dovrà essere confrontato con ciascuna parola di codice per trovare quella più simile. Utilizzando AMR 35 bit per la parola e 4 bit per il loro guadagno, il confronto dovrebbe essere effettuato su 2^{35} possibili parole, ovvero più di 34 miliardi di correlazioni. Ovviamente questo è un costo di calcolo oltremodo alto, quindi si useranno codici algebrici per ridurne la complessità [1]. La procedura verrà esposta nel capitolo successivo. Il codebook algebrico viene detto anche *codebook ad eccitazione sparsa*, infatti le parole del codice saranno formate principalmente di zeri (solo 10 valori non nulli su 40). Le parole di codebook hanno i campioni non nulli con i soli due valori: ± 1. Da questi due indizi su come è formata una parola del codebook, si può già intuire che ci sarà la possibilità di un notevole risparmio computazionale.

1.5 Le linee spettrali di frequenza

Nell'introduzione al capitolo, si è parlato della quantizzazione dei parametri LPC come una delle parti più critiche nella realizzazione di codificatori vocali per comunicazioni mobili. Essi infatti concentrano gran parte dell'informazioni relativa ad un segmento di parlato in pochi termini.

Uno strumento per misurare l'affidabilità di un metodo di quantizzazione è la *sensitività* delle radici di un polinomio rispetto ai suoi coefficienti, rappresentata dalla seguente equazione:

$$\left|\frac{\delta z_h}{\delta a_k}\right| = \left|\frac{z_h^{N-k}}{a_N \prod_{n\neq h}^{N}\left(z_h - z_n\right)}\right| \tag{1.4.8}$$

Si vede pertanto che la sensitività non è nient'altro che il fattore di proporzionalità che lega lo spostamento di una radice alla variazione di un coefficiente, è tanto maggiore quanto è maggiore il grado del polinomio e quanto più sono vicine le altre radici alla radice considerata. Questo ci può dare già delle indicazioni su come la quantizzazione diretta dei coefficienti del filtro di predizione $A(z) = 1 + \sum_{k=1}^{p} a_k z^{-k}$, non sia una buona idea, a meno che non si voglia usare un numero molto elevato (e comunque sproporzionato) di livelli.

In letteratura troviamo un grande numero di soluzioni al problema della quantizzazione dei parametri LPC, ad esempio i *Logaritmic Area Ratios*, LAR [15], usati nel sistema GSM, dove i coefficienti di riflessione determinati dagli a_k tramite il metodo di Schur [SCHUR], vengono quantizzati in questo modo:

$$LAR_k = round\left(\alpha_k \cdot \frac{1-c_k}{1+c_k} + \beta_k\right)$$

dove α_k e β_k sono scelti a seconda del c_k in considerazione.

Un altro metodo, che offre prestazioni decisamente migliori, è la conversione dei parametri LPC in *Line Spectral Frequencies,* LSF, o *Line Spectral Pairs,* LSP. Questi due tipi di conversione sono solo due modi diversi di rappresentare la stessa cosa, quindi verranno presentati insieme; la differenza risulterà chiara durante la trattazione.

Per semplificare la trattazione si suppone che il polinomio da trasformare sia $A(z) = 1 + \sum_{k=1}^{10} a_k z^{-k}$, ovvero quello che si trova in uscita dall'analisi LPC-10 utilizzata dal codec AMR.

Gli LSP sono definiti come le radici dei polinomi somma e differenza $P'(z)$ e $Q'(z)$:

$$\begin{aligned} P'(z) &= A(z) + z^{-11}A\left(z^{-1}\right) \\ Q'(z) &= A(z) - z^{-11}A\left(z^{-1}\right) \end{aligned} \tag{1.4.9}$$

essi saranno quindi legati ad $A(z)$ dalla seguente relazione:

$$A(z) = \frac{P'(z) + Q'(z)}{2} \tag{1.4.10}$$

I due polinomi, hanno due radici fisse in $z = -1$ e $z = 1$ rispettivamente, quindi è possibile eliminarle, definendo i due nuovi polinomi:

$$P(z) = \frac{P'(z)}{1+z^{-1}}$$
$$Q(z) = \frac{Q'(z)}{1-z^{-1}} \qquad (1.4.11)$$

Le radici dei polinomi nell'equazione (1.4.11) sono gli LSP. Essi presentano importanti proprietà [27]:

- tutti gli zeri dei due polinomi sono disposti sul cerchio unitario;
- gli zeri di $P(z)$ e $Q(z)$ sono inter-allacciati, ovvero gli zeri dell'uno e l'altro polinomio si alternano sul cerchio unitario;
- $P(z)$ e $Q(z)$ sono simmetrici;
- se $A(z)$ è a minima fase, questa proprietà sarà mantenuta anche nei due polinomi (che infatti hanno radici sul cerchio unitario).

Un'altra proprietà importante, da cui il nome LSP, è che le radici dei due polinomi sono tutte complesse coniugate, quindi sarà necessario rappresentare solo metà del cerchio unitario, e sfruttare le simmetrie. Questa proprietà è stata usata nella figura 1.9 in cui sono mostrate solo le radici di $P(z)$ e $Q(z)$ appartenenti al dominio $(0 \quad \pi)$, le altre saranno speculari a quelle rappresentate nel dominio $(-\pi \quad 0)$.

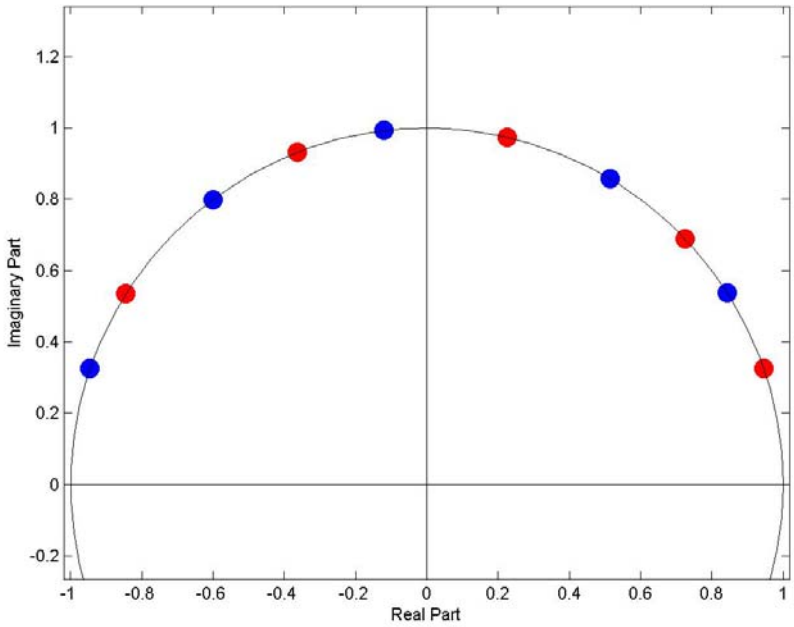


Figura 1.9 Disposizione delle radici di $P(z)$ (cerchi blu) e $Q(z)$ (cerchi rossi) appartenenti al dominio $(0 \quad \pi)$

Si è detto che gli LSP si dispongono sul cerchio unitario $z = e^{j\phi_i}$, quindi possono essere espressi solo in funzione della loro fase, ovvero di ϕ_i. Le fasi convertite dal dominio Z al dominio delle frequenze, cioè $f_i = 4000 \cdot \phi_i / \pi$ (supponendo che $f_c = 8000 Hz$), saranno chiamate linee spettrali di frequenza o LSF.

Per concludere, lo studio svolto presso l'Acoustic Research Department dei Bell Laboratoires da Frank Soong e Bing-Hwang Juang [48] ha dimostrato che la codifica della diffrerenza tra la posizione della $i-$esima linea spettrale di frequenza e quella direttamente successiva ($i+1-$esima) ha proprietà statistiche di media e varianza molto interessanti per la quantizzazione, offrendo migliori prestazioni rispetto ai LAR. Lo schema a blocchi del codificatore proposto dai due studiosi è mostrato in figura 1.10.

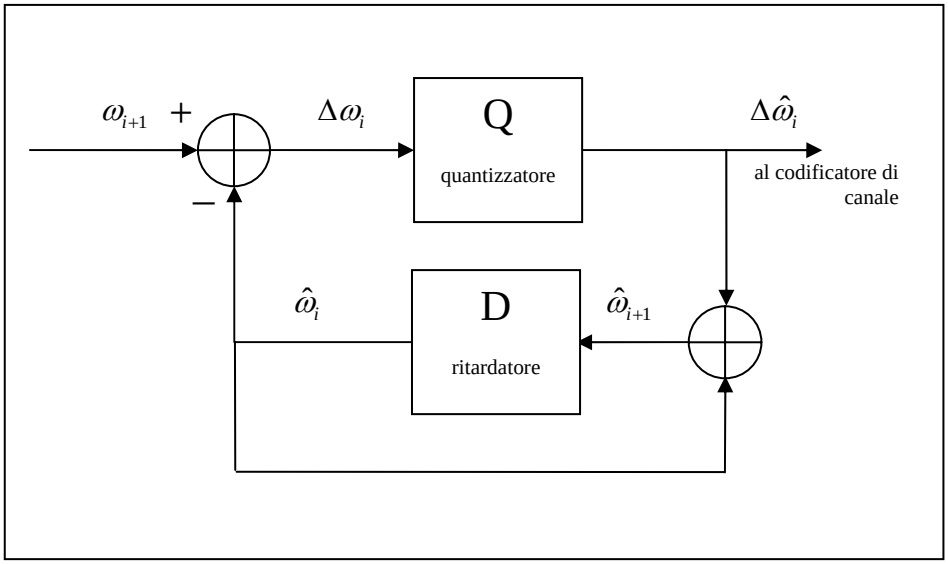


Figura 1.10 Schema a blocchi del codificatore per LSP proposto da Frank Soong e Bing-Hwang Juang

Nel terzo capitolo verranno affrontate le statistiche degli LSF e il loro significato fisico verrà analizzato più a fondo. In particolare, lo studio effettuato metterà alla luce che gli LSP o LSF risultano non solo un robusto metodo di traformazione di parametri per la quantizzazione, ma offrono anche importanti caratteristiche fisiche che li rendono particolarmente appetibili anche per campi di ricerca come lo *speech recognition*.

Capitolo 2

Il codec AMR

Il codec Adaptive Multi Rate, rappresenta lo stato dell'arte in fatto di codificatori vocali per reti radiomobili. Esso è stato infatti introdotto come standard per la terza generazione di telefonia mobile UMTS dalla 3GPP, 3rd Generation Partnership Project, accordo di collaborazione di numerose organizzazioni internazionali per la regolamentazione degli standard quali, ad esempio, la ETSI (European Telecommunications Standards Institute), la ARIB/TTC (Giapponese), la CCSA (Cinese), la ATIS (Nord-Americana) e la TTA (Sud-Coreana).

Questo capitolo pone le basi per la comprensione del lavoro svolto, infatti, dopo aver introdotto la codifica ACELP nel precedente capitolo, affronteremo nel seguito come il segnale viene effettivamente trattato, ovvero con l'encoder e il decoder AMR basati appunto su schemi ACELP. L'Adaptive Multi Rate è un sistema di codifica di tipo multimodale, offrendo otto diverse frequenze di cifra possibili (12.2 $Kbit/s$, 10.2 $Kbit/s$, 7.95 $Kbit/s$, 7.40 $Kbit/s$, 6.70 $Kbit/s$, 5.90 $Kbit/s$, 5.15 $Kbit/s$, 4.75 $Kbit/s$), con la possibilità, durante la trasmissione, di cambiare da un rate a un altro tramite un'interazione continua con la codifica di canale. Questa operazione è chiamata DTX (Discontinous Transmission) ed è una delle opzioni che offre il codec AMR.

Per uno studio più incentrato sull'analisi dei parametri, il codec AMR viene qui presentato nella sua versione a 12.2 $Kbit/s$ (AMR 122) che rappresenta la frequenza di cifra più alta con il quale questo lavora.

Le operazioni principali dell'encoder AMR sono: l'analisi LPC, l'analisi relativa all'eccitazione di pitch e quella relativa al residuo finale con la ricerca nel codebook algebrico. Il codec lavora su trame di 160 campioni (20 ms), ulteriormente divisi in quattro sottotrame di 40 campioni (5 ms).

Le operazioni del decoder AMR sono molto simili all'inverso delle operazioni svolte dall'encoder, o addirittura sono le stesse operazioni come nel caso dell'interpolazione dei parametri LPC, quindi, per brevità, se ne darà solo un veloce cenno.
Non si parlerà invece della quantizzazione dei parametri, a parte quella riguardante gli LSF, in quanto complicata e inutile ai nostri scopi futuri, più analitici che di calcolo.

2.1 Pre-processing ed analisi LPC

L'encoder AMR riceve in ingresso un segnale PCM con rate 128 $Kbit/s$, a seguito di un campionamento a 8000 KHz e una quantizzazione a 65536 livelli (16 $bit/campione$).
Prima di iniziare l'elaborazione, sul segnale PCM viene effettuato un filtraggio passa alto per eliminare le componenti a bassa frequenza che potrebbero disturbare le successive manipolazioni del segnale. Viene imposta una frequenza di taglio di 80 KHz. Questa operazione viene unita ad un'altra operazione di down-scaling, moltiplicando il filtro per un fattore ½, effettuata per evitare la possibilità di avere overflow.
Dopo il filtraggio passa-alto a 80 kHz, vengono effettuate due finestratura sul frame, che includono anche 80 campioni del frame precedente usando due finestre asimmetriche (vedi figura (2.1)).

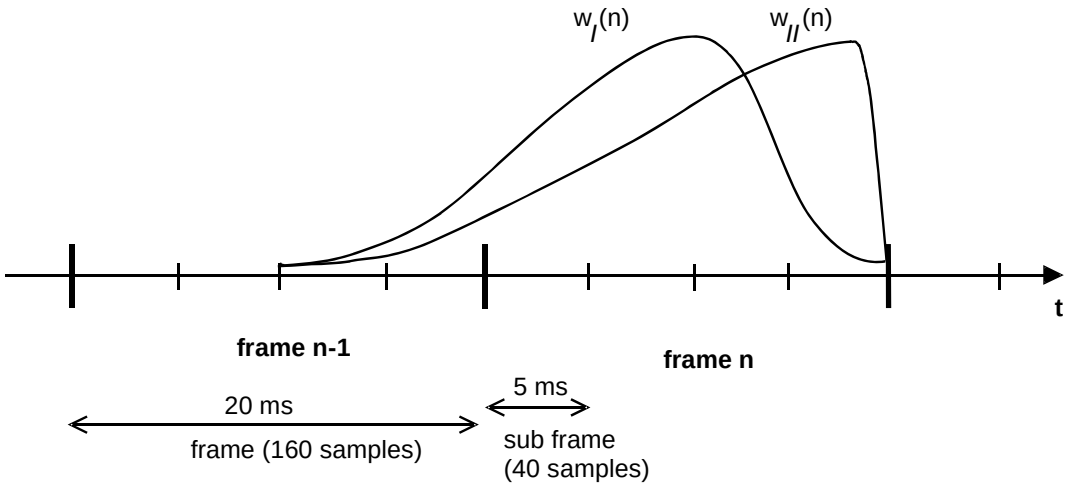


Figura 2.1 Finestre per l'analisi LPC

Si noti che la prima finestra avrà i pesi concentrati sul secondo subframe, mentre la seconda finestra avrà i suoi pesi concentrati sul quarto subframe. Questo risulterà chiaro nel seguito, infatti i parametri LPC del primo e del terzo subframe verranno calcolati come combinazione di quelli ottenuti dal secondo e quarto subframe.

2.2.1 Calcolo dell'autocorrelazione

L'analisi LPC verrà eseguita con il metodo dell'autocorrelazione visto al capitolo precedente. Le due autocorrelazioni per ciascuna sequenza finestrata, $s'(n)$, $n = 0,...,239$, verranno calcolate in questo modo:

$$r_{ac}(k) = \sum_{n=k}^{239} s'(n)s'(n-k) , \qquad k = 0,\ldots,10, \tag{2.1.1}$$

dopodiché si applica un'espansione di banda di 60 Hz finestrando l'autocorrelazione con la finestra:

$$w_{lag}(i) = \exp\left[-\frac{1}{2}\left(\frac{2\pi f_0 i}{f_s}\right)^2\right], \quad i = 1,\ldots 10 \tag{2.1.2}$$

dove $f_0 = 60$ Hz è l'espansione di banda e $f_S = 8000$ Hz è la frequenza di campionamento. Il primo campione dell'autocoorelazione invece verrà moltiplicato per un fattore 1.0001, equivalente ad aggiungere un noise floor di -40 dB.

2.1.2 Algoritmo di Levinson-Durbin per il calcolo dei parametri LPC

I due set di autocorrelazione modificata $r'_{ac}(0) = 1.0001 \cdot r_{ac}(0)$ e $r_{ac}(k) = r_{ac}(k)w_{lag}(k)$, $k = 1,...,10$, saranno ora usati per ottenere i coefficienti di predizione lineare a_k, $k = 1,...,10$, risolvendo il sistema di equazioni:

$$\sum_{k=1}^{10} a_k r'_{ac}(|i-k|) = -r'_{ac}(i) , \qquad i = 1,\ldots,10. \tag{2.1.3}$$

Il sistema dell'equazione (2.1.3) è risolto usando l'algoritmo di Levinson-Durbin, essendo questo computazionalmente più semplice rispetto a trovare la matrice inversa di $r_{ac}(k)$. L'algoritmo trova il polinomio predittore $A_{L+1}(z)$ partendo dal polinomio $A_L(z)$ utilizzando l'*errore di predizione all'indietro*, ovvero, l'innovazione apportata del campione di posto $i - N - 1$ rispetto agli altri campioni già utilizzati per la predizione. In pratica, si utilizzeranno le condizioni di ortogonalità per avere una progressiva identificazione dei coefficienti di riflessione k_i della struttura riverberante corrispondente. L'algoritmo usa la seguente ricorsione:

$E_{LD}(0) = r_{ac}'(0)$
for $i = 1$ to 10 do
$a_0^{(i-1)} = 1$
$k_i = -\left[\sum_{j=0}^{i-1} a_j^{(i-1)} r_{ac}'(i-j)\right] / E_{LD}(i-1)$
$a_i^{(i)} = k_i$
for $j = 1$ to $i-1$ do
$a_j^{(i)} = a_j^{(i-1)} + k_i a_{i-j}^{(i-1)}$
end
$E_{LD}(i) = (1 - k_i^2) E_{LD}(i-1)$
end

La soluzione finale è data dai $a_j = a_j^{(10)}$, $j = 1,...,10$.

I coefficienti del filtro di sintesi a_k verranno ora convertiti in LSP.

2.1.3 Conversione dei parametri LPC in Line Spectral Pairs

I coefficienti LPC vengono convertiti in Line Spectral Pairs per motivi di quantizzazione ed interpolazione, infatti dai due set di LSP relativi al secondo e quarto subframe, verranno trovati gli altri due, relativi al primo e terzo subframe. Rivediamo brevemente come avviene questa trasformazione.

Innanzitutto, per un filtro *Linear Prediction* di decimo ordine, gli LSP sono definiti come le radici dei due polinomi:

$$F_1'(z) = A(z) + z^{-11} A\left(z^{-1}\right) \tag{2.1.4}$$

e

$$F_2'(z) = A(z) - z^{-11} A\left(z^{-1}\right) \tag{2.1.5}$$

rispettivamente. I polinomi $F_1'(z)$ e $F_2'(z)$ sono rispettivamente simmetrico ed anti-simmetrico. Si dimostra che tutte le radici di questi polinomi si trovano sul cerchio unitario e si alternano tra di loro. In più il polinomio $F_1'(z)$ avrà una radice in $z = -1$ $(\omega = \pi)$, cioè alla frequenza di Nyquist, fissa, per qualsiasi vettore di a_k; la stessa cosa accade per $F_2'(z)$, avrà una radice fissa, stavolta però in $z = 1$ $(\omega = 0)$. Si potranno quindi eliminare le due radici, ricordandosi di rimetterle in fase di decodifica:

$$F_1(z) = F_1'(z) \Big/ \left(1 + z^{-1}\right) \tag{2.1.6}$$

e

$$F_2(z) = F_2'(z) \Big/ \left(1 - z^{-1}\right) \tag{2.1.7}$$

Ciascun polinomio avrà cinque radici coniugate sul cerchio unitario $\left(e^{\pm j\omega_i}\right)$, pertanto, i polinomi possono essere scritti come:

$$F_1(z) = \prod_{i=1,3,\ldots,9} \left(1 - 2q_i z^{-1} + z^{-2}\right) \tag{2.1.8}$$

e

$$F_2(z) = \prod_{i=2,4,\ldots,10} \left(1 - 2q_i z^{-1} + z^{-2}\right) \tag{2.1.9}$$

dove $q_i = \cos(\omega_i)$ essendo ω_i le linee spettrali di frequenza (LSF). Le ω_i soddisfano la proprietà di ordinamento $0 < \omega_1 < \omega_2 < \ldots < \omega_{10} < \pi$. Ci riferiremo alle q_i come linee spettrali di frequenza nel dominio coseno.

Le proprietà dei polinomi $F_1(z)$ e $F_2(z)$, permettono un calcolo veloce e preciso delle loro radici. Il calcolo implementato nell'encoder AMR deriva dagli studi effettuati da Kabal e Ramachandran [26], non verrà riportato in quanto esula dai nostri scopi.

2.1.4 Quantizzazione degli LSP

Una volta trovati i due set di LSP relativi al frame trattato, questi vengono quantizzati usando la loro rappresentazione nel dominio delle frequenze; cioè:

$$f_i = \frac{f_S}{2\pi} arccos(q_i), \qquad i = 1,\ldots,10, \tag{2.1.10}$$

dove le f_i sono le linee spettrali di frequenza espresse in Hz [0, 4000] e $f_S = 8000Hz$ è la frequenza di campionamento. Il vettore delle LSF sarà quindi $\underline{f}^t = \left[f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10} \right]^t$ dove t indica l'operatore di trasposizione.

Viene a questo punto applicato un filtro predittore a media mobile di ordine uno e i due residui di predizione vengono quantizzati insieme usando la *split matrix quantization* (SMQ). Definiamo i due vettori $\underline{z}^{(1)}(n)$ e $\underline{z}^{(2)}(n)$ come i vettori LSF della trama n-esima ai quali è stato sottratto il valor medio. I due vettori, residui di predizione, $\underline{r}^{(1)}(n)$ e $\underline{r}^{(2)}(n)$, saranno dati da:

$$\underline{r}^{(1)}(n) = \underline{z}^{(1)} - \underline{p}(n)$$
$$\underline{r}^{(2)}(n) = \underline{z}^{(2)} - \underline{p}(n) \tag{2.1.11}$$

dove $\underline{p}(n)$ è il vettore LSF predetto per il frame n-esimo:

$$\underline{p}(n) = 0.65\underline{r}^{(2)}(n-1) \tag{2.1.12}$$

dove $\underline{r}^{(2)}(n-1)$ si riferisce al secondo set di LSF relative al frame precedente.

La quantizzazione avviene in questo modo: i vettori vengono innanzitutto messi insieme in una matrice 10x2:

$$\begin{pmatrix} r_1^{(1)} & r_1^{(2)} \\ r_2^{(1)} & r_2^{(2)} \\ \dots & \dots \\ r_9^{(1)} & r_9^{(2)} \\ r_{10}^{(1)} & r_{10}^{(2)} \end{pmatrix} \tag{2.1.13}$$

questa matrice viene ora divisa in cinque sottomatrici 2x2:

$$\begin{pmatrix} r_1^{(1)} & r_1^{(2)} \\ r_2^{(1)} & r_2^{(2)} \end{pmatrix} \quad \dots \quad \begin{pmatrix} r_9^{(1)} & r_9^{(2)} \\ r_{10}^{(1)} & r_{10}^{(2)} \end{pmatrix} \tag{2.1.14}$$

si applicherà, infine, una pesatura, per rendere minimo l'errore dovuto alla quantizzazione, verranno infine inseriti gli indici relativi agli LSP nel payload relativo alla trama con, rispettivamente per ogni matrice, 7, 8, 9, 8 e 6 bit.

2.1.5 Interpolazione degli LSP

Il secondo motivo, oltre alla facilità di quantizzazione, come dicevamo in precedenza, è la possibilità di sfruttarli per l'interpolazione. Come abbiamo detto in precedenza infatti, questi vengono usati per il calcolo dei parametri di predizione (già nel dominio degli LSP) relativi al primo e terzo subframe. L'interpolazione viene fatta sugli LSP nel dominio coseno $\underline{q}$ (vedi equazioni (2.1.8), (2.1.9), (2.1.10)). Poniamo $\underline{q}_4^{(n)}$ come il vettore di LSP relativo al quarto subframe della trama corrente, $\underline{q}_2^{(n)}$ relativo al secondo subframe, sempre della trama n-esima, ed infine $\underline{q}_4^{(n-1)}$ relativo al quarto subframe, stavolta della trama precedente (la $n-1$-esima). I vettori interpolati del primo e terzo subframe saranno:

$$\begin{cases} \underline{q}_1^{(n)} = \frac{1}{2}\underline{q}_4^{(n-1)} + \frac{1}{2}\underline{q}_2^{(n)} \\ \underline{q}_3^{(n)} = \frac{1}{2}\underline{q}_2^{(n)} + \frac{1}{2}\underline{q}_4^{(n)} \end{cases} \tag{2.1.15}$$

In questo modo avremo 4 differenti set di LSP (e quindi di a_k) per ciascuna trama.

La medesima operazione viene svolta sugli LSP già quantizzati per motivi che saranno chiari in seguito.

La figura (2.2) riassume le operazioni svolte in questa prima parte.

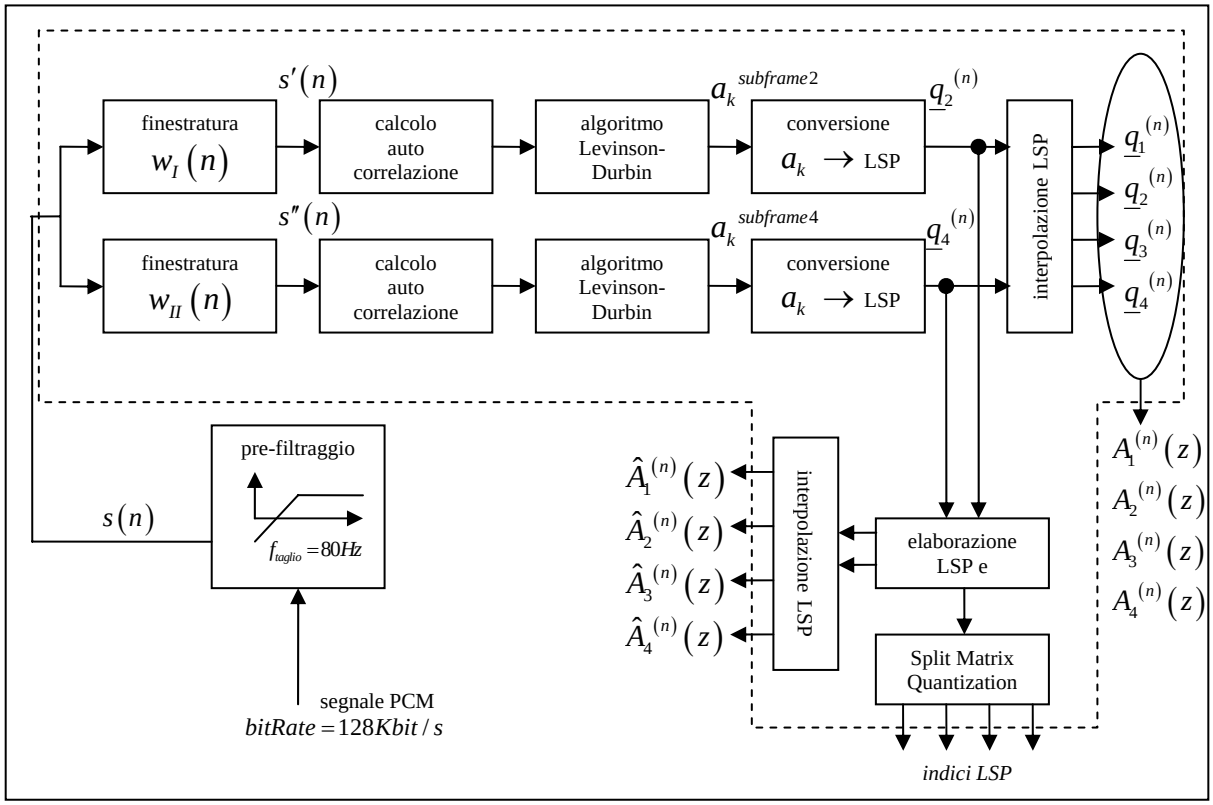


Figura 2.2 Primo stadio dell'encoder AMR: l'analisi LPC

2.2 Analisi relativa all'eccitazione di pitch

L'analisi relativa al *tracking* della frequenza fondamentale, che abbiamo visto in precedenza relativamente all'analisi LP di lungo termine (*Long Term Prediction*), nel codec AMR si divide in due parti. La prima parte consiste nel trovare un intorno di $T_0 = 1/f_0$ (ritardo di pitch), questo avviene tramite un'analisi ad anello aperto (*Open-Loop*) ed avviene tramite il calcolo dell'autocorrelazione del segnale in ingresso $s(n)$ e la scelta tra tre possibili valori di ritardo di pitch. Al passo successivo, viene svolta un'analisi ad anello chiuso (*Closed-Loop*)

in cui, partendo dall'intorno di $T_{open-loop}$ trovato al passo precedente, si cerca una stima migliore del vero valore del ritardo di pitch o *pitch lag.* Vediamo qui di seguito nel dettaglio come questo avviene.

2.2.1 Analisi Open-Loop

Come abbiamo detto, l'analisi open-loop viene svolta per semplificare l'analisi di pitch e restringere la ricerca closed-loop ad un numero minore di possibili ritardi, sempre attorno a quelli stimati dall'analisi open-loop.

L'analisi open-loop è basata sul segnale in ingresso, il quale viene prima filtrato con un *Perceptual Weighting Filter*, ovvero un filtro che tiene conto della percettività e sensibilità dell'udito umano. Infatti, criteri quali la minimizzazione dell'errore quadratico medio, male si addicono all'apparato uditivo; sarà opportuno, invece, cercare di "mascherare" l'errore nelle regioni dove l'udito è meno sensibile. A questo proposito il filtro utilizzato tenderà a modellare il rumore in modo da essere concentrato alle frequenze dove sono presenti le formanti e piccolo nelle regioni di mezzo; questo viene fatto perché le formanti corrispondono alle zone con alta energia e questa energia tenderà a "mascherare" l'errore. Il fitro utilizzato, a questo proposito, nelle specifiche AMR è:

$$W(z) = \frac{A(z/\gamma_1)}{A(z/\gamma_2)} \tag{2.2.1}$$

dove $\gamma_1 = 0.9$ e $\gamma_2 = 0.6$; il filtro $A(z)$, invece, è il filtro realizzato con gli a_k in uscita dall'algoritmo di Levinson-Durbin, interpolati nel dominio LSP e ritrasformati in a_k, ciascun set relativo ad un subframe. Questa pesatura quindi viene svolta su $s(n)$, il segnale appena uscito dal filtro passa-alto iniziale.

L'analisi ad anello aperto verrà svolta due volte per ogni frame per trovare due stime del ritardo di pitch, ciascuna relativa ai primi 10 *ms* e ai successivi 10 *ms* della trama (ricordiamo che questa ha 160 campioni ed ha durata 20 *ms*). Vediamo come l'analisi viene svolta per ciascun segmento temporale.

Il primo passo, dopo aver determinato il segnale pesato con il Perceptual Weighting Filter:

$$s_w(n) = s(n) + \sum_{i=1}^{10} a_i \gamma_1^i s(n-i) - \sum_{i=1}^{10} a_i \gamma_2^i s_w(n-i), \quad n = 0, \ldots, L-1 \tag{2.2.2}$$

dove $L = 40$ (lunghezza del subframe elaborato), sarà il calcolo dell'autocorrelazione sui 10 *ms*:

$$O_k = \sum_{n=0}^{79} s_w(n) s_w(n-k) \tag{2.2.3}$$

Ora, viene divisa l'autocorrelazione in tre segmenti, corrispondenti ai campioni di posto:

$$\begin{aligned} i &= 3: \quad 18,\ldots,35, \\ i &= 2: \quad 36,\ldots,71, \\ i &= 1: \quad 72,\ldots,143. \end{aligned} \tag{2.2.4}$$

per ciascun intervallo si vede dove è il campione di valore massimo e si memorizza il suo valore. Questi tre massimi, che chiameremo O_{t_i}, $i = 1,2,3$, saranno normalizzati per $\sqrt{\sum_n s_w{}^2 (n - t_i)}$, $i = 1,2,3$. Dopo la normalizzazione, si sceglierà quale di questi tre t_i verrà usato per l'analisi closed-loop, tramite il seguente algoritmo:

$T_{op} = t_1$
$M(T_{op}) = M_1$
if $M_2 > 0.85 M(T_{op})$
    $M(T_{op}) = M_2$
    $T_{op} = t_2$
end
if $M_3 > 0.85 M(T_{op})$
    $M(T_{op}) = M_3$
    $T_{op} = t_3$
end

Il "vincitore", T_{op}, viene scelto tra i tre valori possibili favorendo i ritardi con i valori nel terzo range (18,…,35). Si noti, infatti, nell'algoritmo la pesatura dei valori per gli altri due range.

Questa procedura di dividere il dominio dei ritardi in tre parti, favorendo quella con i ritardi più bassi, è usata per evitare di scegliere multipli del ritardo di pitch T_0. I procedimenti relativi all'analisi open loop per la ricerca del pitch lag sono riassunti nella figura (2.3).

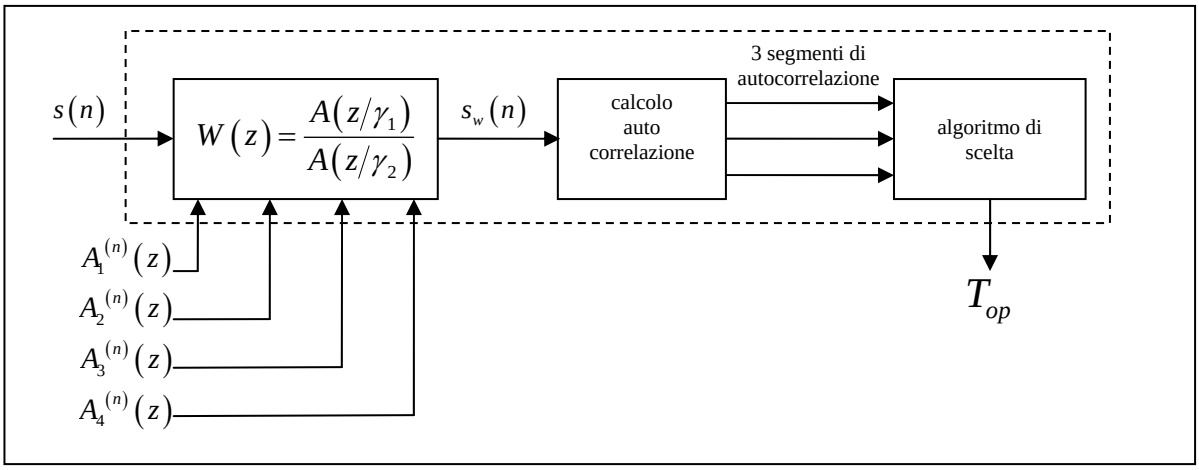


Figura 2.3 Ricerca open-loop del ritardo di pitch T_{op}

2.2.2 Analisi Closed-Loop

L'analisi ad anello chiuso viene anche chiamata *Adaptive Codebook Search*, relativamente al fatto che viene ricercata in un codebook l'eccitazione che più si addice a rappresentare la correlazione di lungo termine. La ricerca, inoltre, viene fatta adattativamente, tenendo conto di quanto sintetizzato precedentemente. Le informazioni inviate saranno quelle relative al ritardo e al guadagno del filtro relativo alla ricostruzione della dinamica del timbro.

Prima di procedere con l'analisi closed-loop e la ricerca di T_0 e g_p, il codec AMR crea due segnali che saranno utili per la ricerca di questi.

Il primo segnale $h(n)$ rappresenta la risposta all'impulso del filtro di sintesi, nella sua versione quantizzata $\hat{A}(z)$, pesato per il perceptual weighting filter (2.2.1):

$$h(n) \quad \longleftrightarrow \quad H(z)W(z) = \frac{A(z/\gamma_1)}{A(z/\gamma_2)\hat{A}(z)} \tag{2.2.5}$$

Questa operazione viene svolta per ciascun $\hat{A}_i(z)$ relativo a ciascun subframe $i = 1,2,3,4$ (figura (2.2)).

Il secondo segnale, $x(n)$, che viene utilizzato viene chiamato *target signal*, questo non è nient'altro che il segnale originale $s(n)$ ricostruito con i nuovi dati, relativi alle varie elaborazioni di $s(n)$ stesso. La procedura utilizzata per calcolare il target signal, utilizzata nel codec AMR, è il filtraggio del residuo di predizione LP

$$res_{LP}(n) = s(n) + \sum_{i=1}^{10} \hat{a}_i s(n-i). \tag{2.2.6}$$

per la combinazione del filtro di sintesi quantizzato $\hat{A}(z)$ e il perceptual weighting filter

$$H(z)W(z) = \frac{A(z/\gamma_1)}{A(z/\gamma_2)\hat{A}(z)} \tag{2.2.7}$$

Questa operazione viene svolta per ciascun subframe. In seguito, vedremo come il residuo di predizione calcolato $res_{LP}(n)$ non viene esclusivamente utilizzato per il calcolo del target signal $x(n)$, esso viene utilizzato anche per estendere il buffer dell'eccitazione passata, consentendo una semplificazione della ricerca per pitch lag inferiori alla durata di un subframe (40 campioni).

Siamo ora pronti ad affrontare l'analisi closed-loop vera e propria. Dopo aver trovato due possibili intorni, con l'analisi open-loop, del possibile pitch lag per la prima e la seconda metà del frame, l'analisi del pitch viene svolta a livello di subframe.

Nel primo e terzo subframe, viene usato un ritardo di pitch fratto con risoluzione $1/6$ nel range $\left[17+\frac{3}{6} \quad 94+\frac{3}{6}\right]$, intero invece nel range $[95 \quad 143]$. Per il secondo e quarto subframe, invece, una risoluzione di $1/6$ è sempre usata nell'intervallo $\left[T_1-\left(5+\frac{3}{6}\right) \quad T_1+\left(4+\frac{3}{6}\right)\right]$, dove T_1 è l'intero a cui il valore di pitch lag, selezionato al primo e terzo subframe, si avvicina di più; si noti che la risoluzione è fissa a $1/6$, indipendentemente dal valore selezionato al subframe precedente, i valori in uscita saranno limitati all'intervallo $[18 \quad 143]$.

L'analisi closed-loop viene svolta attorno ai valori stimati dall'analisi open-loop, però a livello di subframe. Infatti, nel primo e terzo subframe, si sceglierà di muovere la ricerca nell'intervallo $T_{op} \pm 3$ (limitato in $[18 \quad 143]$), mentre nel secondo e quarto subframe, si procederà come visto in precedenza, partendo dal valore intero di pitch lag del frame precedente.

La ricerca del pitch viene svolta minimizzando l'errore quadratico medio pesato tra il parlato originale e quello sintetizzato, questo avviene massimizzando il termine:

$$R(k) = \frac{\sum_{n=0}^{39} x(n)\, y_k(n)}{\sqrt{\sum_{n=0}^{39} y_k(n)\, y_k(n)}} \tag{2.2.8}$$

dove $x(n)$ abbiamo già visto essere il target signal, mentre $y_k(n)$ è l'eccitazione passata al ritardo k (eccitazione passata convoluta per $h(n)$).

La convoluzione $y_k(n)$ è calcolata per il primo ritardo $t_{\min}$ nel range di ricerca e, successivamente, per tutti gli altri ritardi sempre nel range di ricerca predefinito $k = t_{\min}+1,...,t_{\max}$; la convoluzione viene aggiornata con la seguente formula ricorsiva:

$$y_k(n) = y_{k-1}(n-1) + u(-k)h(n) \tag{2.2.9}$$

dove $u(n)$, con $n = -(143+11),...,39$, è il buffer di eccitazione. Si noti che nel processo di ricerca, i campioni $u(n)$, con $n = 0,...,39$, non sono conosciuti, tuttavia sono necessari per ritardi di pitch inferiori a 40. Come abbiamo accennato in precedenza, per sopperire a questa mancanza e per semplificare la ricerca, il residuo di predizione LP (2.2.6) viene copiato nel buffer $u(n)$ in modo da rendere sempre valida per ogni T_0 la relazione (2.2.9).

Una volta che il ritardo intero ottimo è ottenuto, vengono analizzate le frazioni da $-3/6$ a $3/6$ con passo $1/6$ attorno al valore intero. La ricerca del pitch lag fratto viene svolta interpolando la correlazione normalizzata dell'equazione (2.2.8) e cercando il suo massimo. L'interpolazione viene fatta usando un filtro usando un filtro FIR che chiameremo b_{24}, basato su un seno cardinale $\sin(x)/x$ finestrato con una Hamming window e troncato a ± 23, con uno zero padding ai campioni ± 24 ($b_{24}(24) = 0$). Il filtro ha una frequenza di cut-off (-3 dB) a $3600Hz$ nel dominio sovra-campionato. I valori interpolati di $R(k)$ per le frazioni da $-3/6$ a $3/6$ vengono ottenute usando la formula di interpolazione:

$$R(k)_t = \sum_{i=0}^{3} R(k-i)\, b_{24}(t+i\cdot 6) + \sum_{i=0}^{3} R(k+1+i)\, b_{24}(6-t+i\cdot 6), \quad t = 0,\ldots,5, \tag{2.2.10}$$

dove $t = 0,...,5$ corrisponde alle frazioni $0, 1/6, 2/6, 3/6, -2/6, -1/6$, rispettivamente.

Una volta che il pitch lag fratto è trovato, viene calcolato il vettore del codebook adattativo $v(n)$ interpolando l'eccitazione passata $u(n)$ ad un dato valore di ritardo intero k e fase (parte fratta) t:

$$v(n) = \sum_{i=0}^{9} u(n-k-i)\, b_{60}(t+i\cdot 6) + \sum_{i=0}^{9} u(n-k+1+i)\, b_{60}(6-t+i\cdot 6), \quad \begin{matrix} n = 0,...,39 \\ t = 0,...,5 \end{matrix} \tag{2.2.11}$$

Il filtro di interpolazione b_{60} è basato su un seno cardinale $\sin(x)/x$ finestrato con una Hamming window troncata a ± 59 inoltre viene posto $b_{60}(\pm 60)$=0; il filtro avrà una frequenza di taglio a $3600Hz$ nel dominio sovra-campionato.

Il guadagno dell'eccitazione adattativa per il subframe in considerazione è trovata dall'equazione:

$$g_p = \frac{\sum_{n=0}^{39} x(n)\,y(n)}{\sum_{n=0}^{39} y(n)\,y(n)} \qquad (2.2.12)$$

il quale viene limitato per $0 \le g_p \le 1.2$; $y(n) = v(n) * h(n)$ è il vettore di eccitazione del codebook adattativo filtrato per il filtro formatore quantizzato trovato nell'analisi LPC (2.2.5), ha quindi il significato di essere il segnale sintetizzato ricostruito, $x(n)$, il target signal, ha lo stesso significato, come abbiamo visto però, viene ricostruito in modo diverso (equazioni (2.2.6) e (2.2.7)). Infine, per la stabilità del filtro di ricostruzione del pitch, il guadagno dell'eccitazione g_p viene limitato a $GP_{threshold} = 0.95$.

La figura (2.4) offre un riassunto schematico delle operazioni closed-loop per la ricerca del tempo e guadagno di pitch relativa a ciascun subframe.

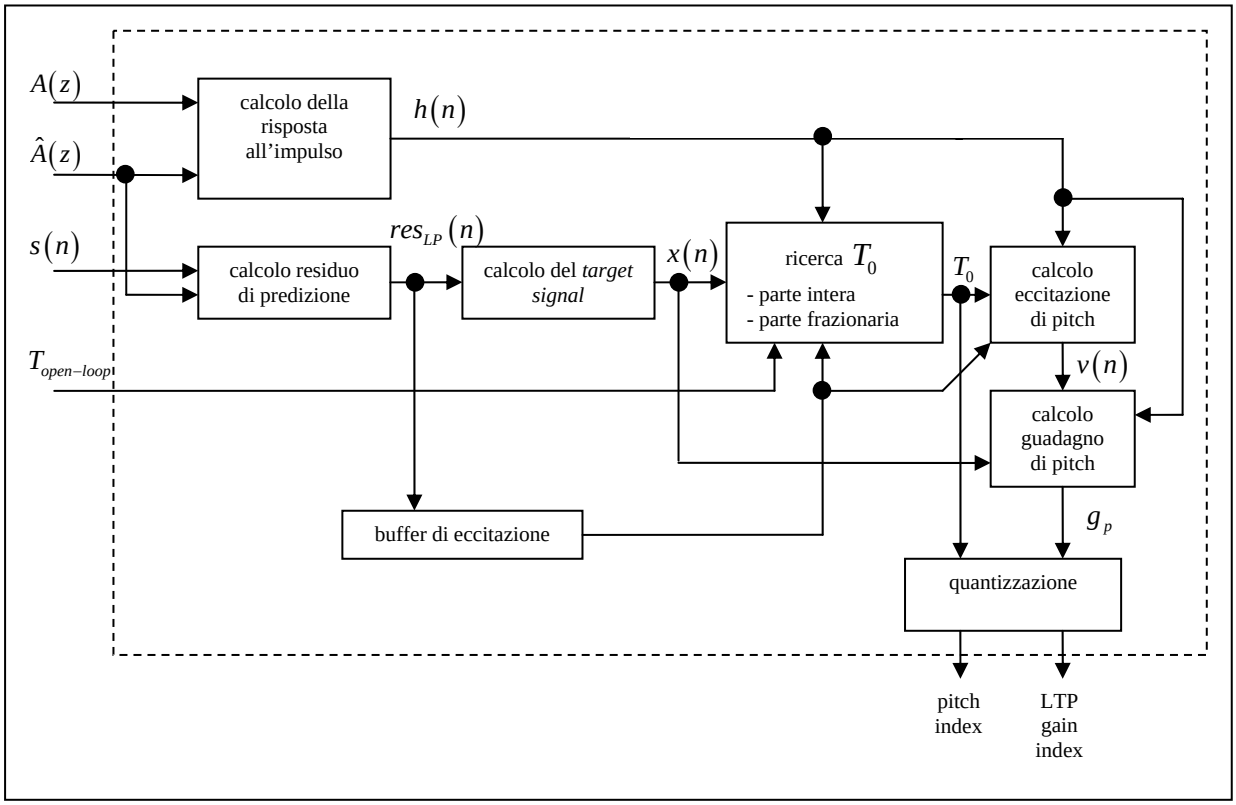


Figura 2.4 Ricerca closed-loop per l'eccitazione di pitch (per ogni subframe)

2.3 Analisi relativa all'eccitazione algebrica

Una volta tolta al segnale la correlazione di breve termine, con l'analisi LP vista nella prima sezione, e di lungo termine, con l'analisi di pitch vista nella seconda sezione, il residuo verrà ulteriormente trattato; infatti, infinite sono le realizzazioni del segnale gaussiano bianco in uscita dai due stadi di analisi precedenti. In questa sezione si analizzerà come l'encoder AMR fornisce, con pochi bit, informazioni importanti sull'ultimo residuo di predizione.
L'analisi dell'eccitazione algebrica viene anche chiamata *fixed codebook search*, cioè fisso, in contrasto con il codebook adattativo.

2.3.1 Struttura del codebook algebrico

Nel codebook utilizzato dall'encoder AMR, il vettore di innovazione per ciascun subframe, contiene solo 10 impulsi non nulli, i loro valori possono avere solo le due ampiezze ± 1. Le 40 posizioni nel subframe sono divise in 5 tracce, con ciascuna traccia contenente due impulsi, come mostrato nella tabella (2.1).

Traccia	**Impulso**	**Posizioni**
1	i_0, i_5	0, 5, 10, 15, 20, 25, 30, 35
2	i_1, i_6	1, 6, 11, 16, 21, 26, 31, 36
3	i_2, i_7	2, 7, 12, 17, 22, 27, 32, 37
4	i_3, i_8	3, 8, 13, 18, 23, 28, 33, 38
5	i_4, i_9	4, 9, 14, 19, 24, 29, 34, 39

Tabella 2.1 Possibili posizioni degli impulsi individuali nel codebook algebrico

Ciascuna posizione dei due impulsi viene codificata con 6 bit (per un totale di 30 bit, 3 bit a posizione); il segno del primo impulso nella traccia viene codificato con 1 bit, per un totale di 5 bit.
Per due impulsi localizzati nella stessa traccia è necessario solo un bit per il segno, infatti, come già detto, questo indicherà solo il segno del primo impulso. Il segno del secondo impulso dipende dalla posizione relativa al primo impulso: se questa è minore, allora è di segno opposto, altrimenti avrà lo stesso segno del primo impulso.

Tutti e tre i bit relativi alle posizioni vengono trattati con la codifica di Gray, per aumentarne la robustezza contro gli errori del canale. Questo da un totale di 35 bit per la sintesi dell'eccitazione algebrica.

2.3.2 Ricerca dell'eccitazione algebrica

La ricerca nel codebook algebrico avviene minimizzando l'errore quadratico medio tra il segnale in ingresso pesato e il segnale sintetizzato anch'esso pesato. Il target signal già visto nell'analisi di pitch closed-loop viene aggiornato sottraendo il contributo del codebook adattativo:

$$x_2(n) = x(n) - \hat{g}_p y(n), \quad n = 0,\ldots,39 \tag{2.3.1}$$

dove $y(n)$ è il vettore di eccitazione del codebook adattativo filtrato per il filtro formatore quantizzato trovato nell'analisi LPC (2.2.5), $y(n) = v(n) * h(n)$, e $\hat{g}_p$ è il guadagno del codebook adattativo quantizzato. Poniamo $\mathbf{c}_k$ come la parola di indice k nel codebook algebrico, allora la ricerca in quest'ultimo sarà effettuata massimizzando il termine:

$$A_k = \frac{(C_k)^2}{E_{Dk}} = \frac{\left(\mathbf{d}^t \mathbf{c}_k\right)^2}{\mathbf{c}_k^t \Phi \mathbf{c}_k} \tag{2.3.2}$$

dove $\mathbf{d} = \mathbf{H}^t \mathbf{x}_2$ è la correlazione tra il nuovo segnale target $x_2(n)$ e la risposta all'impulso $h(n)$, infatti $\mathbf{H}$ è una matrice con struttura di Toepliz triangolare inferiore con diagonale $h(0)$ e diagonali inferiori $h(1),...,h(39)$, e $\Phi = \mathbf{H}^t \mathbf{H}$ è la matrice di correlazione di $h(n)$. Il vettore $\mathbf{d}$ e la matrice Φ sono calcolate prima della ricerca nel codebook algebrico. Gli elementi del vettore $\mathbf{d}$ vengono calcolati così:

$$d(n) = \sum_{i=n}^{39} x_2(n) h(i-n), \quad n = 0,\ldots,39 \tag{2.3.3}$$

mentre gli elementi della matrice simmetrica Φ derivano dalla seguente relazione:

$$\phi(i,j) = \sum_{n=j}^{39} h(n-i) h(n-j), \quad (j \geq i) \tag{2.3.4}$$

La struttura del codebook algebrico, data la sua sparsità, permette una procedura di ricerca della parola ottima $\mathbf{c}_k$ molto veloce. La correlazione al numeratore dell'equazione (2.3.2) è data da:

$$C = \sum_{i=0}^{N_p - 1} \vartheta_i d(m_i) \tag{2.3.5}$$

dove m_i è la posizione dell'$i-$esimo impulso, ϑ_i è l'ampiezza, e $N_p = 10$ è il numero di impulsi. L'energia al denominatore invece è data da:

$$E_D = \sum_{i=0}^{N_p-1} \phi(m_i, m_i) + 2 \sum_{i=0}^{N_p-2} \sum_{j=i+1}^{N_p-1} \vartheta_i \vartheta_j \phi(m_i, m_j). \tag{2.3.6}$$

Per semplificare l'operazione di ricerca nel codebook, viene creato un segnale di cui viene preso solo il segno campione per campione, questo segnale, chiamato $b(n)$ e calcolato per ogni subframe, per la versione AMR 122 sarà:

$$b(n) = \frac{res_{LTP}(n)}{\sqrt{\sum_{i=0}^{39} res_{LTP}(i)\, res_{LTP}(i)}} + \frac{d(n)}{\sqrt{\sum_{i=0}^{39} d(i)\, d(i)}}, \quad n = 0, \ldots, 39, \tag{2.3.7}$$

dove $d(n)$ abbiamo già visto essere la correlazione tra il nuovo segnale target $x_2(n)$ e la risposta all'impulso $h(n)$ (equazione (2.3.3)), mentre $res_{LTP}(n)$ è il segnale residuo dopo aver tolto il contributo relativo alla sintesi della dinamica di timbro (correlazione di pitch). Una volta calcolato $b(n)$, per prima cosa si procede con l'operazione di estrazione del segno, $s_b(n) = sign(b(n))$ per ogni n; viene inoltre calcolato il segnale $d'(n) = d(n) s_b(n)$. Ora, la matrice Φ è modificata includendo le informazioni relative al segno; ovvero, $\phi'(i, j) = s_b(n) s_b(n) \phi(i, j)$. La correlazione nell'equazione (2.3.5) è ora data da:

$$C = \sum_{i=0}^{N_p-1} d'(m_i) \tag{2.3.8}$$

e l'energia nell'equazione (2.3.6) è data da:

$$E_D = \sum_{i=0}^{N_p-1} \phi'(m_i, m_i) + 2 \sum_{i=0}^{N_p-2} \sum_{j=i+1}^{N_p-1} \phi'(m_i, m_j). \tag{2.3.9}$$

Avendo prefissato le ampiezze degli impulsi di $b(n)$ come spiegato, le posizioni ottime degli impulsi vengono calcolate provando a massimizzare l'equazione (2.3.2) per una piccola percentuale di posizioni possibili.

Per prima cosa, per ciascuna delle cinque tracce, si cercano le posizioni degli impulsi con i massimi valori assoluti di $b(n)$. Da queste, viene trovato il massimo valore globale per tutte le posizioni degli impulsi. Il primo impulso i_0 è sempre posto nella posizione corrispondente al massimo valore globale.

Successivamente, vengono effettuate quattro iterazioni. Durante ciascuna di queste iterazioni, l'impulso i_1 viene messo nella posizione corrispondente al massimo locale di una traccia. Il resto degli impulsi viene cercato raggruppando le coppie $\{i_2, i_3\}$, $\{i_4, i_5\}$, $\{i_6, i_7\}$ e $\{i_8, i_9\}$;

ogni impulso ha otto possibili posizioni, quindi ci saranno quattro cicli con 8x8 combinazioni, per un totale di 256 combinazioni da provare per ciascuna iterazione.
In ciascuna iterazione, le posizioni dei nove impulsi da sistemare sono spostate ciclicamente, in modo che le coppie di campioni cambiano e i_1 viene posto nel massimo locale relativo ad un'altra traccia. Il resto degli impulsi sono cercati anche per le altre posizioni nelle tracce. Per finire almeno un impulso sarà posto dove $b(n)$ aveva il suo massimo globale ed un impulso sarà posto nella posizione corrispondente a uno dei 4 massimi locali.
Una caratteristica speciale incorporata nel codebook è quella nella quale la parola di codice viene filtrata attraverso un pre-filtro adattativo $F_E(z)$ che migliora particolari componenti spettrali così da aumentare la qualità dello *speech* sintetizzato. Il filtro in questione è:

$$F_E(z) = \frac{1}{\left(1-\beta z^{-T}\right)} \tag{2.3.10}$$

dove T è l'intero più vicino al pitch lag trovato nell'analisi closed-loop e β è il guadagno di pitch. Si noti che la risposta all'impulso $h(n)$ dovrà includere la presenza di questo filtro $F_E(z)$, questo risulterà nell'avere $h(n) = h(n) - \beta h(n-T)$ con $n = T,...,39$.
Il guadagno del codebook algebrico sarà dato da:

$$g_c = \frac{\mathbf{x}_2^t \mathbf{z}}{\mathbf{z}^t \mathbf{z}} \tag{2.3.11}$$

dove $\mathbf{x}_2$ è il *target vector* per il codebook algebrico (equazione (2.3.1)) e **z** è il vettore relativo alla parola di codice, convoluto con $h(n)$:

$$z(n) = \sum_{i=0}^{n} c(i)\, h(n-i), \qquad n = 0,\ldots,39. \tag{2.3.12}$$

La figura 2.5 riassume le operazioni svolte per la ricerca dell'eccitazione algebrica.

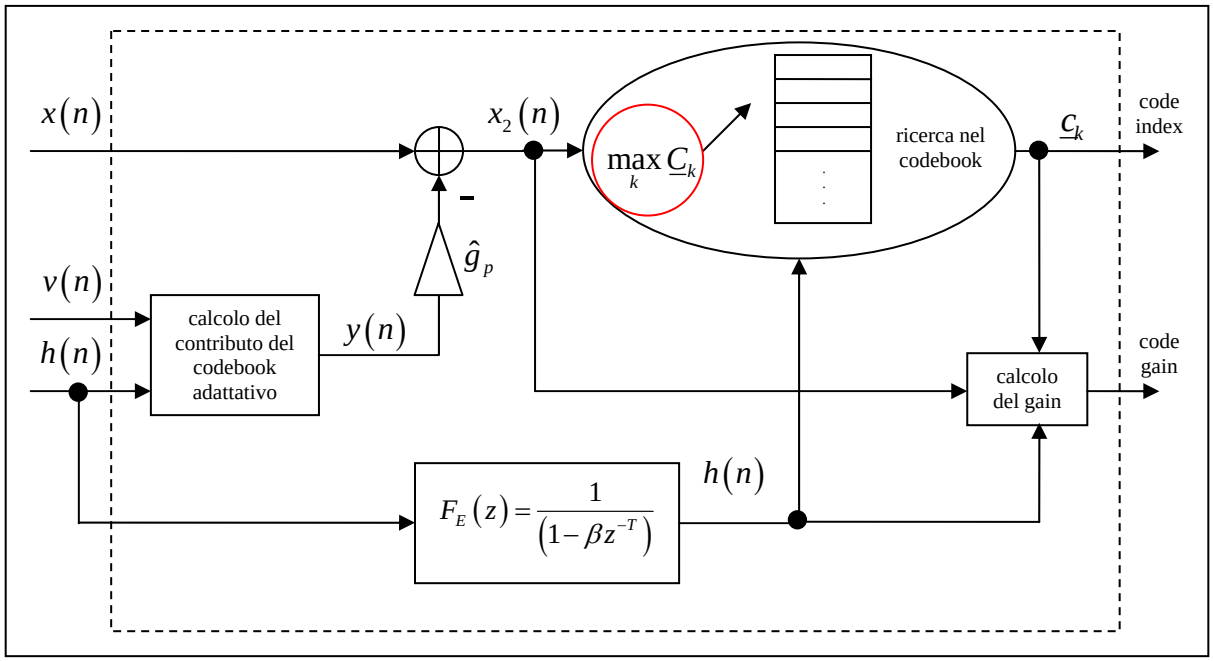


Figura 2.5 Operazioni relative alla ricerca dell'eccitazione algebrica

2.4 Allocazione dei bit nel payload AMR

Una volta effettuati tutti gli stadi di codifica, saranno pronti per essere spediti tutti i parametri estratti dal codificatore; per ogni trama avremo:

- due set di parametri LPC convertiti in LSP nel dominio coseno, quantizzati con SMQ (2.1.14)
- ritardo di pitch (*pitch index*) e relativo guadagno (*adaptive codebook gain*)
- parola di codice algebrico (si indicano le posizioni dei dieci impulsi e il segno di soli cinque) e relativo guadagno (*fixed codebook gain*).

La tabella 2.2 mostra dettagliatamente come ciascuno di questi parametri viene inserito nel payload AMR, il totale dei è di 244 bit, relativi a 20 ms di parlato, per un rate di 12.2 $kbit/s$.

Bits (MSB-LSB)	Descrizione
s1 - s7	Indice della prima sottomatrice LSF
s8 - s15	Indice della seconda sottomatrice LSF
s16 - s23	Indice della terza sottomatrice LSF
s24	Segno della terza sottomatrice LSF
s25 - s32	Indice della quarta sottomatrice LSF
s33 - s38	Indice della quinta sottomatrice LSF
subframe #1	
s39 - s47	adaptive codebook index
s48 - s51	adaptive codebook gain
s52	Informazioni sul segno per il primo e sesto impulso
s53 - s55	Posizione del primo impulso
s56	Informazioni sul segno per il secondo e settimo impulso
s57 - s59	Posizione del secondo impulse
s60	Informazioni sul segno per il terzo e ottavo impulso
s61 - s63	Posizione del terzo impulso
s64	Informazioni sul segno per il quarto e nono impulso
s65 - s67	Posizione del quarto impulso
s68	Informazioni sul segno per il quinto e decimo impulso
s69 - s71	Posizione del quinto impulso
s72 - s74	Posizione del sesto impulso
s75 - s77	Posizione del settimo impulso
s78 - s80	Posizione dell'ottavo impulso
s81 - s83	Posizione del nono impulso
s84 - s86	Posizione del decimo impulso
s87 - s91	fixed codebook gain
subframe #2	
s92 - s97	adaptive codebook index (differenza rispetto a quello del subframe precedente)
s98 - s141	Stessa descrizione dei bit s48 - s91
subframe #3	
s142 - s194	Stessa descrizione dei bit s39 - s91
subframe #4	
s195 - s244	Stessa descrizione dei bit s92 - s141

Tabella 2.2 Parametri in uscita dall'encoder AMR e loro allocazione nel payload AMR 122

2.5 Principali funzioni del decoder AMR

Il decoder si occuperà di "interpretare" il segnale in arrivo dal decodificatore di canale: andrà quindi a sintetizzare i parametri ricevuti per ricostruire il segnale vocale.

Il segnale vocale ricostruito sarà inoltre post-processato per migliorarne le proprietà psico-acustiche. Lo schema delle principali funzioni svolte dal decoder AMR è mostrato in figura 2.6.

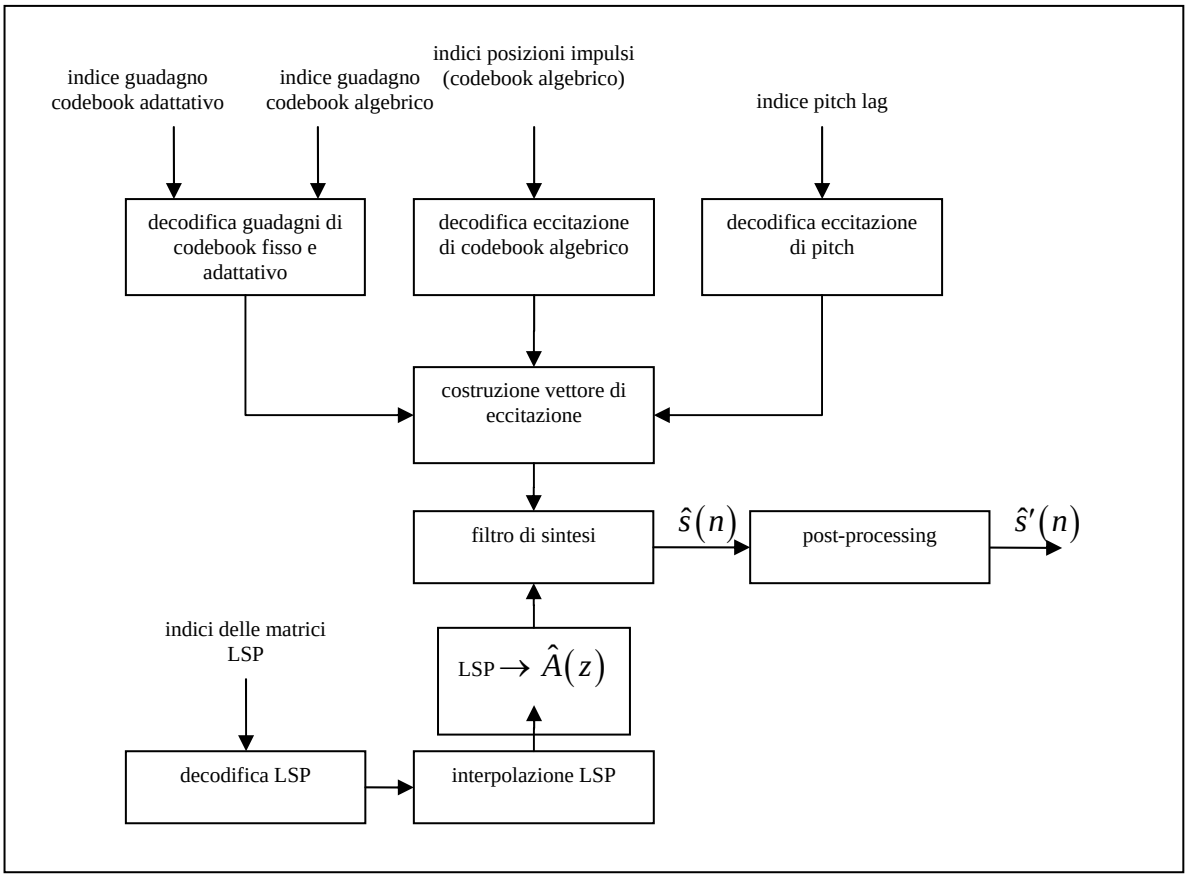


Figura 2.5 Schema a blocchi con le principali operazioni svolte dal decoder AMR

2.5.1 Decodifica dei parametri e sintesi del segnale vocale

Il processo di decodifica dei parametri per la successiva sintesi del segnale vocale viene svolto nel seguente ordine:

1. **Decodifica dei parametri del filtro LP: g**li indici relativi alla quantizzazione matriciale SMQ effettuata sugli LSP vengono usati per ricostruire quest'ultimi. L'interpolazione svolta è la medesima introdotta nella sezione 2.1.5; in uscita avremo quindi 4 vettori di LSP corrispondenti ai 4 subframe che il decoder ricostruirà. Per ciascun subframe, il corrispondente vettore sarà trasformato nei parametri LP a_k, usati per filtrare l'eccitazione ricostruita ai passi successivi.
2. **Ricostruzione dell'eccitazione del codebook adattativo:** l'indice relativo al pitch, viene usato per trovare la parte intera e la parte frazionaria del pitch lag. Il vettore di eccitazione $v(n)$ è trovato interpolando l'eccitazione passata $u(n)$ (al tempo di pitch trovato) con il filtro FIR descritto nell'equazione (2.2.11). La procedura quindi è la stessa mostrata nella sezione 2.2.2.

3. **Decodifica dell'eccitazione algebrica:** grazie agli indici presenti nel payload AMR contenenti le posizioni degli impulsi e il loro valore (che varrà sempre ±1, quindi l'informazione sarà solo sul segno), si ricostruirà il vettore $c(n)$, eccitazione del codebook algebrico. Esisterà inoltre un algoritmo anti-sparsità [16] per compensare ad eventuali errori sulle posizioni degli impulsi
4. **Decodifica del guadagno di codebook algebrico e del guadagno di codebook adattativo:** esisterà una algoritmo (che non presenteremo per brevità) per cercare congiuntamente i valori migliori per entrambi, questo per evitare innaturali fluttuazioni energetiche tra un subframe e l'altro.
5. **Decodifica del vettore d'innovazione.** L'ultimo passo sarà combinare le eccitazioni algebrica e adattativa:

   $$u(n) = \hat{g}_p v(n) + \hat{g}_c c(n) \tag{2.5.1}$$

   Dove $\hat{g}_p$ e $\hat{g}_c$ sono stati trovati al punto precedente e non sono i guadagni direttamente estratti dal pacchetto in ricezione.

   Si effettuerà inoltre una modifica di $u(n)$ per enfatizzare il contributo del codebook adattativo:

   $$\hat{u}(n) = u(n) + 0.25\beta\hat{g}_p v(n) \tag{2.5.2}$$

   se $\hat{g}_p > 0.5$ (valido solo per il modo 12.2 $Kbit/s$). Il parametro β è il guadagno di pitch effettivamente estratto dal payload AMR.

A questo punto, l'eccitazione sintetizzata viene filtrata da $\hat{A}(z)$ (corrispondente all'inverso del filtro IIR con coefficienti trovati nel primo passo dagli LSP):

$$\hat{s}(n) = \hat{u}(n) - \sum_{i=1}^{10} \hat{a}_i \cdot \hat{s}(n-i) \quad n = 0,...,39 \tag{2.5.3}$$

Il segnale vocale sintetizzato $\hat{s}(n)$ sarà quindi passato attraverso un filtro adattativo, descritto nella prossima sezione.

2.5.3 Post-elaborazione del segnale vocale sintetizzato

Il filtro adattativo che segue il processo di decodifica è la combinazione di due filtri: un filtro per enfatizzare le formanti, e un filtro per la cosiddetta *tilt compensation*, ovvero un "aggiustamento" della risposta in frequenza del filtro di enfatizzazione delle formanti.
Nel decoder i due filtri sono rispettivamente:

$$H_f(z) = \frac{\hat{A}(z/\gamma_n)}{\hat{A}(z/\gamma_d)} \tag{2.5.4}$$

e:

$$H_t(z) = 1 - \mu z^{-1} \tag{2.5.5}$$

dove $\hat{A}(z)$ è il filtro LP inverso, già utilizzato nell'equazione (2.5.3) per ricostruire il segnale vocale. I parametri γ_n e γ_d controllano l'influenza del filtraggio formantico e valgono rispettivamente 0.7 e 0.75. Mentre μ sarà il *tilt factor*, e dipenderà dal primo coefficiente di riflessione calcolato sulla risposta all'impulso, troncata ($L_h = 22$) , del filtro formantico $h_f(n)$.

Infine, il segnale viene filtrato passa-alto con una frequenza di taglio di $60\,Hz$. L'*up-scaling* verrà effettuato direttamente moltiplicando il filtro passa-alto per un fattore 2, questo compenserà il *down-scaling* effettuato dall'encoder in fase di pre-processing.

Capitolo 3

Analisi statistica dei parametri ACELP

In questo capitolo ci occuperemo dell'analisi delle proprietà e dello sviluppo di modelli statistici riguardanti i parametri in uscita dall'encoder AMR usati negli algoritmi presentati nei capitoli successivi. Uno studio statistico approfondito risulta particolarmente utile per apprezzare al meglio gli algoritmi di Voice Activity Detection e Acoustic Echo Cancellation presentati nel seguito del lavoro di tesi.

Si comincerà analizzando le linee spettrali di frequenza o LSF, i parametri più sensibili ma anche i più interessanti (e i più studiati) dal punto di vista delle proprietà e delle statistiche, essendo esse legate direttamente allo spettro del segnale.

Si passeranno in rassegna anche i parametri riguardanti la correlazione di lungo termine o *Long Term Prediction*. Direttamente legati al timbro e al tono di voce, questi parametri sono anche utilizzati in algoritmi di *speech recognition* [41]; nel lavoro di tesi, risulteranno particolarmente utili nell'implementazione dell'algoritmo di Voice Activity Detection.

L'ultimo parametro analizzato è il guadagno di codebook algebrico. Questo parametro risulterà particolarmente utile offrendo una diretta conoscenza del livello energetico del segnale. Nel seguito, potrà essere utilizzato, senza ulteriori manipolazioni sia per la discriminazione voce-rumore, sia per la cancellazione d'eco acustico.

I parametri analizzati in questo capitolo sono stati calcolati con gli stessi metodi dell'encoder AMR, questo principalmente sia per semplificare la trattazione, sia per rendere le considerazioni fatte direttamente applicabili agli stessi parametri generati dal codec.

3.1 Le linee spettrali di frequenza

3.1.1 Definizione e proprietà delle linee spettrali di frequenza

Nel primo e secondo capitolo abbiamo già visto le linee spettrali di frequenza essere le posizioni delle radici dei polinomi somma e differenza $P'(z)$ e $Q'(z)$:

$$\begin{aligned} P'(z) &= A(z) + z^{-11}A(z^{-1}) \\ Q'(z) &= A(z) - z^{-11}A(z^{-1}) \end{aligned} \tag{3.1.1}$$

legati ad $A(z) = 1 + \sum_{k=1}^{10} a_k z^{-k}$, polinomio ottenuto dall'analisi LP di ordine 10, dalla seguente relazione:

$$A(z) = \frac{P'(z) + Q'(z)}{2} \tag{3.1.2}$$

I due polinomi, hanno due radici fisse in $z = -1$ e $z = 1$ rispettivamente, quindi è possibile eliminarle, definendo i due nuovi polinomi:

$$\begin{aligned} P(z) &= \frac{P'(z)}{1 + z^{-1}} \\ Q(z) &= \frac{Q'(z)}{1 - z^{-1}} \end{aligned} \tag{3.1.3}$$

Si noti che si è scelto di svolgere l'analisi con un predittore di ordine 10 operante su intervalli temporali di $5\,ms$, come nell'encoder AMR.

Le radici dei polinomi nell'equazione (3.1.3) sono gli LSP, *Line Spectrum Pairs*. Essendo tutti gli zeri della forma $z = e^{j\phi_i}$, potranno essere espressi solo in funzione della loro fase, ovvero di ϕ_i; $f_i = 4000 \cdot \phi_i / \pi$ saranno gli LSF, *Line Spectrum Frequencies*, espressi in Hz. Rivediamo le principali proprietà degli LSF:

- tutti gli zeri dei due polinomi sono disposti sul cerchio unitario;
- gli zeri di $P(z)$ e $Q(z)$ sono inter-allacciati, ovvero gli zeri dell'uno e l'altro polinomio si alternano sul cerchio unitario;
- $P(z)$ e $Q(z)$ sono simmetrici;
- se $A(z)$ è a minima fase, questa proprietà sarà mantenuta anche nei due polinomi (che infatti hanno radici sul cerchio unitario).

La proprietà di minima fase del polinomio $A(z)$ viene mantenuta anche dopo la quantizzazione e la ricostruzione degli LSF, a patto che gli zeri di $P(z)$ e $Q(z)$ siano ancora inter-allacciati dopo la decodifica e si mantengano nell'ordine originale $0Hz < f_1 < f_2 < ... < f_{10} < 4000Hz$ [48], di questo si tiene conto anche nel codec AMR con degli speciali controlli per sopperire ad eventuali errori derivanti dal canale.

Se volessimo analizzare la funzione di trasferimento $H(z)$, nell'intervallo di frequenza fondamentale tramite l'analisi sul cerchio unitario ($z = e^{j\omega T}$), usando le equazioni (3.1.2) e (3.1.3), potremmo scrivere con confidenza:

$$H\left(e^{j\omega T}\right) = \frac{1}{A\left(e^{j\omega T}\right)} \approx \frac{2}{P\left(e^{j\omega T}\right) + Q\left(e^{j\omega T}\right)} \tag{3.1.4}$$

E' possibile dimostrare che il polinomio $A(z)$, avendo trovato i suoi coefficienti con il metodo dell'autocorrelazione (equazione (1.3.7)), avrà radici con una disposizione uniforme nel piano Z, se il processo osservato è bianco (caratterizzato da uno spettro piatto), oppure con radici vicine al cerchio unitario e una disposizione non uniforme se il processo è colorato (e quindi con una presenza di picchi nello spettro) [50]. Le linee spettrali di frequenza, per il modo in cui vengono calcolate avranno una relazione molto forte con le radici di $A(z)$. In particolare le radici di $P(z)$ e $Q(z)$, disposte sul cerchio unitario, tenderanno a disporsi nelle vicinanze delle radici di $A(z)$. Un esempio di questo è mostrato nelle figure 3.1 e 3.2.

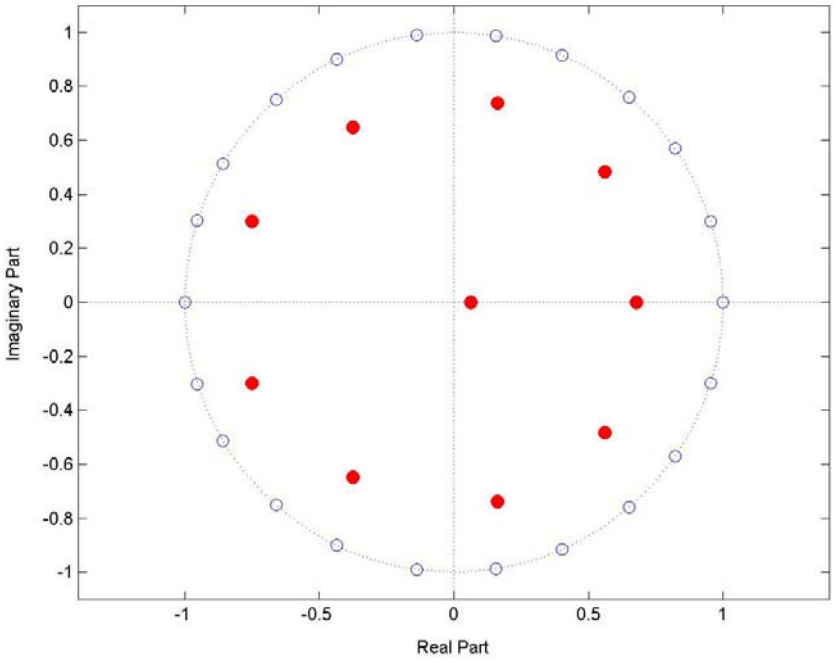


Figura 3.1 Disposizione delle radici del polinomio $A(z)$ (pallini rossi) e degli LSF (pallini bianchi) nel caso di rumore gaussiano bianco

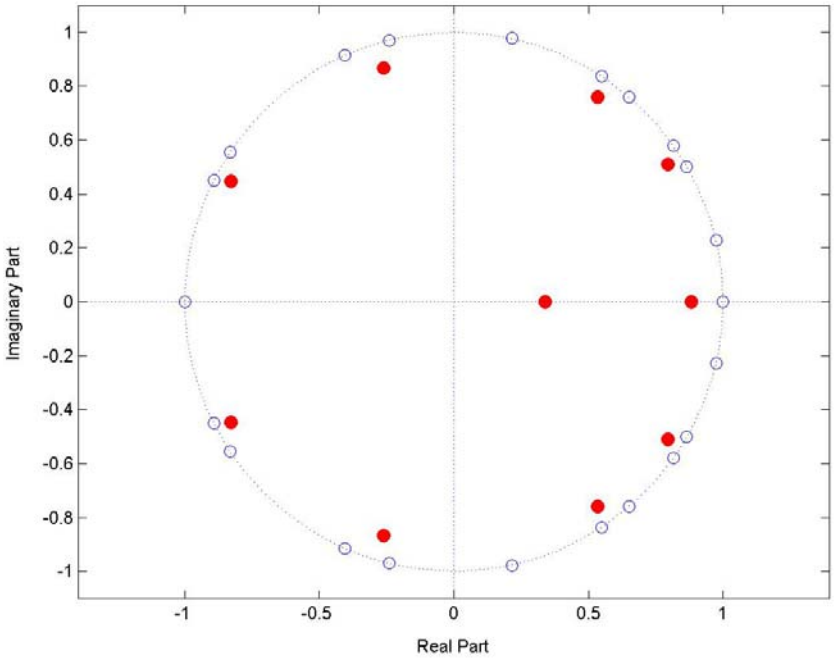


Figura 3.2 Disposizione delle radici del polinomio $A(z)$ (pallini rossi) e degli LSF (pallini bianchi) nel caso di suono vocalico

Quindi supponendo di trovarci in presenza di un processo con una forte caratterizzazione spettrale, valutando $\left[P\left(e^{j\omega T}\right)+Q\left(e^{j\omega T}\right)\right]$ in $\omega=\omega_0$ dove ω_0 è una frequenza vicina a due linee spettrali consecutive, essendo queste inter-allacciate e derivanti ciascuna rispettivamente dai polinomi $P(z)$ e $Q(z)$, $\left|P\left(e^{j\omega T}\right)+Q\left(e^{j\omega T}\right)\right|$ tenderà a un valore basso, mentre $\left|H\left(e^{j\omega T}\right)\right|$ tenderà ad avere un picco in ω_0. Un esempio ulteriore di questo è mostrato nelle figure 3.3 e 3.4 dove si mostrano gli andamenti delle tre funzioni di trasferimento $1/A(z)$, $1/P(z)$ e $1/Q(z)$. Si nota chiaramente che una concentrazione di due o tre linee spettrali di frequenza, o LSF *cluster*, caratterizza un massimo della funzione di trasferimento.

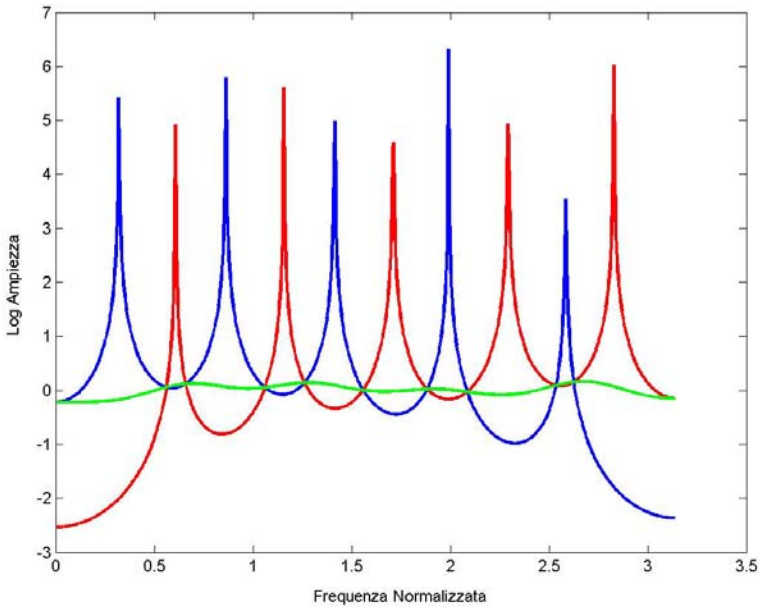


Figura 3.3 Andamento delle funzioni $\log\left(1/Q\left(e^{j\omega T}\right)\right)$ (blu), $\log\left(1/P\left(e^{j\omega T}\right)\right)$ (rosso) e $\log\left(1/A\left(e^{j\omega T}\right)\right)$ (verde) nel caso di rumore gaussiano bianco

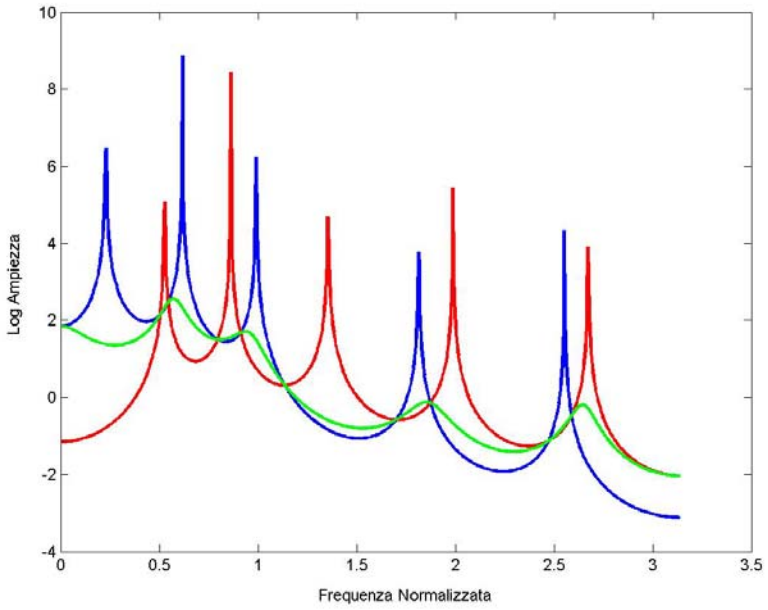


Figura 3.4 Andamento delle funzioni $\log\left(1/Q\left(e^{j\omega T}\right)\right)$ (blu), $\log\left(1/P\left(e^{j\omega T}\right)\right)$ (rosso) e $\log\left(1/A\left(e^{j\omega T}\right)\right)$ (verde) nel caso di suono vocalico

Questo ci consente di formulare due ipotesi:

- la posizione delle radici del polinomio $A(z)$ sarà determinante nella disposizione delle linee spettrali di frequenza e queste si disporranno nelle loro vicinanze;
- il comportamento spettrale di $H(z)$ potrà essere sintetizzato dalla semplice posizione delle linee spettrali di frequenza seguendo queste la disposizione della caratteristiche spettrali (nello specifico dei picchi) [56].

In particolare la seconda ipotesi risulterà particolarmente conveniente nell'implementazione degli algoritmi presentati nei capitoli seguenti, non sarà infatti necessario fare nessun tipo di calcolo per conoscere il comportamento spettrale del subframe analizzato (ad esempio la FFT), l'unica cosa che ci servirà sarà la conoscenza della disposizione delle linee spettrali di frequenza, condizione che potremo definire sufficiente all'identificazione del processo.

Per dimostrare la veridicità della seconda ipotesi è stato sviluppato un algoritmo estremamente semplice, in cui si ricrea la forma della funzione di trasferimento $H\left(e^{j\omega T}\right)$ con dei rettangoli. In particolare, da quanto detto finora, se alcune linee spettrali tendono a raggrupparsi attorno ad una frequenza ω_c, $H\left(e^{j\omega T}\right)$ tenderà ad avere un picco attorno a

questa frequenza, il metodo implementato dovrà mostrare questa caratteristica. Dato un vettore di LSF, $L = (0, l_1, l_2, ..., l_{10,} \pi)$, si costruisce un rettangolo ad ogni posizione degli LSF in questo modo:

$$H_i^L(\omega) = \begin{cases} \dfrac{A}{l_{i+1} - l_{i-1}} & l_{i+1} < f < l_{i-1} \\ 0 & altrimenti \end{cases} \tag{3.1.5}$$

Dove A è una costante senza importanza e visto che è uguale per tutti i rettangoli, può essere solo vista come un fattore di scala. Infine si sommano tutti questi rettangoli e si ottiene la funzione di trasferimento desiderata:

$$H^L(\omega) = \sum_{i=1}^{P} H_i^L(\omega) \tag{3.1.6}$$

In figura 3.5 è mostrato un esempio di questa implementazione, confrontata con la vera funzione di trasferimento.

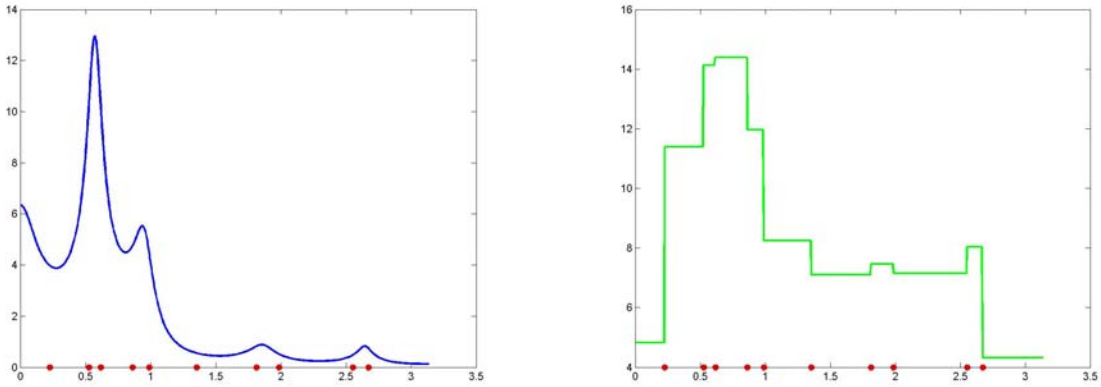


Figura 3.5 Andamento vero delle funzione $H(\omega)$ e la sua ricostruzione con il metodo dei rettangoli $H^L(\omega)$, i punti rossi identificano le posizioni degli LSF

Le considerazioni fatte finora sulle proprietà delle linee spettrali di frequenza, permettono di affermare con sicurezza che misure fatte sull'andamento delle linee spettrali di frequenza, saranno sufficienti a garantirne un'analisi spettrale completa per l'intervallo considerato.

3.1.2 Relazione tra formanti e posizione delle linee spettrali di frequenza

Nel primo capitolo abbiamo già introdotto il modello matematico del tratto vocale dove questo è stato considerato come una struttura riverberante, un "tubo" che collega le corde vocali alla bocca. Il segnale modellato dalla struttura riverberante presenterà dei picchi che saranno posti in corrispondenza di alcune frequenze di risonanza del tratto vocale, dove le

armoniche prodotte dalle corde vocali verranno rinforzate. Queste frequenze di risonanza vengono chiamate *formanti*. La posizione delle formanti caratterizza le vocali, nella tabella 1.1 sono stati mostrate le vocali usate nella lingua italiana con la loro posizione formantica. Nella figura 3.6 è mostrato lo spettrogramma della vocale /a/ dove le linee nere identificano la posizione delle formanti.

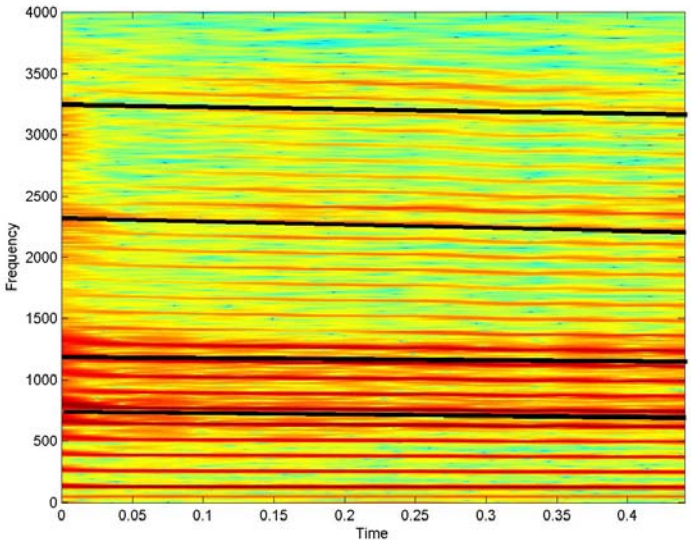


Figura 3.6 Spettrogramma della vocale /a/ e corrispondenti frequenze formantiche (linee nere)

In questa sezione siamo interessati nel trovare una misura che sia in grado di darci l'informazione formantica. Nello studio effettuato sul comportamento delle linee spettrali di frequenza risulta che, dato un vettore di LSF $L = (0, l_1, l_2, ..., l_{10,}\pi)$, il vettore composto dall'inverso della differenza di due LSF adiacenti risulterà essere uno strumento potente per l'identificazione di formanti, l' i – esima componente del vettore sarà data da:

$$IDLSF_i = \frac{1}{\Delta l_i} = \frac{1}{l_i - l_{i-1}}. \tag{3.1.7}$$

Infatti, conseguenza delle proprietà degli LSF, questo vettore, tenderà ad avere una deviazione standard elevata, nel caso di suoni vocalici con una forte caratterizzazione spettrale, mentre sarà praticamente nulla nel caso di una disposizione uniforme degli LSF sul cerchio unitario, corrispondente a uno spettro pressoché piatto; questa proprietà verrà ripresa nel capitolo successivo per la discriminazione tra voce e rumore. Nella figura 3.7 è mostrato l'andamento della funzione $IDLSF\left(\omega/T\right)$ dove nell'intervallo $l_i \le \omega/T \le l_{i-1}$ è stato posto il valore di $IDLSF_i$, la figura si riferisce al suono vocalico /a/ , si possono confrontare le formanti indicate nella figura 3.6 con quelle indicate nel grafico.

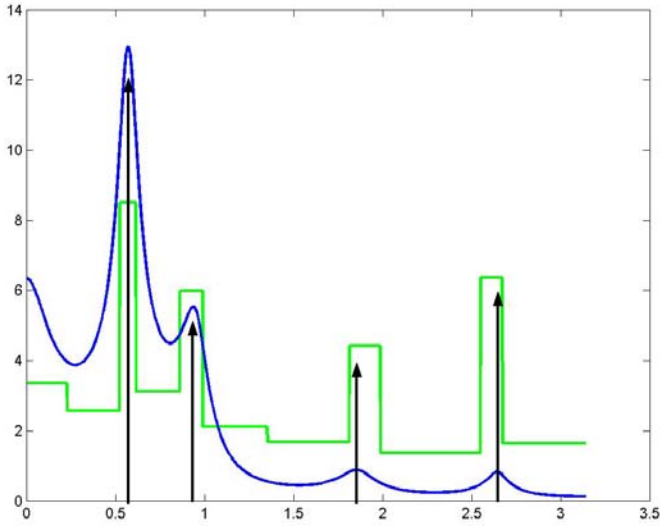


Figura 3.7 Confronto tra l'andamento in frequenza del suono vocalico /a/ (blu), funzione $IDLSF\left(\omega/T\right)$ per l'identificazione delle formanti (verde) e posizione delle formanti (frecce nere)

La posizione delle formanti abbiamo detto che caratterizza le vocali, tuttavia caratterizzerà anche il parlatore: basti pensare che numerosissimi fattori fisici possono far variare il modello matematico proposto per il tratto vocale (ad esempio, la lunghezza, il diametro, la forma, ecc…). Ciascuno di noi avrà un tratto vocale che risuonerà a frequenze diverse ed essendo le linee spettrali di frequenza uno strumento robusto per valutarne la posizione, queste potranno essere usate per campi di studio come lo *speech recognition*. Questo esula dalla nostra trattazione, comunque numerosi studi sono stati effettuati nell'uso delle linee spettrali di frequenza (con relativi studi statistici) applicate al riconoscimento del parlato, [30] [37] [38] [56] sono senz'altro i più apprezzati.

3.1.3 Valutazioni statistiche delle linee spettrali di frequenza

In questa sezione ci occuperemo dello studio statistico delle linee spettrali di frequenza. Innanzitutto, cominceremo osservando il comportamento delle linee spettrali di frequenza: queste tenderanno a rimanere più o meno equidistanti, sul cerchio unitario, nel caso di rumore a spettro piatto, mentre tenderanno a mostrare dei raggruppamenti netti nel caso di segnale vocalico. Un esempio del loro comportamento è mostrato in figura 3.8.

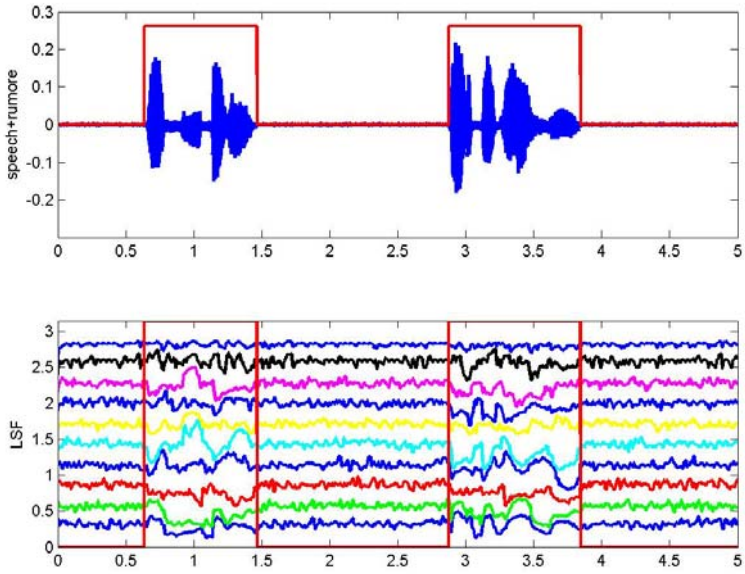


Figura 3.8 Andamento delle linee spettrali di frequenza calcolate in intervalli temporali di 5 ms sul segnale mostrato nella parte superiore ($SNR = 35dB$)

Non risulta particolarmente facile definire una densità di probabilità delle linee spettrali di frequenza in forma chiusa in quanto queste si muoveranno in intervalli non definibili a priori e fortemente correlati con il tipo di segnale analizzato. L'unico caso in cui è possibile definire con confidenza la densità di probabilità delle linee spettrali di frequenza è quella in cui il segnale osservato è gaussiano bianco ed è stazionario, infatti, per il teorema di Mann e Wald i parametri della stima AR saranno gaussiani [35], esisterà poi una procedura iterativa per il calcolo della d.d.p. degli LSF (introdotta da Tourneret e Ghogho [50]) dove si arriverà a dimostrare che ciascun LSF presenta una densità di probabilità gaussiana con valore medio per l'$i-$esima $(i = 1,..., p)$ linea spettrale pari a $i \cdot \left(\pi / \left(p+1 \right) \right)$, dove p è l'ordine di predizione e anche il numero di LSF. Questo ha un significato latente importante: gli LSF provenienti dall'osservazione analitica di un segnale gaussiano, in media, saranno perfettamente equispaziati, concordemente con la disposizione sul cerchio unitario per segnali a spettro piatto. La varianza invece cambierà a seconda della linea spettrale considerata, sarà comunque facilmente calcolabile. Un esempio di questo è mostrato in figura 3.9. L'analisi è stata effettuata su una realizzazione di un processo gaussiano bianco di 20 secondi, gli LSF vengono calcolati ogni 5 *ms* (4000 intervalli temporali), la figura mostra l'istogramma trovato, per ogni LSF, in rosso, è mostrata la d.d.p. gaussiana con media e varianza calcolata su ciascun LSF.

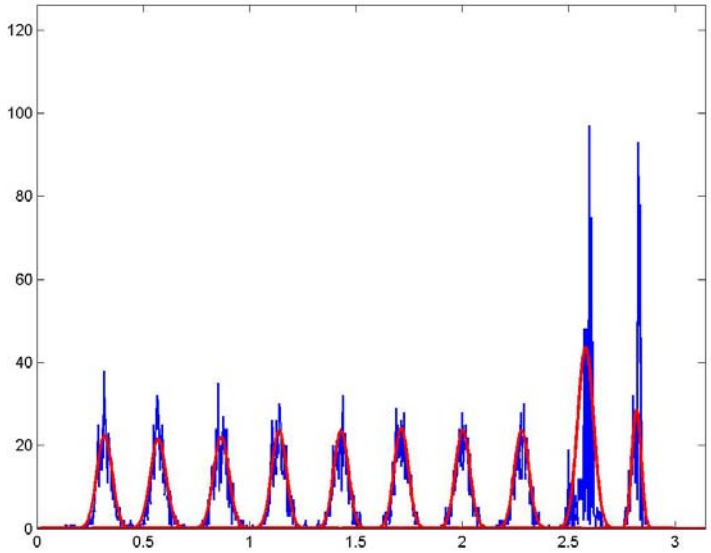


Figura 3.9 Istogramma dell'andamento delle linee spettrali di frequenza (blu) e loro comportamento analitico gaussiano (rosso) per un segnale gaussiano bianco

In questa prima parte siamo riusciti a dimostrare quello che intuitivamente potevamo già inferire dalle proprietà degli LSF. Per un segnale decisamente non stazionario come il parlato umano, invece, sarà praticamente impossibile definire una densità di probabilità in forma chiusa, in quanto gli LSF si muovono molto più rapidamente e in maniera meno prevedibile che con un rumore gaussiano in ingresso. In particolare, a seconda del parlatore e del fonema prodotto le statistiche cambieranno radicalmente. Inoltre, esisterà anche una correlazione piuttosto marcata tra la posizione di ciascun LSF rispetto a tutti gli altri. Tuttavia, effettuando un'analisi sperimentale *speaker-dependent*, ovvero su un singolo parlatore e quindi da egli dipendente, è stato possibile definire per un insieme considerevole di fonemi prodotti una matrice di auto-correlazione normalizzata [28]:

$$\Omega(i,k) = \left| \frac{E\left[\omega_i \omega_k\right] - E\left[\omega_i\right] \cdot E\left[\omega_j\right]}{\sqrt{E\left[\omega_i^2\right] \cdot E\left[\omega_k^2\right]}} \right| \quad i = 1,...,10, \quad j = 1,...,10 \tag{3.1.8}$$

dove ω_i è la posizione dell'$i-$esimo LSF. La tabella 3.1 mostra i risultati ottenuti per il fonema /a/, sono state prese in considerazione 10 realizzazioni diverse provenienti dal medesimo parlatore ciascuna di durata 500 ms, al solito campionate a 8000 Hz e con analisi predittiva di ordine 10 effettuata ogni 5 ms (40 campioni), per un totale di 1000 vettori LSF. I risultati ottenuti sono particolarmente significativi, infatti si è scelta questa particolare normalizzazione per mostrare come alcuni LSF siano decisamente più correlati, a seconda del

fonema, con altri più lontani, che non con quelli più vicini, diretta conseguenza della correlazione tra la posizione di ciascuna formante e quella delle altre.

i	*j*									
	1	2	3	4	5	6	7	8	9	10
1	1.00	0.65	0.30	0.35	0.33	0.33	0.39	0.40	0.36	0.20
2	0.65	1.00	0.77	0.11	0.07	0.13	0.07	0.05	0.06	0.07
3	0.30	0.77	1.00	0.44	0.25	0.24	0.46	0.54	0.39	0.28
4	0.35	0.11	0.44	1.00	0.72	0.31	0.46	0.42	0.45	0.21
5	0.33	0.07	0.25	0.72	1.00	0.44	0.52	0.47	0.34	0.26
6	0.33	0.13	0.24	0.31	0.44	1.00	0.71	0.61	0.49	0.28
7	0.39	0.07	0.46	0.46	0.52	0.71	1.00	0.73	0.58	0.41
8	0.40	0.05	0.54	0.42	0.47	0.61	0.73	1.00	0.58	0.46
9	0.36	0.06	0.39	0.45	0.34	0.49	0.58	0.58	1.00	0.77
10	0.20	0.07	0.28	0.21	0.26	0.28	0.41	0.46	0.77	1.00

Tabella 3.1 Matrice degli indici di correlazioni tra l'$i-$esimo e il $j-$esimo LSF

A questo punto, definiti gli indici di correlazione per ciascuna coppia di LSF, avendo questi un andamento medio piuttosto marcato con una deviazione piuttosto bassa e quindi con una d.d.p. convessa, potremmo stabilire, sempre in maniera sperimentale, che è possibile definire la densità di probabilità. Questa, per il teorema del limite centrale, sarà gaussiana, a patto di osservare numerose realizzazioni ciascuna definita su un intervallo di tempo piuttosto breve (circa 300-600 *ms* per una vocale). In figura 3.10 è mostrato il risultato sugli stessi vettori di LSF usati per il calcolo della matrice delle correlazioni.

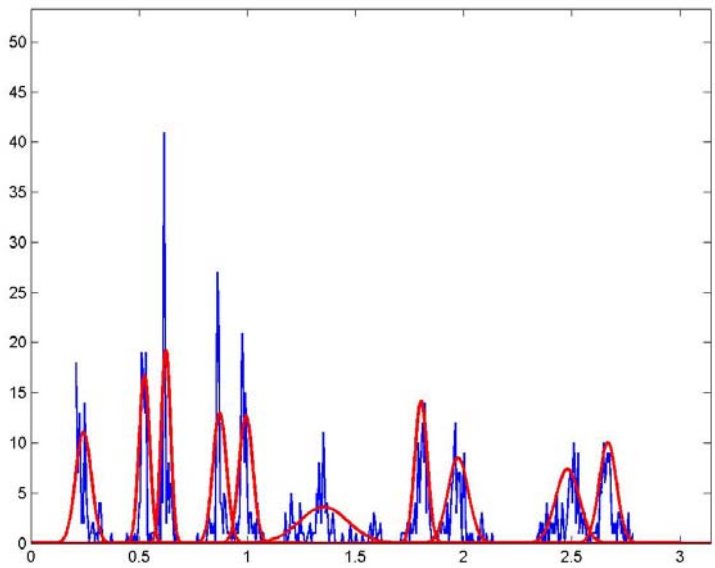


Figura 3.10 Istogramma dell'andamento delle linee spettrali di frequenza (blu) e loro comportamento analitico gaussiano (rosso) per la vocale /a/ osservato su 1000 vettori di LSF

Si noti come la varianza di tutti i vettori si sia ridotta drasticamente (tranne il quinto), in media non saranno più equidistanti ma dipenderanno dalla formante che andranno a rappresentare.
Sempre usando il teorema del limite centrale si può affermare che, osservando per un lungo periodo di tempo l'andamento degli LSF in presenza di una normale conversazione telefonica, (in cui, in media, si parla solo per il 50% del tempo), visto che la posizione degli LSF cambierà radicalmente a seconda della formante, l'andamento totale sarà gaussiano, molto simile a quello osservato in figura 3.9, riguardante gli LSF del rumore gaussiano bianco. Considerazione comunque poco utile ai fini pratici dello sviluppo di algoritmi per la modifica delle posizioni degli LSF, per, ad esempio, Noise Reduction o Acoustic Echo Cancellation. Questa generalizzazione, inoltre, non varrà se il tempo di osservazione non sarà dell'ordine dei minuti: l'ipotesi di gaussianità non reggerà, le densità di probabilità inoltre saranno multimodali e quindi con la presenza di massimi locali. Un esempio di questo è mostrato per un tempo di analisi di 10 secondi per i primi 4 LSF, i più coinvolti nel movimento delle formanti.

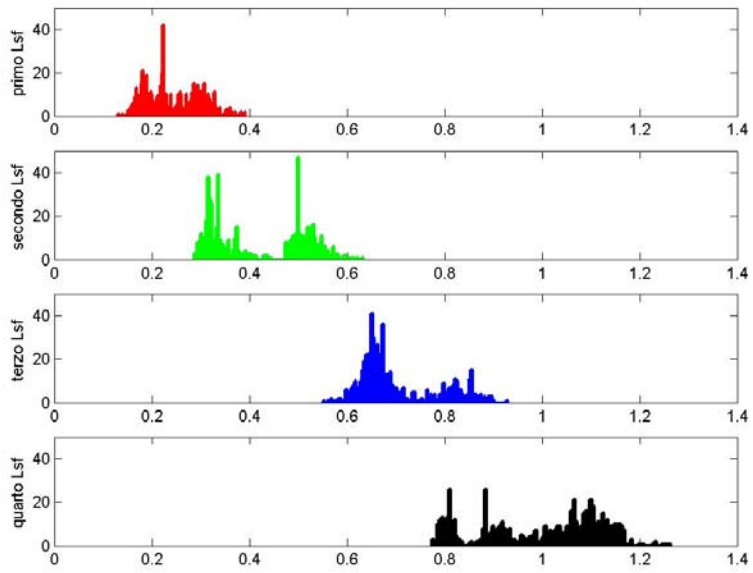


Figura 3.11 Istogramma dell'andamento delle prime quattro linee spettrali di frequenza in un segmento di parlato di 10 secondi

L'importanza di essere riusciti a dimostrare, anche se in maniera sperimentale, che la densità di probabilità di ciascun LSF per un dato fonema è gaussiana, risiede nella possibilità di definire il comportamento degli LSF solamente con i dati di media e varianza di ciascuno di

essi. Questo comporta, ad esempio, che in un determinato numero di subframe, rappresentanti di un fonema, se il segnale sarà corrotto da rumore, sarà possibile sostituire agli LSF il loro comportamento medio *noto* in presenza di quello stesso fonema, sicuri che questa scelta sarà la migliore possibile (d.d.p. convessa con un solo massimo). Queste considerazioni saranno fondamentali soprattutto nel caso di algoritmi di *noise reduction* in cui si andranno a modificare gli LSF per "ripulire" il segnale utile da componenti spurie. Ovviamente, andare a toccare gli LSF, significa modificare i coefficienti del polinomio $A(z)$, quindi l'errore di predizione potrà solo peggiorare, contraddicendo le condizioni di ortogonalità (equazione 1.3.4) tra dati ed errore; questa strada è risultata comunque praticabile.

3.1.4 Correlazione *inter-frame* delle linee spettrali di frequenza

Nell'ultima parte dell'analisi riguardante le linee spettrali di frequenza, ci occuperemo della correlazione *inter-frame*, ovvero, effettueremo un'analisi della relazione che intercorre tra LSF nella stessa posizione ma calcolati su intervalli temporali diversi. L'importanza di questo studio risiede nella possibilità di *prevedere* il comportamento futuro di ciascun LSF per due scopi, uno più pratico per la riduzione del rate di bit di informazione (ad esempio, usando codifiche differenziali) e uno più speculativo per comprendere quanto influirà il vettore di LSF attuale sui futuri. In realtà abbiamo già introdotto l'argomento parlando nel secondo capitolo del codec AMR, questo infatti codifica solo metà dei vettori LSF, gli altri li trova calcolando la media tra quelli adiacenti, sfruttando la forte correlazione che esiste tra interpolante e interpolando. Un indice che si è ritenuto significativo per questo tipo di misura è [28]:

$$\Phi(i,k)=\left|\frac{E\left[\omega_{n,i}\omega_{n-k,i}\right]-E\left[\omega_{n,i}\right]\cdot E\left[\omega_{n-k,i}\right]}{\sqrt{E\left[\omega_{n,i}^2\right]\cdot E\left[\omega_{n-k,i}^2\right]}}\right| \quad i=1,2,...,10, \quad k=1,2,...,10 \tag{3.1.9}$$

dove k indica la distanza tra il vettore di LSF considerato e quello rispetto al quale si calcola la correlazione. L'indice i rappresenta la posizione della linea spettrale di frequenza all'interno del vettore. I risultati sono mostrati nella tabella 3.2. Per il calcolo sono stati presi 4000 vettori di LSF calcolati su intervalli temporali di 5 *ms* (totale $20\,s$ di conversazione), tutti provenienti dallo stesso parlatore in una conversazione tipo. L'ordine di predizione usato è $p=10$, uguale alla cardinalità del vettore di LSF.

i	*k*									
	1	2	3	4	5	6	7	8	9	10
1	0.93	0.84	0.76	0.68	0.61	0.55	0.50	0.45	0.41	0.36
2	0.89	0.75	0.63	0.54	0.46	0.38	0.32	0.27	0.22	0.18
3	0.92	0.80	0.70	0.60	0.51	0.43	0.36	0.30	0.24	0.20
4	0.92	0.82	0.73	0.64	0.56	0.49	0.43	0.37	0.32	0.27
5	0.95	0.88	0.81	0.74	0.67	0.61	0.54	0.48	0.43	0.37
6	0.94	0.85	0.77	0.69	0.62	0.56	0.49	0.44	0.38	0.33
7	0.93	0.83	0.75	0.66	0.58	0.50	0.43	0.37	0.31	0.26
8	0.91	0.81	0.72	0.64	0.56	0.49	0.43	0.37	0.32	0.28
9	0.87	0.73	0.64	0.55	0.48	0.42	0.37	0.33	0.29	0.25
10	0.82	0.66	0.57	0.50	0.44	0.38	0.34	0.30	0.27	0.24

Tabella 3.2 Matrice degli indici di correlazioni $\phi(i,k)$ tra l'$i-$esimo LSF del vettore $n-$esimo e l'$i-$esimo LSF del vettore $(n-k)-$esimo (il valore massimo possibile sarà 1, relativo alla totale identità statistica dei processi osservati)

I risultati ottenuti sono estremamente significativi, innanzitutto non esiste solo una forte correlazione tra vettori direttamente adiacenti ma anche tra vettori calcolati fino a 5-6 intervalli successivi.

Le considerazioni fatte e i risultati trovati sulla correlazione inter-frame saranno utili soprattutto nella parte del lavoro di tesi riguardante la discriminazione voce-rumore in cui sarà importante sviluppare una certa "lungimiranza" sul comportamento futuro del segnale analizzato (soprattutto per il parlato) per non intercorrere in errori di valutazione, come sarà chiaro in seguito.

3.2 Statistiche relative ai parametri di *Long Term Prediction*

In questo paragrafo affronteremo le statistiche relative tempo di pitch e il guadagno di pitch, i parametri in uscita dal blocco di *Long Term Prediction* per l'individuazione e lo sbiancamento di correlazioni di lungo termine. In particolare, ne affronteremo le proprietà nei casi di suoni vocalici (*voiced*) ben definiti e nei casi di rumore o di suoni sordi (*unvoiced*), assimilabili a rumore colorato.

Si è utilizzata l'analisi LTP a un solo polo, ancora una volta per applicare le considerazioni fatte di seguito direttamente sui due parametri in uscita dall'encoder AMR.

Nel grafico di figura 3.12 è mostrato l'andamento dei due parametri analizzati in funzione del tempo, i fonemi analizzati sono le vocali dell'alfabeto italiano (/a/, /e/, /i/, /o/, /u/).

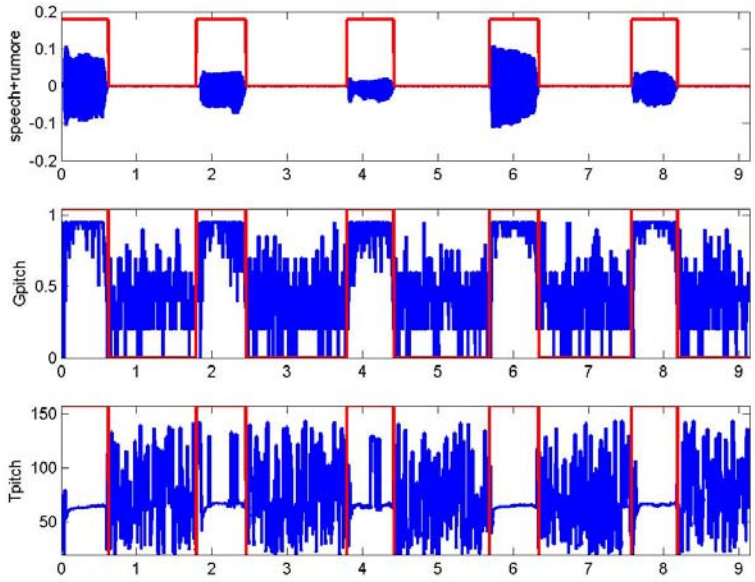


Figura 3.12 Andamento del guadagno e del tempo di pitch nelle vocali italiane (/a/, /e/, /i/, /o/, /u/), al segnale pulito è stato sovrapposto del rumore AWGN. SNR=25 dB

3.2.1 Il tempo di pitch

Nel primo capitolo si è già data un'interpretazione fisica del tempo di pitch come inverso della frequenza fondamentale, ovvero della periodicità del fenomeno di occlusione-divaricazione delle corde vocali, il quale determina suoni di tipo *voiced* o vocalici.

La curiosità riguardante il tempo di pitch, è la possibilità di essere usato sia come discriminante per algoritmi di Voice Activity Detection, sia per identificare segmenti di parlato di persone diverse [31]. Il tempo di pitch, infatti, è direttamente associato al timbro della voce, quindi può differire notevolmente da persona a persona; basti pensare alla voce di un bambina e alla voce di un uomo adulto particolarmente roca, le frequenze fondamentali del loro parlato si aggirano, in media, attorno ai $350\,Hz$ e ai $70\,Hz$, rispettivamente. Questo ovviamente è un caso estremo, tuttavia differenze si possono trovare anche tra persone con voci molto simili. Un'altra considerazione importante riguarda l'andamento del T_{pitch}, questo infatti non rimarrà costante durante il suono vocalico, ma tenderà ad avere un andamento diverso a seconda del tono che l'individuo imprime alla voce, sarà infatti decrescente se si esprime un'interrogazione o crescente in caso dichiarativo.

In questo breve paragrafo si darà un'analisi del tempo di pitch, cercando di definirne una d.d.p. in maniera empirica. L'analisi è stata effettuata sui parametri in uscita dall'encoder AMR, i valori del tempo di pitch quindi saranno indicati in campioni:

$$T_p^{AMR} = \frac{f_s}{f_0} \tag{3.2.1}$$

dove f_s è la frequenza di campionamento (8000 Hz) e f_0 è la frequenza fondamentale, il range di valori possibili sarà tra 18 e 143 campioni, corrispondenti a un range di frequenza per f_0 pari a $56Hz \div 457Hz$.

Analizzando numerosi segmenti di parlato, siamo arrivati a definire una densità di probabilità per il tempo di pitch gaussiana, con media e varianza dipendenti dal parlatore. In figura 3.12 è mostrato un esempio della distribuzione del tempo di pitch dei suoni vocalici italiani (/a/, /e/, /i/, /o/, /u/) per due parlatori della stessa età, dello stesso sesso e della stessa lingua. Si noti la riprova di quanto affermato in precedenza, ovvero che il tempo di pitch può essere discriminante anche per persone con voce molto simile.

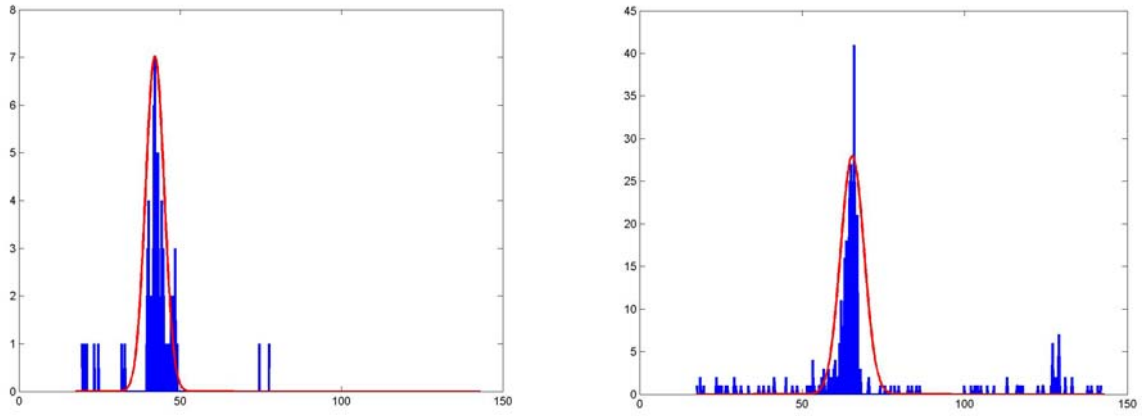


Figura 3.12 Istogrammi della distribuzione del tempo di pitch dei suoni vocalici italiani (/a/, /e/, /i/, /o/, /u/) per due parlatori uomini della stessa età (blu) e andamento analitico gaussiano (rosso)

Non è possibile invece, come risulta chiaro dalla figura 3.11, definire un andamento del pitch in condizioni sia di rumore che di suoni sordi (consonanti) poiché in questi casi le corde vocali in primis non emettono un segnale periodico e quindi non esisterà una correlazione di lungo termine da trovare. In figura 3.13 è mostrato un esempio, l'unico comportamento statistico che possa avere significato attribuire a questa distribuzione è quello dell'uniformità nel range di valori possibili.

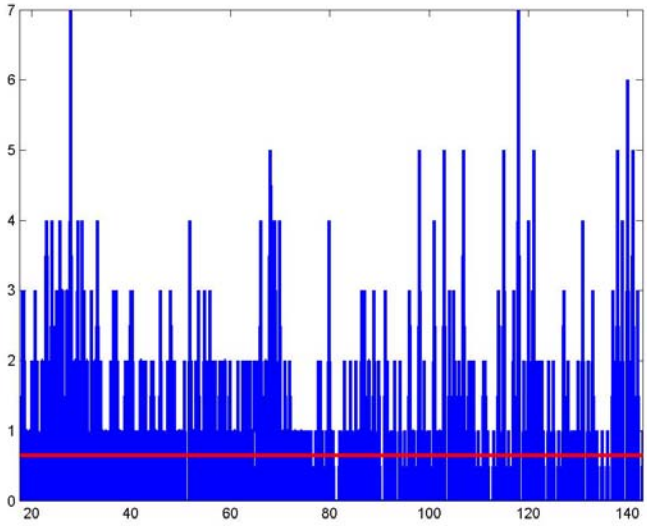


Figura 3.13 Istogramma della distribuzione del tempo di pitch per rumore bianco, colorato e suoni sordi (consonanti). In rosso è mostrata la d.d.p. uniforme assegnatagli .

Concludendo, il tempo di pitch è un parametro che vedremo nel seguito essere estremamente importante per la discriminazione tra voce e rumore: in presenza di un suono vocalico questo si manterrà praticamente costante identificando univocamente segmenti temporali di suono *voiced*.

3.2.2 Il guadagno di pitch

Il guadagno di pitch risulta seguire molto da vicino l'andamento del tempo di pitch. Infatti, quando si presenterà una correlazione di lungo termine, ovvero quando sarà possibile identificare un andamento quasi costante della frequenza fondamentale, il guadagno di pitch tenderà ad essere vicino al suo valore massimo consentito dal codec AMR, ovvero 0.95 (riguardante la stabilità del filtro IIR LTP (equazione (1.4.7)); questo sarà dovuto alla maggiore energia racchiusa nell'informazione LTP dei suoni vocalici.
Dall'osservazione sul suo andamento, si è notata la convessità degli istogrammi relativi alle parti di parlato e di rumore con una differenza importante dell'andamento medio tra i due. In figura 3.14 sono mostrati i due istogrammi.

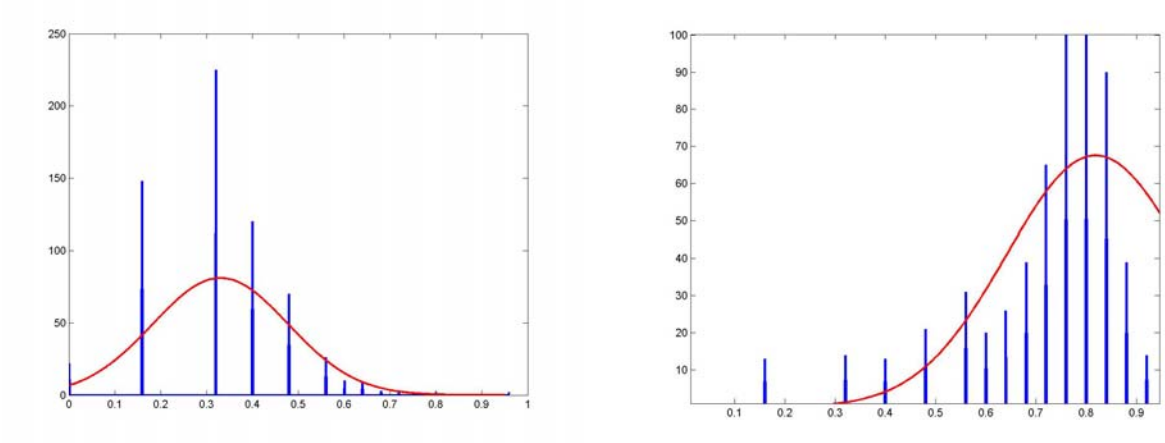


Figura 3.13 Istogramma della distribuzione del guadagno di pitch per rumore gaussiano bianco (sinistra) e suoni vocalici con forte caratterizzazione di pitch (destra). In entrambi è mostrata in rosso la d.d.p. convessa simil-gaussiana assegnata loro.

Il guadagno di pitch è quindi utilizzabile per una discriminazione tra suoni vocalici e sordi o rumorosi, sia in abbinamento all'informazione di pitch, sia da solo visto che la sua distribuzione permette discriminazioni basate sulla distanza.

3.3 Statistiche relative al guadagno di codebook algebrico

L'ultimo parametro che andremo ad analizzare, sarà il guadagno di codebook algebrico. Questo parametro offre importanti qualità in quanto risulta strettamente legato alla quantità di energia presente nell'intervallo temporale analizzato. Effettuando uno studio sui parametri AMR infatti potremmo riscrivere la funzione di trasferimento relativa al subframe $m-$esimo, relativa a $5\,ms$ di segnale analizzato, ignorando l'eccitazione di codebook algebrico, come:

$$H(z,m)=\frac{g_{codebook}(m)}{\left(1-g_p(m)\cdot z^{-T(m)}\right)\left(1+\sum_{i=1}^{10}a_i(m)\cdot z^{-i}\right)} \tag{3.3.1}$$

definendo così $g_{codebook}(m)$ come una semplice costante moltiplicativa della parte di modellazione del segnale gaussiano bianco in ingresso (l'eccitazione di codebook algebrico). Nella figura 3.14 viene mostrato l'andamento del guadagno di codebook in presenza di parlato.

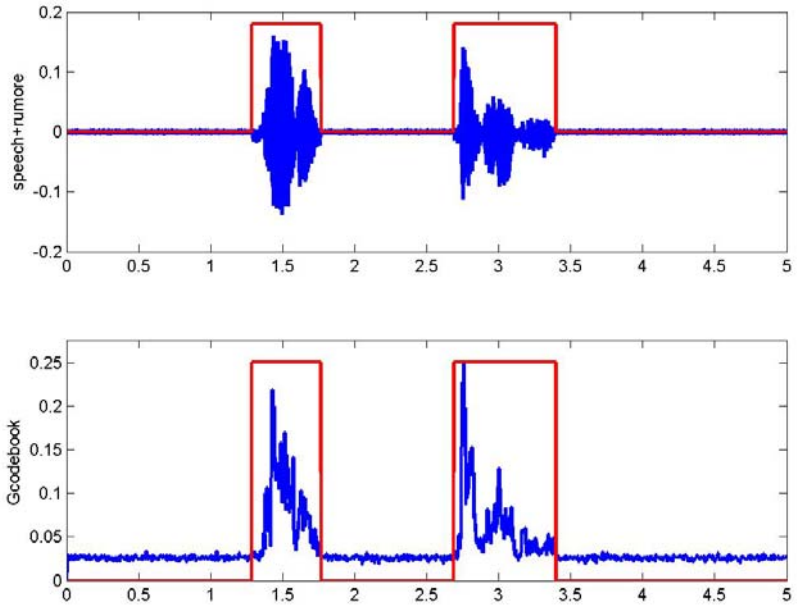


Figura 3.14 Andamento del guadagno di codebook algebrico, calcolato ogni intervallo temporale di $5\,ms$ sul segnale mostrato nella parte superiore ($SNR = 35dB$)

Per il guadagno di codebook algebrico dei subframe di solo rumore è semplice definire un andamento statistico di tipo gaussiano; infatti, se il rumore sarà gaussiano bianco (stazionario), l'andamento energetico del segnale sarà anch'esso gaussiano e di conseguenza anche $g_{codebook}(m)$. La d.d.p. relativa a $g_{codebook}(m)$ in condizioni di parlato, sarà invece difficile da calcolare, tuttavia, si può stabilire empiricamente che in condizioni normali (con livello energetico della voce impressa sul microfono circa costante) questa sarà gaussiana, anche se con una varianza decisamente più ampia, dovuta alla non stazionarietà del processo. Un esempio è mostrato in figura 3.15, dove l'istogramma in verde è derivato dai subframe di rumore, mentre l'istogramma in blu, deriva dai subframe con parlato e rumore sovrapposti.
Senza anticipare il lavoro di Voice Activity Detection, svolto nel capitolo successivo, si noti che, lavorando con un rapporto segnale rumore significativo ($SNR > 12 \div 15\,dB$), il guadagno di codebook algebrico sarà un importante fattore di discriminazione tra voce e rumore. Infatti, con un approccio statistico al problema, per l'$n-$esimo subframe, basterà valutare se sarà maggiore la probabilità a posteriori di essere un subframe di voce $P(V / subframe_n)$ o di rumore $P(N / subframe_n)$ [47], usando la formula di Bayes, diventa:

$$P(V)\cdot P(subframe_n/V)\underset{N}{\overset{V}{>}}P(N)\cdot P(subframe_n/N) \qquad (3.3.2)$$

calcolo che diventa molto semplice: infatti, in generale in una conversazione normale, $P(V)=P(N)$ e quindi $P(subframe_n/V)$ e $P(subframe_n/N)$ diventano rispettivamente la probabilità di "etichettare" un subframe come *Voice* o *Noise*. Queste probabilità sono integrali derivanti da d.d.p. gaussiane con media e varianza *note* per entrambe e quindi tutto si baserà su una semplice valutazione di distanze da una soglia (stima *maximum-likehood*), che con SNR costante e rumore stazionario, potrà anche essere stabilita a priori (infatti, non è difficile immaginare di poter porre una soglia in figura 3.15 per la discriminazione tra voce e rumore).

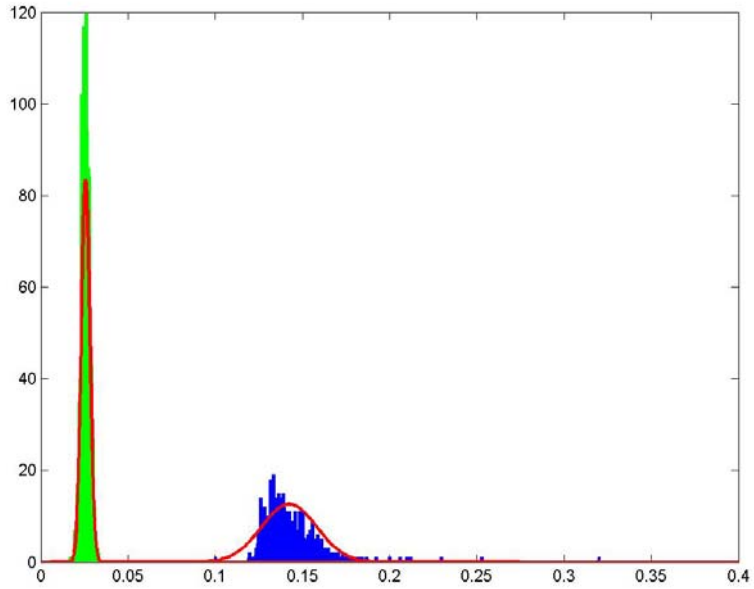


Figura 3.15 Distribuzione dei valori di $g_{codebook}(m)$ per rumore gaussiano bianco (verde) e parlato (blu) a 35 dB

Questo tipo di discriminazione sembra molto robusto, tuttavia i problemi nasceranno quando il rapporto segnale rumore comincia a scendere sotto i 10 dB, infatti, sarà possibile discriminare solo suoni vocalici i quali hanno un contenuto energetico maggiore, mentre la discriminazione per le consonanti sarà praticamente impossibile.

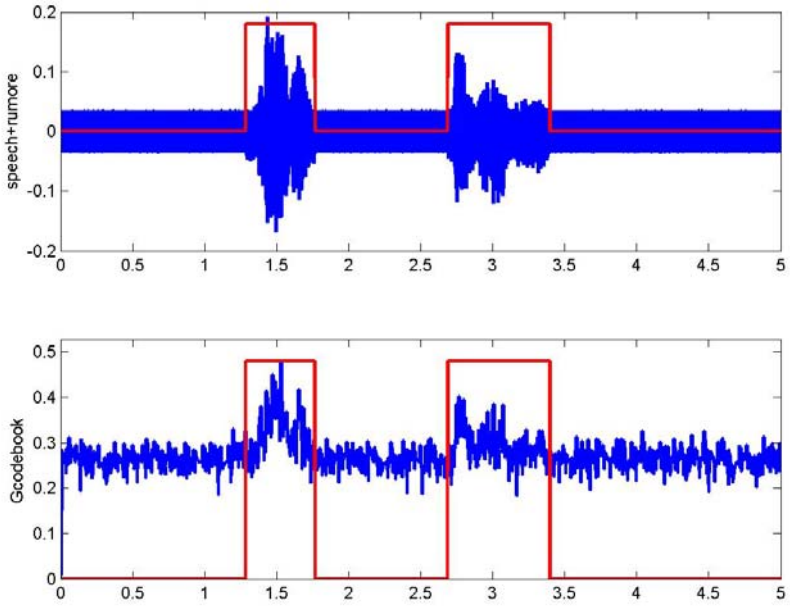


Figura 3.16 Andamento del guadagno di codebook algebrico, calcolato ogni intervallo temporale di $5\,ms$ sul segnale mostrato nella parte superiore ($SNR = 0dB$)

Attorno agli $0\,dB$ (figura 3.16) invece, qualsiasi tipo di discriminazione non sarà possibile, le due campane relative all'andamento statistico di voce e rumore saranno sovrapposte, a questo punto per un algoritmo di VAD questo parametro sarà totalmente inutile.

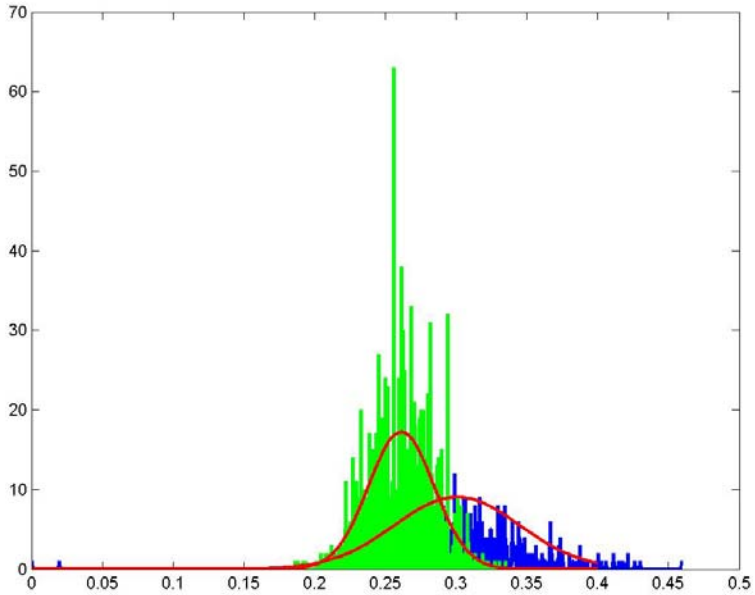


Figura 3.17 Distribuzione dei valori di $g_{codebook}\,(m)$ per rumore gaussiano bianco (verde) e parlato (blu) a $0\,dB$

Un esempio di questo è mostrato in figura 3.17. Si noti che sotto gli $0\,dB$ di rapporto segnale-rumore, la distribuzione di $g_{codebook}(m)$ per segmenti di voce tenderà, in media, ad essere sempre maggiore rispetto a quella dei segmenti rumorosi: questo è dovuto alla presenza nei segmenti classificati come *Voice* della presenza congiunta di voce e rumore.

Capitolo 4

Voice Activity Detection nel dominio codificato

Le tecniche di rilevazione dell'attività vocale nei segnali codificati sono un argomento molto interessante per le applicazioni nell'ambito dei sistemi di comunicazione. Il corretto funzionamento dei discriminatori di voce e rumore permette infatti di inviare sulla rete solo il segnale che veramente deve essere trasmesso, migliorando di conseguenza l'efficienza complessiva del sistema. Inoltre, durante le pause di parlato si trasmette il cosiddetto *comfort noise*, informazione sul rumore ambientale codificato con un numero di bit inferiore, al fine di evitare al ricevitore la fastidiosa sensazione di linea caduta. In più, l'informazione sulla presenza o assenza di parlato, permette ai blocchi VQE della rete, come vedremo nella parte relativa alla cancellazione d'eco, di operare una più efficace soppressione dei disturbi. L'idea centrale di ogni *VAD* consiste nel cercare di utilizzare uno o più caratteristiche che permettano la discriminazione fra voce e rumore. Tipicamente queste caratteristiche vengono cercate operando analisi nel dominio dei tempi e nel dominio delle frequenza; la novità del VAD presentato in questo capitolo è il suo funzionamento: esso infatti riceve in ingresso il segnale codificato ACELP esattamente come viaggia nella rete radiomobile, codificato con il codec AMR 122.

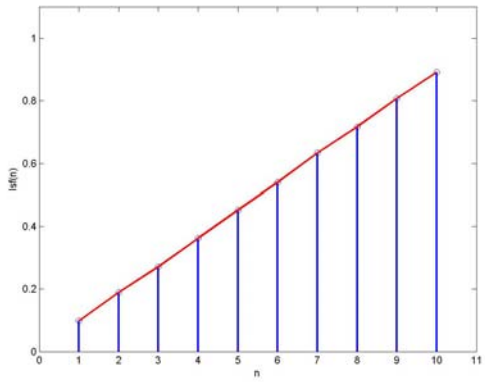
Il capitolo si apre con la presentazione delle manipolazioni svolte sui parametri per estrarre informazione per discriminare voce e rumore. Nel seguito viene mostrato il funzionamento del VAD soprattutto la sua adattività all'ambiente in cui opera. Infine saranno presentate le prestazioni.

4.1 I parametri utilizzati

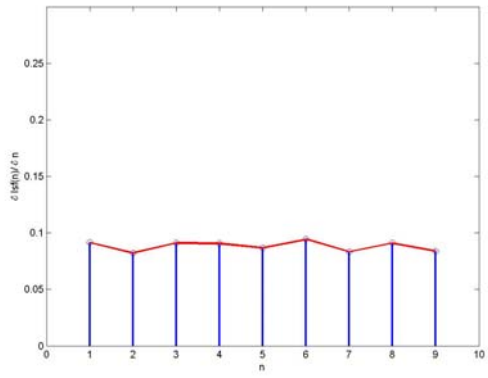
I parametri che si è deciso di utilizzare in ingresso al VAD implementato sono gli LSF, il ritardo di pitch e il guadagno di codebook algebrico; in questo capitolo ci occuperemo dell'elaborazione svolta su di essi per discriminare la voce dal rumore. Nel capitolo precedente sono state valutate le proprietà e le statistiche dei parametri principali del codificatore, analizziamo ora come queste proprietà possono essere usate nel nostro VAD.

4.1.1 Le linee spettrali di frequenza

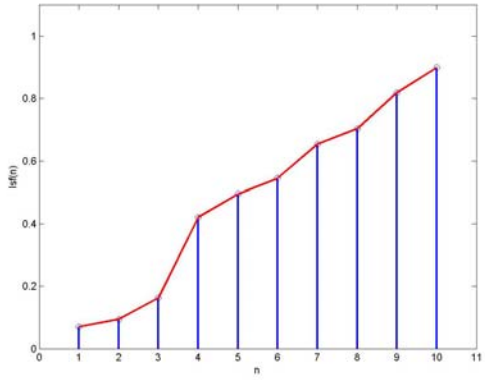
Per quanto riguarda gli LSF, questi risultano equidistanti tra loro nei subframes di rumore, mentre tendono a raggrupparsi nei subframe di parlato. Considerando la funzione $lsf(n)$, dove $n = 1,...,10$ e il valore puntuale corrisponde al valore del $n-$esimo LSF, questa approssimerà bene una retta in condizioni di rumore essendo gli LSF equidistanti, mentre non lo farà in presenza di parlato dove l'esistenza di una formante comporta un avvicinamento di 2 o 3 LSF nei dintorni in cui essa cade, come visto al capitolo precedente. Da queste considerazioni, possiamo affermare che la derivata di questa funzione sarà costante in presenza di rumore mentre non lo sarà in fase di parlato. E' possibile avere un riscontro di questa proprietà nella figura 4.1 dove sono rappresentati esempi della funzione $lsf(n)$ nei due casi di interesse: voce e rumore. Queste considerazioni portano ad analizzare la dispersione della funzione $lsf'(n) = \delta lsf(n)/\delta n$.



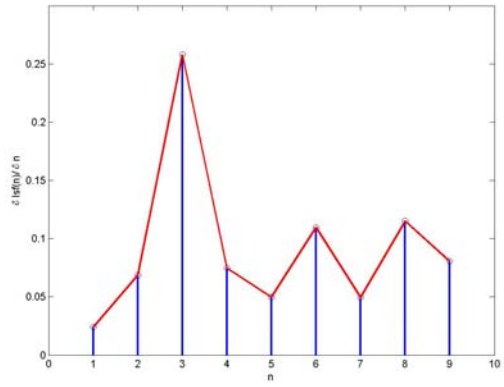
(a) $lsf\left(n\right)$ di subframe di rumore



(b) $lsf'\left(n\right)$ di subframe di rumore



(c) $lsf\left(n\right)$ di subframe di parlato



(d) $lsf'\left(n\right)$ di subframe di parlato

Figura 4.1 Confronto tra le funzioni $lsf\left(n\right)$ e $lsf'\left(n\right)$ di subframes di parlato e di rumore

Entropia

In condizioni molto rumorose si osserva che il segnale codificato, anche se l'energia del rumore di fondo è simile a quella del parlato, risulta più "organizzato" rispetto allo spettro in condizioni di solo rumore. Un a caratteristica appropriata per misurare tale ordine è l'entropia di Shannon definita come:

$$H\left(S\right)=-\sum_{i=1}^{N}P\left(s_i\right)\log_2 P\left(s_i\right) \tag{4.1.1}$$

dove $S=\left[s_1,...,s_N\right]$ rappresenta l'ingresso di cui si vuole misurare il disordine e $P\left(s_i\right)$ rappresenta l'ingresso normalizzato alla propria energia. Nel nostro caso l'approssimazione dello spettro deriva da quanto visto al capitolo precedente. L'entropia sarà massima quando saremo in presenza di rumore gaussiano bianco (spettro piatto e LSF equispaziati) e minima

quando l'ingresso è un sinusoide pura (spettro con un solo picco, LSF fortemente raggruppati). Viste queste considerazioni, sarà:

$$Entropia = -\sum_{n=1}^{9}\left[\frac{\left|lsf'(n)\right|^2}{\sum_{n=1}^{9}\left|lsf'(n)\right|^2}\log_2\left(\frac{\left|lsf'(n)\right|^2}{\sum_{n=1}^{9}\left|lsf'(n)\right|^2}\right)\right] \tag{4.1.2}$$

In figura 4.2 sono mostrati gli istogrammi dell'entropia di rumore e di parlato.

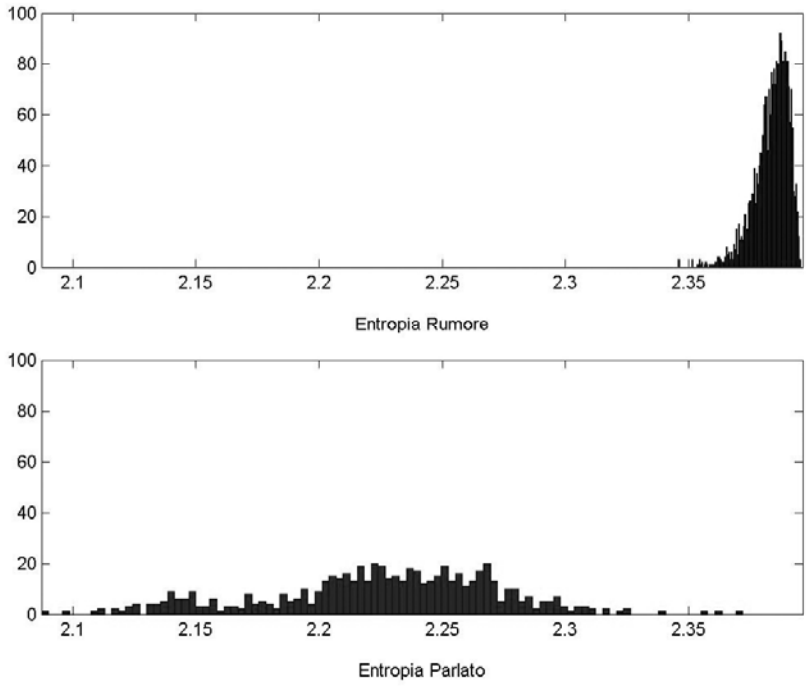


Figura 4.2 Istogrammi dell'entropia di rumore e di parlato

Dai grafici, ottenuto con rumore AWGN, si vede che l'entropia del rumore si distribuisce attorno al massimo possibile (corrispondente a una sequenza costante cioè *equiprobabile*), mentre l'entropia del parlato si distribuisce su valori minori, mostrando quindi un ordine molto maggiore dell'entropia del rumore. La dinamica dell'entropia è quindi limitata ed è stato possibile, con la simulazione, fissare il valore della soglia nella fase di training cioè a priori. Questo tipo di misura è abbastanza robusto rispetto al rumore. Tuttavia se il rumore è colorato tenderanno ad avere una distribuzione più ampia e quindi le prestazioni di questo metodo caleranno.

In figura 4.3 è mostrato un esempio dell'andamento della funzione *Entropia* in funzione del parlato.

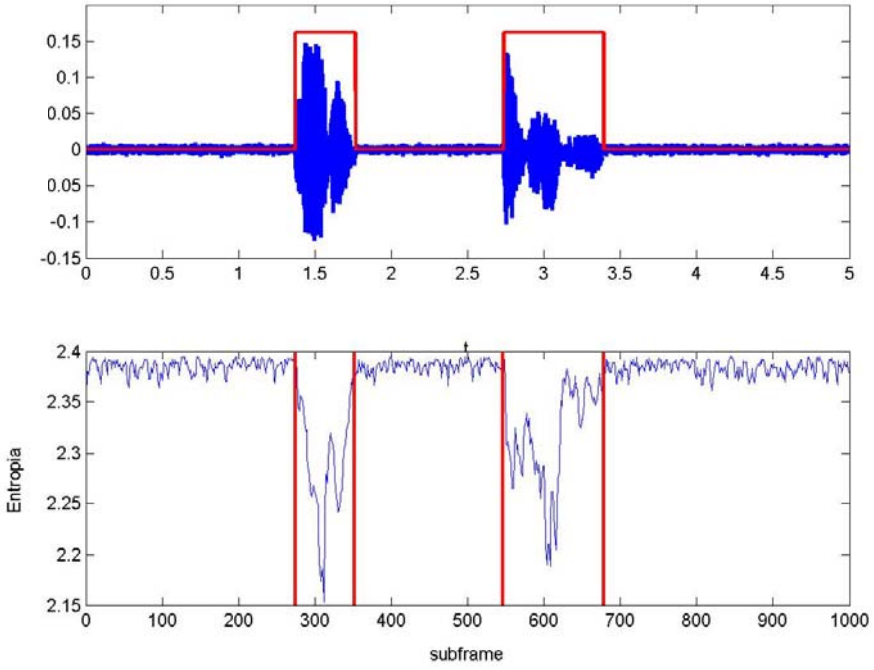


Figura 4.3 Andamento dell'entropia in condizioni rumorose

Le decisioni di VAD possono essere prese confrontando l'entropia con la soglia calcolata a priori; si rimanda questo passaggio alle sezioni successive in cui verranno approfonditi i metodi di sfogliatura e di decisione.

Varianza

Un'altra caratteristica utilizzata per valutare la dispersione della funzione $lsf'(n)$ è la varianza, definita come:

$$VarDiffLsf = -\sum_{n=1}^{9}\left[lsf'(n) - \frac{1}{9}\sum_{n=1}^{9} lsf'(n)\right]^2 \qquad (4.1.3)$$

Dato che la varianza indica la dispersione dei valori di una funzione, in presenza di rumore risulterà minima in quanto i valori della funzione $lsf'(n)$ sono all'incirca costanti, ed essendo quindi il valor medio circa uguale al valore considerato, l'argomento tenderà a zero. In presenza di parlato invece la varianza risulterà maggiore in quanto i valori della funzione si scostano molto dal suo valor medio. In figura 4.4 sono mostrati esempi di istogrammi della varianza in presenza di rumore bianco e di parlato.

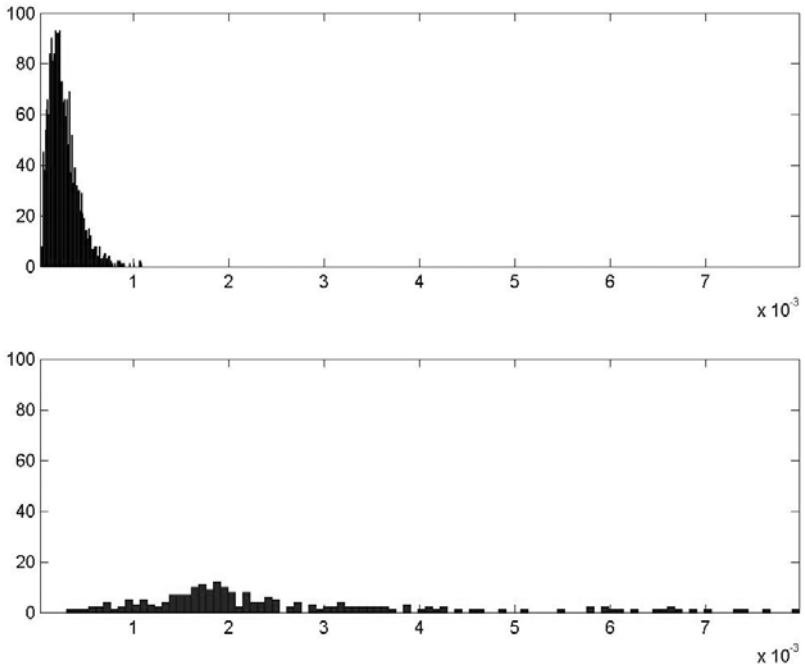


Figura 4.4 Istogramma della varianza di rumore e di parlato

Dagli istogrammi di figura 4.4, si ritrova ciò che è stato detto in precedenza: la regione occupata dal rumore è ben distinta dalla regione occupata dal parlato, infatti la varianza risulta minima per subframes di rumore, mentre è massima per subframes di parlato. Possiamo quindi affermare che anche i risultati giustificano il fatto che la varianza rappresenta una caratteristica di discriminazione tra voce e rumore. Tuttavia in condizioni di rumore colorato, l'affidabilità di questa caratteristica diminuisce, peggiorando così le prestazioni del sistema. Anche per quanto riguarda la varianza possiamo visualizzarne in figura 4.5 l'andamento e vedere la netta variazione tra la condizione di rumore e di parlato, come detto in precedenza. Per quanto riguarda la soglia della caratteristica, si rimanda alle sezioni successive in cui verrà approfondito meglio l'argomento.

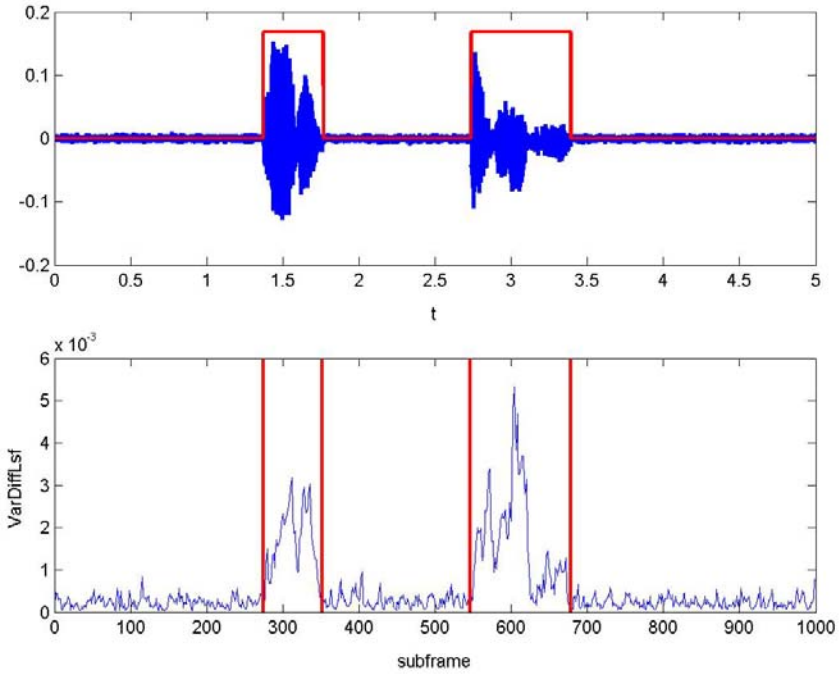


Figura 4.5 Andamento della varianza in condizioni rumorose

Conservazione delle vocali

Possiamo dire, almeno per le vocali dove la periodicità è più marcata, che gli LSF si raggruppano la dove è presente una formante; per le consonanti invece questo non è così evidente avendo caratteristiche spettrali molto simili a un rumore poco colorato. In questi casi sia l'entropia, sia la varianza non subiscono grandi variazioni, inducendo il VAD a prendere decisioni errata. Per evitare che ciò accada, è stato applicato un filtro ricorsivo a rilascio, il quale opera solamente se la funzione d'ingresso è minore dell'uscita. Questo è il principio del filtro di rilascio, che entra in gioco solamente se il segnale in ingresso sta decrescendo a seguito di un incremento dovuto alla presenza di un suono vocalico. La funzione analitica del filtro è:

$$\begin{cases} y(n) = a_{RT}\,x(n) + (1-a_{RT})\,y(n-1) & y(n-1) > x(n) \\ y(n) = x(n) & y(n-1) < x(n) \end{cases} \tag{4.1.4}$$

con $a_{RT} = 1 - e^{-5/N_{RT}}$, dove N_{RT} rappresenta il numero di campioni della risposta al gradino del filtro. In figura 4.6 è mostrato in blu l'andamento delle due caratteristiche per ogni subframe e in verde l'andamento dopo il filtro di rilascio. Il segmento di voce analizzato ha $SNR = 20dB$.

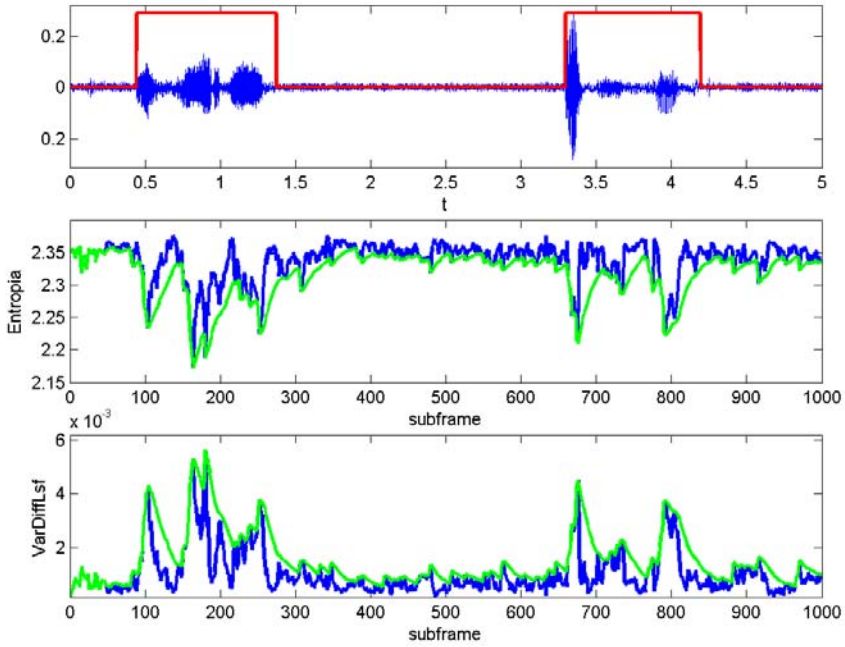


Figura 4.6 Andamento delle caratteristiche di Entropia e Varianza della Differenza

4.1.2 Il ritardo di pitch

Il ritardo di pitch fornisce una rappresentazione della periodicità del subframe analizzato rispetto ai tre precedenti, infatti è ottenuto valutando l'argomento che massimizza l'autocorrelazione tra i subframe correnti e i tre precedenti. Il parlato di tipo *voiced* nel dominio temporale è quasi periodico ed ha una struttura armonica nel dominio della frequenza, mentre il parlato di tipo *unvoiced* è circa casuale e a banda larga, così pure per il rumore.

Possiamo quindi dire che il ritardo di pitch si mantiene quasi costante durante il parlato, o meglio nei tratti dove sono presenti le vocali; questa caratteristica non è riscontrabile per i suoni unvoiced, dove la caratteristica del ritardo di pitch è più simile a quella del rumore. Dato che la decisione di VAD viene presa dopo un intero frame si è scelto di calcolare la varianza dei quattro valori di pitch, creando poi un intervallo di confidenza in cui ritenere se il frame è di parlato o di rumore. La caratteristica utilizzata sarà quindi:

$$\mathrm{var}Pitch = \sum_{n=1}^{4}\left[T_0(n) - \frac{1}{4}\sum_{n=1}^{4}T_0(n)\right]^2 \qquad (4.1.5)$$

In figura 4.7 sono mostrati gli istogrammi della varianza del ritardo di pitch calcolata per ogni frame in presenza di rumore bianco e di parlato. Si può notare dalla figura che la varianza di

pitch del parlato si distribuisce prevalentemente in valori molto bassi (dovuto al movimento quasi costante); la varianza del pitch del rumore invece si distribuisce invece su un'ampia gamma di valori.

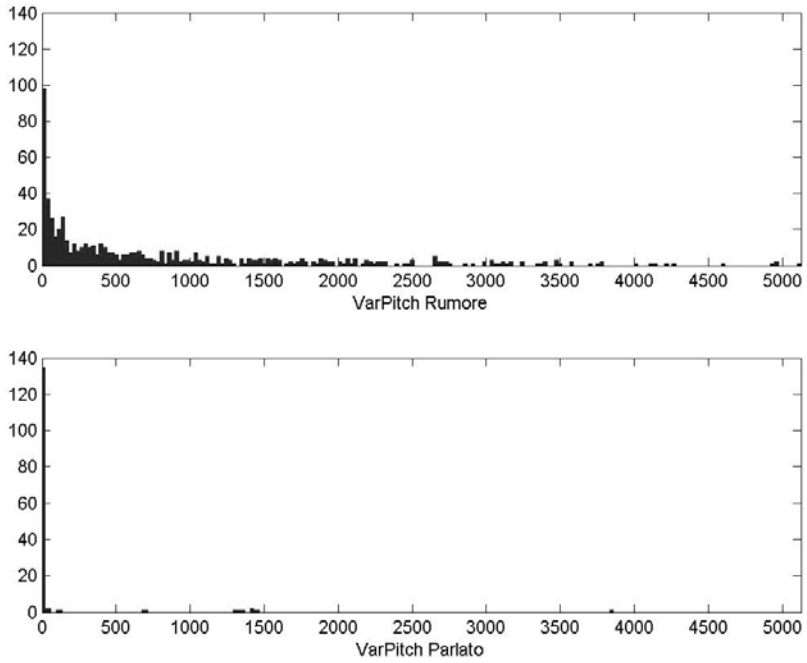


Figura 4.7 Istogramma della varianza di pitch di rumore e di parlato

La caratteristica utilizzata non è fortemente discriminante perché le regioni di appartenenza del rumore e del parlato non sono completamente distinte, ma si sovrappongono nei valori più bassi; bisogna comunque tenere conto del fatto che, se la varianza di pitch è bassa è più probabile che siamo in presenza di parlato invece che di rumore. Si è pertanto deciso di utilizzare anche questa caratteristica per la realizzazione del VAD.

Anche a questa caratteristica è stato applicato un filtro di rilascio per poter conservare la decisione di VAD durante frame di consonanti, dove le caratteristiche di pitch si avvicinano a quelle del rumore. In figura 4.8 è mostrato un esempio dell'andamento della caratteristica utilizzata per il ritardo di pitch in presenza di suono vocalico e di rumore con $SNR = 10dB$.

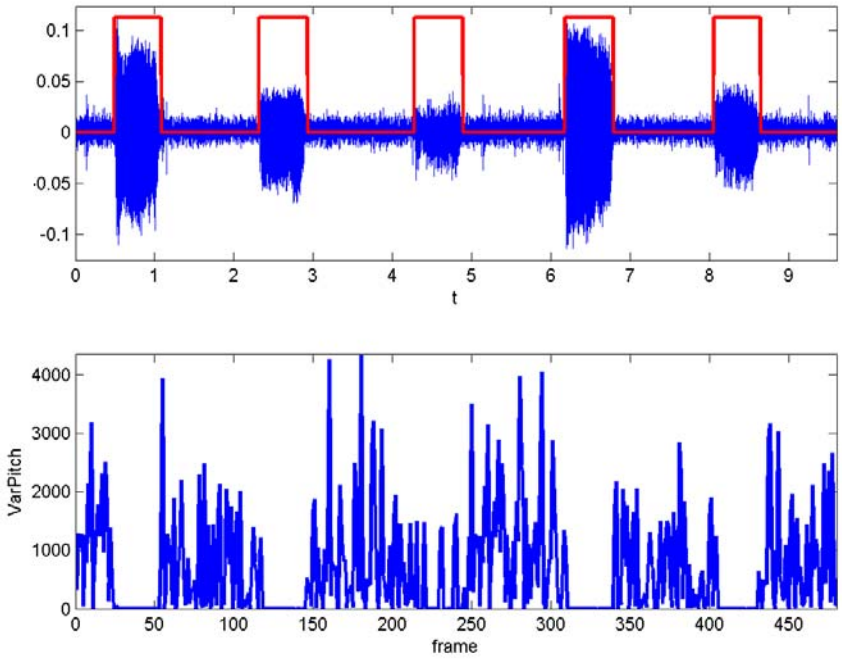


Figura 4.8 Andamento della caratteristica VarPitch in presenza di vocali

4.1.3 Guadagno di codebook algebrico

Il guadagno di codebook algebrico, come detto nei capitoli precedenti, è legato all'energia del segnale codificato, quindi se si verifica una brusca variazione del suo valore vi è una buona possibilità che si stia passando dallo stato di parlato allo stato di rumore o viceversa.

Facendo un'analisi sulla funzione dei trasferimento $H(z)$ in uscita da ogni subframe possiamo scrivere:

$$H(z) = \frac{g_c}{\left(1 - g_{pitch} z^{-T_{pitch}}\right)\left(1 - \sum_{k=1}^{10} a_k z^{-k}\right)} \tag{4.1.6}$$

Considerando quindi g_c come semplice fattore moltiplicativo e quindi direttamente legato all'energia del subframe.

Il guadagno non necessita nessun tipo di manipolazione per l'estrazione di informazione di classificazione Voce-Rumore, viene semplicemente filtrato per lo stesso filtro di rilascio visto nell'equazione 4.1.4. In figura 4.9 è mostrato l'andamento del guadagno ed in verde l'andamento dopo il filtraggio. L'esempio è stato creato con $SNR = 20dB$ e rumore *car*.

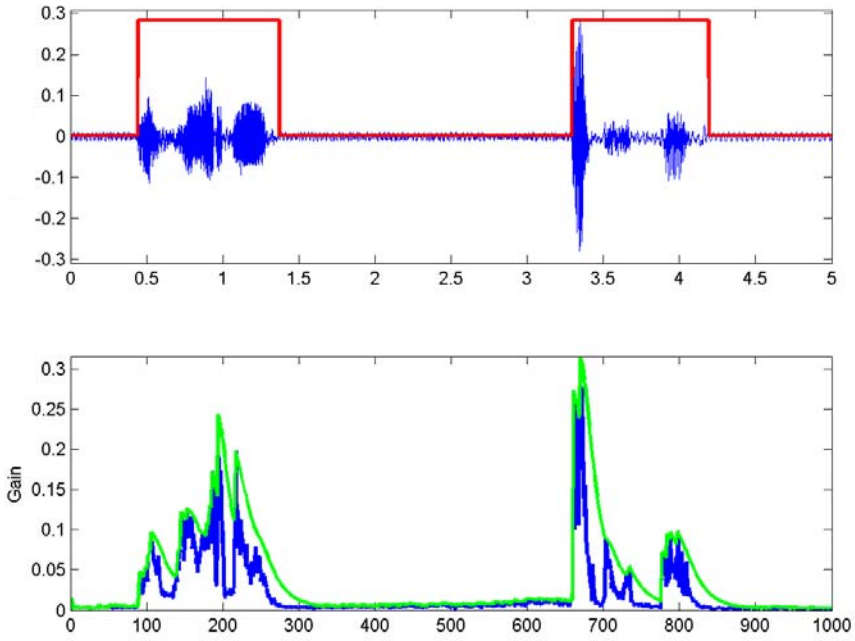


Figura 4.9 Andamento del guadagno di codebook senza (blu) e con l'uso del filtro di rilascio

Questo parametro è di difficile trattazione perché è molto dipendente dal *SNR* e non offre alcuna netta discriminazione quando questo è troppo basso. Il suo utilizzo tuttavia, valutando delle soglie a priori e sfruttandone la gaussianità è possibile comunque utilizzare il suo contributo per la realizzazione del VAD.

4.2 Struttura del Voice Activity Detector

In questa sezione verrà spiegato il funzionamento del VAD. All'inizio della conversazione telefonica si suppone che per i primi $100\,ms$ non ci sia parlato; questo intervallo di tempo verrà utilizzato per l'addestramento dell'algoritmo, valutando le statistiche delle caratteristiche utilizzate, l'entropia e la varianza degli LSF, la varianza del pitch e il guadagno. In questo modi si otterranno anche le soglie per poi compiere le decisioni di VAD. Successivamente a questa operazione di training, si provvederà per ogni subframe a calcolare ciascuna caratteristica utilizzata. Le decisioni di VAD vengono prese su più livelli per ciascuna e poi combinate tra loro ottenendo una decisione *sfumata* a più livelli sulle due uscite possibili.

L’ultima operazione consiste nell’aggiornare le soglie di decisione nel caso in cui il VAD=0, ovvero in assenza di parlato, per adattarsi all’ambiente. In figura 4.10 è mostato lo schema dell’algoritmo.

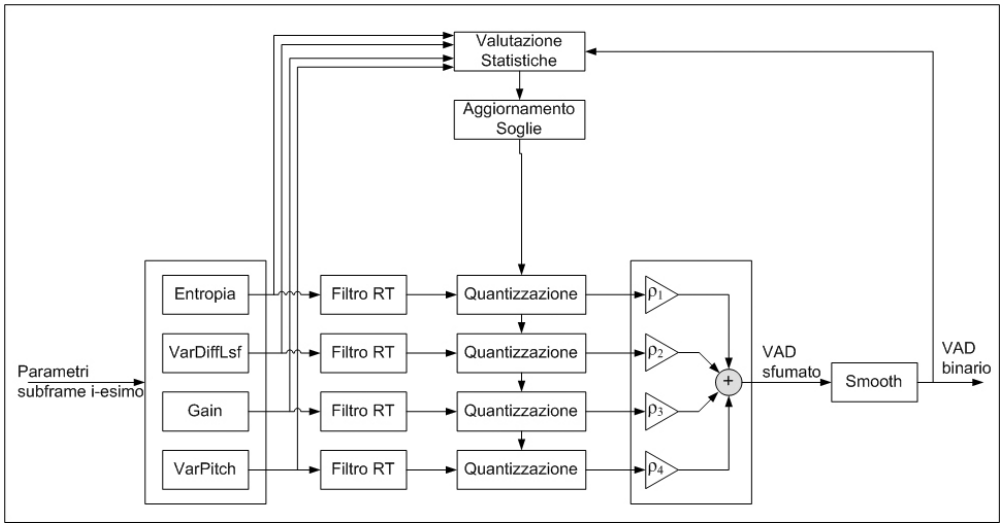


Figura 4.10 Schema a blocchi del VAD

4.3 Modalità di funzionamento

Il VAD ha tre modalità di funzionamento differenti: la prima è rappresentata dall’addestramento iniziale nei primi 100 *ms* della telefonata, la seconda dal funzionamento normale, cioè il VAD fornisce una stima dell’uscita desiderata, la terza, sfruttando la decisione di VAD presa per il subframe corrente, nella quale vengono aggiornate le soglie di quantizzazione in assenza di parlato.

In figura 4.11 è mostrato l’asse temporale e le modalità di funzionamento in una conversazione telefonica.

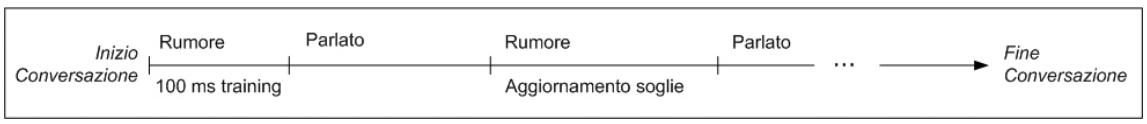


Figura 4.11 Modalità di funzionamento del VAD

4.3.1 Addestrameto di inizio conversazione

Come possiamo vedere dalla figura 4.11 si suppone che i primi 100 *ms* di una conversazione telefonica siano costituiti solo dal rumore, in modo da poter valutare le soglie di quantizzazione.
Durante la sequenza di training viene costruita una densità di probabilità, suddivisa in sei regioni di confidenza, ad ognuna delle quali sarà assegnata una certa probabilità di corretta decisione. In figura 4.12 è mostrato un esmpio utilizzando come caratteristica la varianza della funzione $lsf'(n)$.

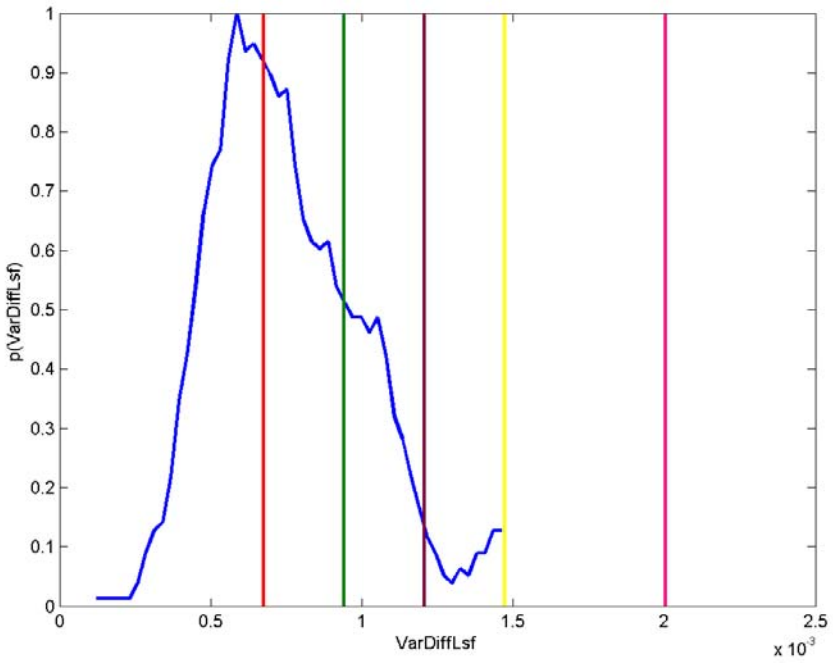


Figura 4.12 Densità di probabilità in fase di training della varianza degli LSF

Inizialmente per ogni caratteristica viene stabilita a priori la probabilità desiderata di corretta decisione. Successivamente si procede alla valutazione delle soglie corrispondenti con l'ipotesi che tale probabilità abbia una densità gaussiana. Di conseguenza si determineranno cinque soglie per la quantizzazione per ciascuna caratteristica considerata. Le probabilità che definiscono le soglie sono poste rispettivamente al 50%, 85%, 97% 99%, 99.9%.

4.3.2 Funzionamento normale

Dopo i 100 *ms* di training, il VAD entra in funzionamento calcolando ogni caratteristica per ogni subframe. Successivamente ogni caratteristica viene quantizzata secondo le soglie

stabilite durante la fase di addestramento; in questo modo si uniformeranno le misure nell'intervallo $[0,\ 1]$ e saranno pronte per essere combinate insieme secondo i pesi ρ determinati nelle sezioni successive. Una volta combinate insieme si avrà un'unica decisione di VAD su 24 livelli, la quale, tramite una *smoothing rule* verrà convertita nel valore 0 (assenza di parlato) o 1 (presenza di parlato).

4.3.3 Funzionamento durante le pause di parlato

Durante il funzionamento normale dell'algoritmo, se l'uscita del VAD binaria è a 0, a meno di errori, significa che siamo in presenza di rumore. Durante questi stati si provvederà all'aggiornamento delle soglie in modo che il sistema si adatti alle condizioni di rumore. Le modalità con cui si aggiornano le soglie è spiegato dettagliatamente nella sezione successiva.

4.4 Algoritmi di aggiornamento

4.4.1 Determinazione delle soglie di quantizzazione

Una volta presa la decisione di VAD binaria, come detto in precedenza, è necessario aggiornare le soglie fissate nella fase di training; questo è necessario perché se le caratteristiche del rumore cambiano le prestazioni del VAD peggiorano, in quanto l'addestramento è stato effettuato su un rumore con caratteristiche differenti. Con l'ipotesi di gaussianità delle caratteristiche utilizzate è sufficiente stimare il momento del primo e del secondo ordine; supponendo che il rumore sia non stazionario e lentamente variabile nel tempo, possiamo utilizzare uno stimatore che aggiorni ad ogni campione la stima di media e varianza per ottimizzare le soglie di decisione. L'aggiornamento di tali soglie verrà effettuato solamente in assenza di parlato e quindi se la decisione di VAD è pari a zero.
Lo stimatore del valore medio è realizzato con un filtro IIR ad un solo polo, dove l'ingresso è costitutito dalla media di una finestra di N campioni del parametro analizzato:

$$\mu(k) = a_\mu \mu(k-1) + \frac{1-a_\mu}{N} \sum_{n=k-N}^{k} x(n) \tag{4.4.1}$$

dove k rappresenta il subframe corrente, mentre il parametro a_μ costiuisce il tempo della risposta al gradino del filtro, in modo tale da controllare il tempo di convergenza del filtro.

Questo vale $a_\mu = 1 - e^{-5/N_\mu}$ dove N_μ rappresenta il numero di campioni della risposta al gradino del filtro.

Lo stimatore della deviazione standard è realizzato anch'esso tramite un filtro IIR ad un solo polo, dove l'ingresso è costituito dalla deviazione standard di una finestra di N campioni del parametro analizzato:

$$\sigma(k) = a_\sigma \sigma(k-1) + (1-a_\sigma) \sum_{n=k-N}^{k} \left| x(n) - \frac{1}{N} \sum_{l=k-N}^{k} x(l) \right| \tag{4.4.2}$$

Il parametro a_σ è la costante che controlla il tempo di convergenza del filtro, secondo la stessa relazione espressa per il valore medio. In figura 4.13 è mostrato l'andamento dei parametri e delle soglie in condizioni di rumore non stazionario *a gradino*, che rappresenta il caso peggiore per l'algoritmo.

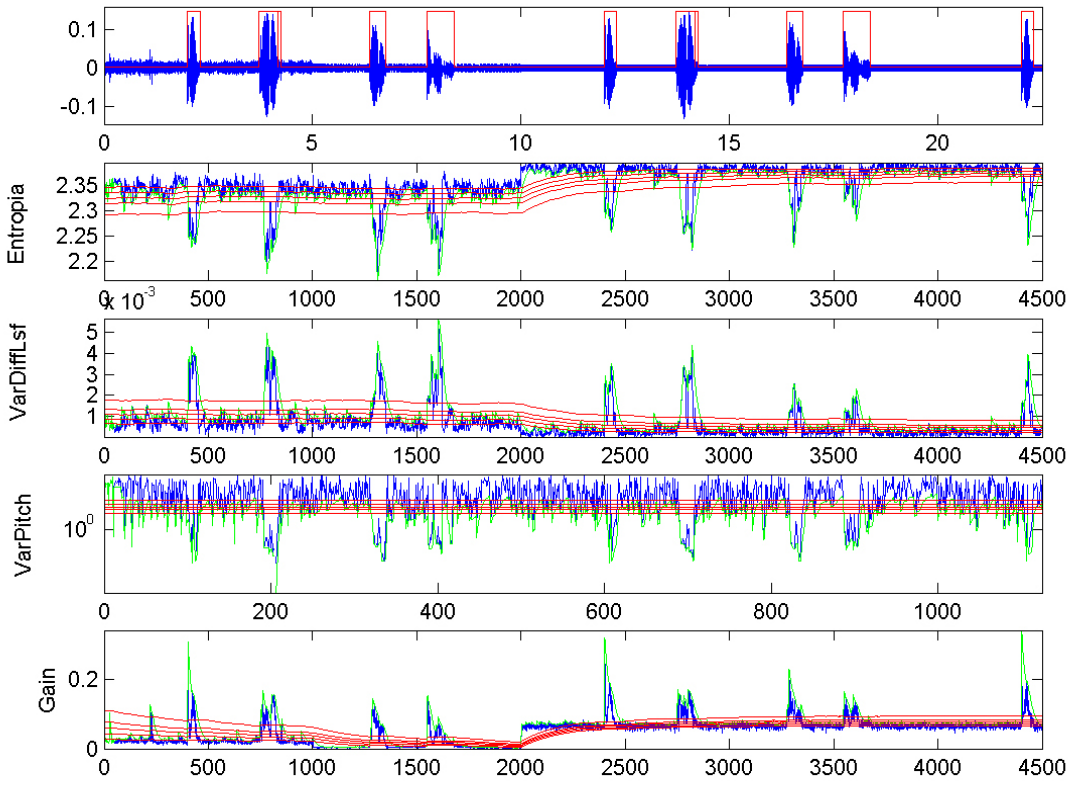


Figura 4.13 Andamento delle metriche con rumore non stazionario a gradino

4.4.2 Determinazione dei pesi

E' necessario combinare le decisisioni di VAD di ramo. Dalle considerazioni fatte e giustificate in precedenza, le caratteristiche più affidabili e più robuste risultano essere la varianza e l'entropia della funzione $lsf'(n)$; ci si aspetta che il guadagno di codebook algebrico sia significativo a buoni livelli di rapporto segnale-rumore mentre la varianza del ritardo di pitch sia significativa in sola presenza di suoni particolarmente armonici. Inoltre data la frequente ripetizione di consonanti nel parlato, l'affidabilità di quest'ultima risulta particolarmente ridotta. Da queste considerazioni si è scelto come metodo di combinazione una somma pesata dove il fattore di peso è stato scelto in base all'affidabilità e alla robustezza del singolo VAD, pesando quindi la varianza e l'entropia maggiormente rispetto al guadagno e alla varianza di pitch. Inoltre, essendo l'affidabilità della varianza di pitch bassa, il fattore di peso di questa, risulta essere minore di quella del guadagno.

Per calcolare questi fattori di pesatura sono state valutate le prestazioni di ciascun VAD. Il criterio per valutare l'affidabilità del VAD realizzato è basato sulla probabilità di corretta decisione e sulla probabilità d'errore. Per probabilità di corretta decisione si intende la probabilità che il VAD prenda la decisione corretta in presenza di parlato (probabilità condizionata). La probabilità d'errore invece rappresenta la probabilità che il VAD prenda la decisione errata in qualsiasi stato si trovi. E' di grande interesse la probabilità di corretta decisione perché quando nella comunicazione vi è parlato, la decisione deve essere il più possibile corretta. In figura 4.15 è mostrata la probabilità di corretta decisione e la probabilità di errore di ogni VAD in funzione del SNR, ottenuta combinando otto sequenze di parlato pulite da ogni rumore, in particolare due sequenze di voce femminile e sei di voce maschile, e cinque tipologie diverse di rumore: *babble, car, wgn, rain, street.*

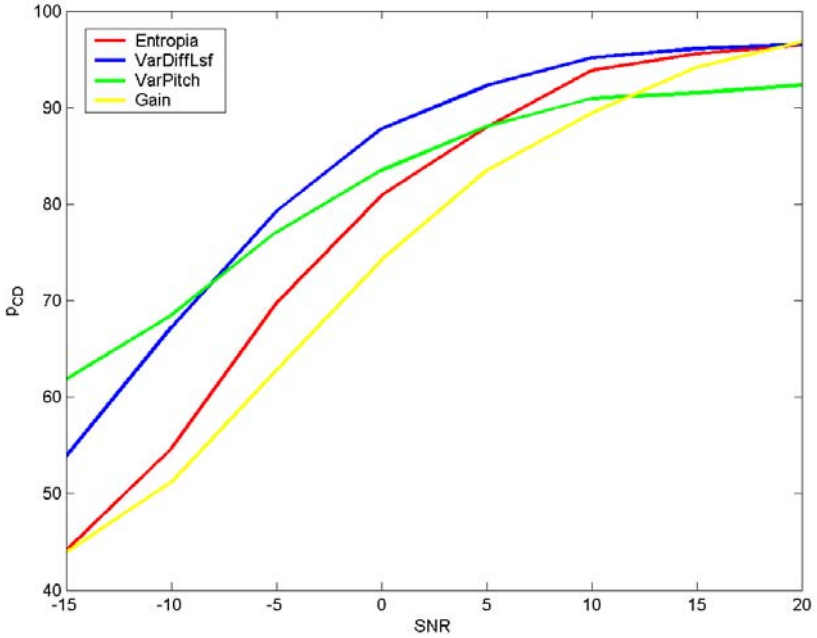


Figura 4.14 Andamento della probabilità di corretta decisione

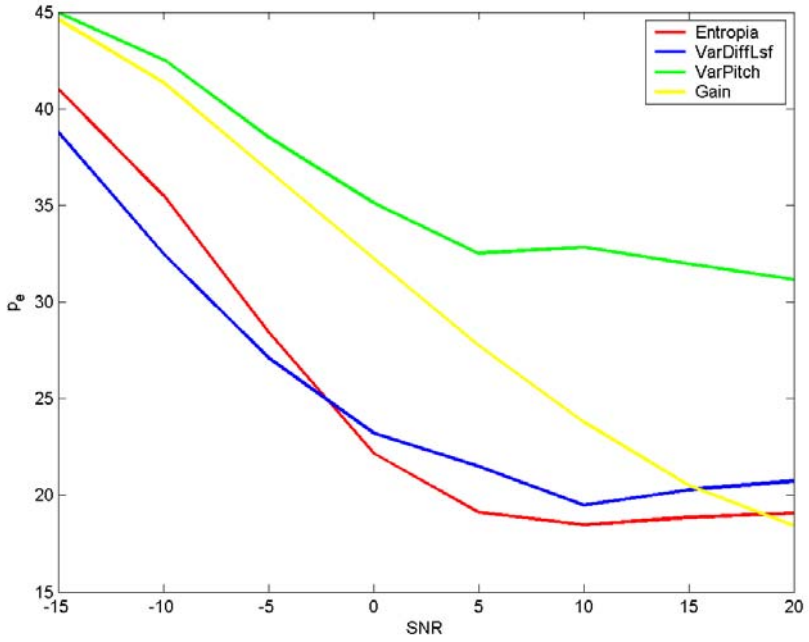


Figura 4.15 Andamento della probabilità di errore

Dalle prestazioni di ogni VAD si è quindi valutato il fattore di peso ρ, calcolato come:

$$\rho(i) = \frac{\sum_{SNR} p_{CD}^{i}(SNR)}{\sum_{SNR} p_{e}^{i}(SNR)} \tag{4.4.3}$$

Da questa relazione si può verificare che viene privilegiato il ramo con probabilità di corretta decisione maggiore e probabilità di errore minore. I risultati ottenuti mostrano che la caratteristica più affidabile è la varianza della derivata della funzione $lsf'(n)$, mentre quella più instabile è la varianza del ritardo di pitch. Il fattore di peso ottenuto, rispettivamente per l'entropia, la varianza della funzione $lsf'(n)$, la varianza del pitch e il guadagno, è il seguente: $[0.5505 \quad 0.5874 \quad 0.4038 \quad 0.4346]$.

4.4.2 Smoothing Rule

Una volta ottenuto un unico VAD come combinazione lineare di ogni ramo si applica un algoritmo di smoothing per eliminare eventi isolati, per esempio un falso allarme in uno o comunque pochi subframe, ed ottenere così una decisione binaria per ogni subframe. Si è visto che l'algoritmo fornisce buone prestazioni per la discriminazione di suoni *voiced – noise*, tuttavia si sono presentate delle difficoltà in presenza di suoni *unvoiced*, come le consonanti, dove l'intervento del filtro di rilascio non è stato sufficiente.

L'algoritmo di smooothing si basa sul fatto che un suono unvoiced non è mai un evento isolato ma è sempre seguito o preceduto da un suono voiced; ovviamente non si potranno correggere le decisioni prese in passato, ma queste potranno essere utilizzate per migliorare la decisione corrente.

Per prendere la decisione sul subframe corrente si tengono in conto le decisioni sfumate dei tre frames precedenti, se la media di queste decisioni con la decisione sfumata corrente è maggiore di un certo parametro h definito a priori empiricamente, allora la decisione di VAD sarà 1, altrimenti 0. Questo parametro è stato posto a 0.55. In equazione 4.4.4 è mostrato il funzionamento dell'algoritmo di smoothing:

$$VAD_{binario}(k) = \begin{cases} 1 & se \quad \frac{1}{15}\sum_{n=0}^{15} VAD_{sfumato}(k-n) > h \\ 0 & altrimenti \end{cases} \tag{4.4.4}$$

In figura 4.16 è mostrato un esempio della combinazione delle decisioni di VAD di ogni ramo, con a valle l'algoritmo di smoothing.

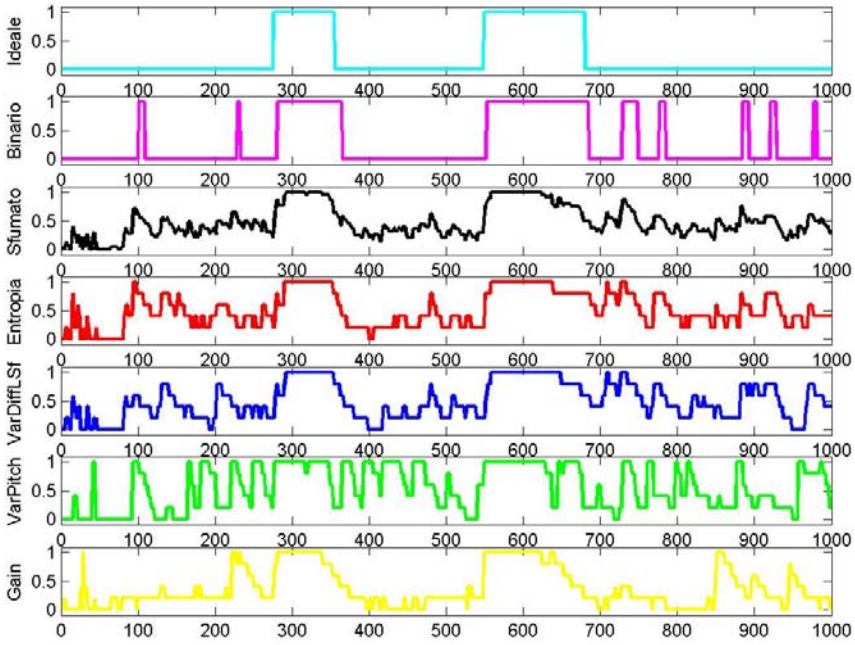


Figura 4.16 VAD totale

4.5 Prestazioni

Per valutare le prestazioni dell'algoritmo sono state prese diverse sequenze di parlato senza rumore, maschili e femminili, e sono stati annullati i campioni in cui non vi è parlato per creare così un VAD di riferimento. Successivamente ad ogni sequenza sono state sommate diverse tipologie di rumore: babble, car, rain e street, con diversi SNR compresi tra -15 e 20 dB. Nelle tabelle successive sono mostrate le probabilità medie di corretta decisione e di errore per diverse tipologie di rumore con SNR pari a $5\,dB$, $12\,dB$, $20\,dB$. Questi valori, usati in letteratura, rappresentano condizioni molto frequenti in una conversazione radiomobile.

RUMORE	p_{CD}%	p_{CD}%
WGN	88.87	10.45
RAIN	96.46	15.12
CAR	93.91	11.51
STREET	95.06	21.34
BABBLE	79.17	20.51
Globale	90.70	15.18

Tabella 4.1 Risultati VAD a 5 dB

RUMORE	p_{CD}%	p_{CD}%
WGN	94.11	9.34
RAIN	97.81	13.91
CAR	95.59	10.55
STREET	96.80	20.18
BABBLE	91.43	14.11
Globale	95.17	14.13

Tabella 4.2 Risultati VAD a 12 dB

RUMORE	p_{CD}%	p_{CD}%
WGN	96.17	8.18
RAIN	98.06	13.64
CAR	96.05	10.47
STREET	97.45	19.51
BABBLE	95.65	12.01
Globale	96.68	13.80

Tabella 4.3 Risultati VAD a 20 dB

In figura 4.17 è mostrata la probabilità di corretta decisione e in figura 4.18 quella di errore al variare del rapporto segnale-rumore.

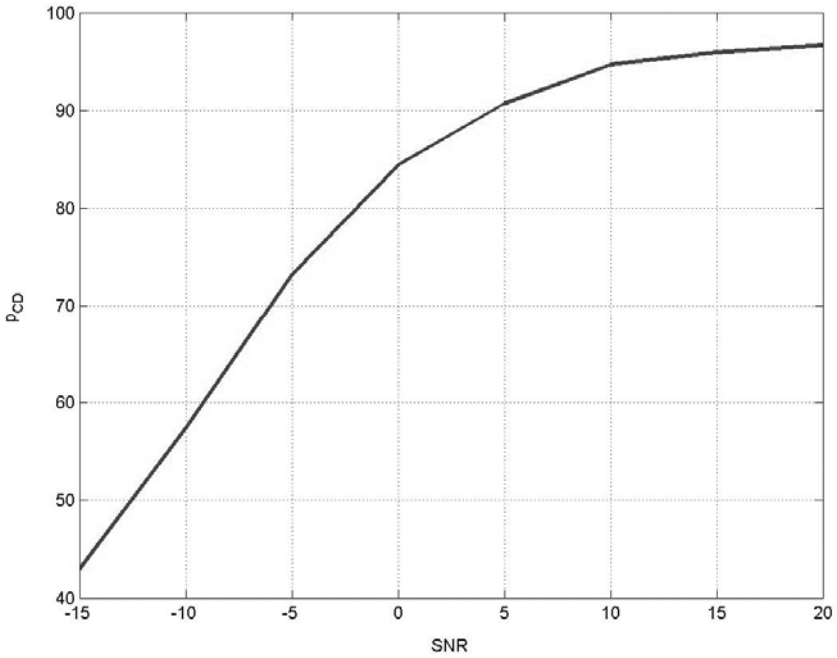


Figura 4.17 Andamento della probabilità di corretta decisione media

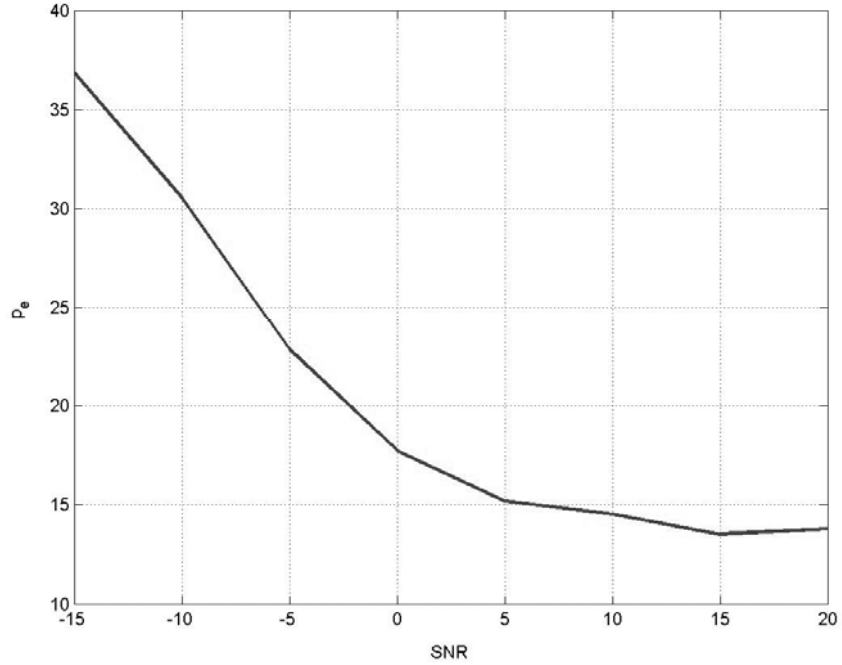


Figura 4.18 Andamento della probabilità d'errore media

Possiamo vedere dai risultati ottenuti che l'algoritmo è sufficientemente robusto anche ad SNR bassi fino a circa $0\,dB$, avendo una probabilità di corretta decisione pari al 84.5% e di errore pari al 17% risultato sicuramente accettabile. Questo risultato è principalmente dato dalla robustezza delle caratteristiche di entropia e di varianza degli LSF, le quali non degradano pesantemente in condizioni SNR pessime, rispetto al comportamento del gain e della varianza di pitch.

Analizzando le condizioni asintotiche possiamo vedere che l'algoritmo "si siede" al 96.7% per quanto riguarda la probabilità di corretta decisione e al 13.8% per quanto riguarda la probabilità d'errore.

La corretta decisione in media non supera questo valore perché la regola di smoothing non permette una transizione di VAD da 1 a 0 se non dopo 2-3 subframe, per evitare eventi isolati e quindi falsi allarmi; perciò l'algoritmo introduce un ritardo nella transizione dallo stato 0 allo stato 1.

La probabilità di errore invece, non riesce in media ad essere inferiore al valore asintotico, a causa il filtro di rilascio; se si verifica una transizione da 1 a 0, questa non sarà immediata appunto perché il filtro introduce un ritardo nella risposta pari alla risposta allo scalino del filtro.

Le valutazioni dell'algoritmo sono estremamente positive, infatti risulta scarsamente sensibile al SNR e molto efficiente nella conservazione dei suoni unvoiced, quindi delle consonanti, pur non degradando troppo le prestazioni asintotiche dell'algoritmo.

Capitolo 5

Cancellazione d'eco acustico nel dominio codificato

Dopo esserci occupati nei capitoli precedenti dello studio statistico dei parametri ACELP e della rilevazione dell'attività vocale implementando un *voice activity detector*, in questa terza parte del lavoro di tesi tratteremo della cancellazione d'eco acustico nel dominio codificato.
La necessità di controllare l'eco acustico sorge nel momento in cui l'altoparlante o *loudspeaker* e il microfono sono posti in maniera tale che il segnale emesso dal primo viene raccolto dal secondo.

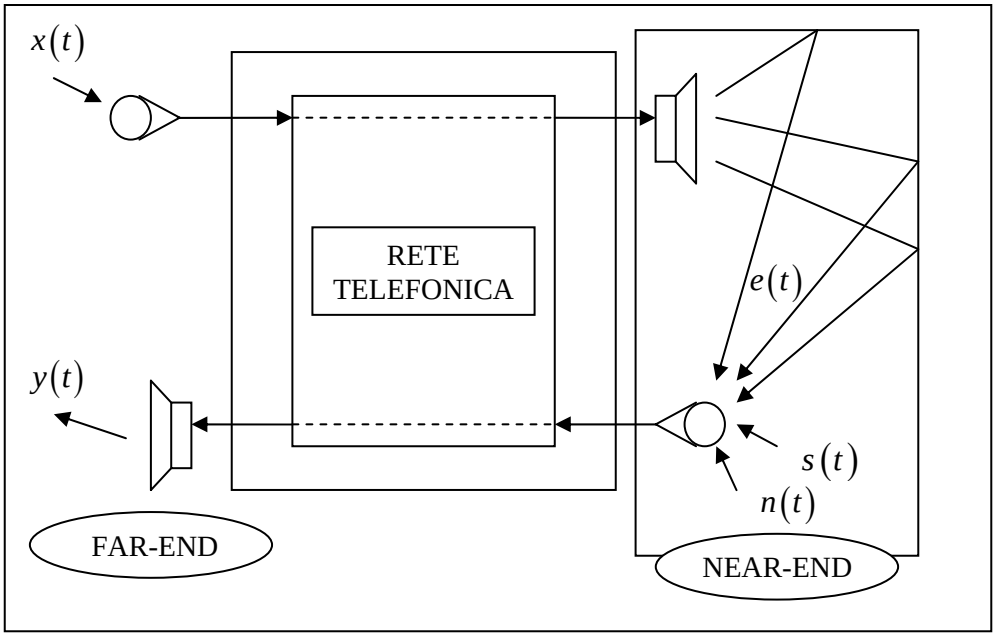


Figura 5.1 Scenario classico in cui si opera la cancellazione d'eco

In questo caso infatti il parlatore remoto, ovvero posto al *far-end*, viene disturbato dalla versione ritardata, più o meno distorta, della sua stessa voce. In figura 5.1 è mostrato un esempio tipico in cui l'eco è presente. Il segnale $x(t)$ proveniente dal *far-end* viene distorto nell'ambiente *near-end* rientrando nel microfono, tornando all'altoparlante del far-end.

Solitamente, nella concezione classica, il segnale non voluto $e(t)$ è visto come la convoluzione del segnale $x(t)$ con la risposta impulsiva dell'ambiente $h(t)$, ovvero $y(t)=s(t)+e(t)+n(t)=s(t)+x(t)\otimes h(t)+n(t)$, nella realtà invece sono presenti numerose non-linearità di cui bisognerà tener conto per applicazioni pratiche [45]. Inoltre, sarà necessario porre attenzione al fenomeno del *Double-Talk*, ovvero quando i due segnali $e(t)$ e $s(t)$, in un intervallo di tempo solitamente breve, si presentano sovrapposti.

I cancellatori d'eco acustico sono usatissimi nella telefonia moderna e molti algoritmi sono stati studiati e implementati [6]. Il dominio in cui si lavora, ovvero quello dei parametri della codifica ACELP, rappresenta una novità nel campo della cancellazione d'eco. Questo spingerà alla realizzazione di nuovi algoritmi, studiati apposta per la risoluzione dei problemi in analisi.

Il capitolo si apre con la presentazione di questo scenario e del fenomeno fisico dell'eco, leggermente diverso da quello classico in cui i cancellatori d'eco lavorano, come mostrato in figura 5.1.

In seguito viene presentato il rilevatore d'eco e l'inseguitore d'eco, entrambi necessari in un ambiente con ritardi variabili come quello degli apparati radiomobili. Infine, verranno presentati gli algoritmi implementati per la cancellazione d'eco.

In ogni sezione del capitolo saranno inoltre presentati, per validare le scelte effettuate, i risultati ottenuti con gli algoritmi implementati.

5.1 Descrizione del fenomeno fisico e dello scenario

5.1.1 Il fenomeno fisico

L'eco acustico, come si diceva precedentemente, deriva dall'accoppiamento tra altoparlante e microfono. In realtà, non si deve pensare che questo fenomeno sia del tutto negativo; le onde sonore ricevute dall'emettitore posto al far-end, fino a ritardi di 5-6 ms, non creano disturbo, ma anzi contribuiscono a dare alla conversazione un senso di "vitalità", come dimostrato da studi psicoacustici [55]. Tuttavia nel caso delle comunicazioni radiomobili, il ritardo di *round-trip* e la codifica e decodifica del segnale, fanno crescere il ritardo fino a rendere la presenza dell'eco particolarmente fastidiosa, solitamente quando $\tau_{delay} > 25ms \div 35ms$.

Il fenomeno fisico dell'eco che si andrà ad affrontare, come accennato nell'introduzione, è leggermente diverso da quello canonico studiato per i sistemi di videoconferenza o per i cosiddetti *hands-free devices*, tra i quali compaiono anche i sistemi vivavoce ormai obbligatori per comunicare quando si è alla guida. In questi sistemi infatti l'eco deriva principalmente dall'ambiente, il segnale in uscita dall'altoparlante, "rimbalza" all'interno di esso (ad esempio, sui muri di una stanza o sui vetri di un'auto) presentandosi al microfono dopo aver seguito numerosi percorsi, questo crea un fenomeno di *cammini multipli*, i quali si ricombinano, ciascuno con il suo ritardo e la sua attenuazione, al microfono, creando una replica distorta del segnale proveniente dal far-end che, unita al ritardo che necessariamente l'apparato telefonico introduce, risulta deleterio alla gradevolezza della conversazione per l'utente remoto.

Il fenomeno che andremo ad analizzare riguarda invece unicamente il terminale mobile. L'accoppiamento tra microfono e altoparlante deriva innanzitutto dalla dimensione ridotta dei terminali radiomobili e quindi dalla distanza tra i due elementi, a volte inferiore anche ai 7-8 *cm*. Il segnale esce dal loudspeaker ed entra nel microfono in due modi: tramite la propagazione in aria o tramite le vibrazioni dovute al comportamento non rigido del *chassis* (figura 5.2), ovvero l'involucro - di metallo e plastica - che ricopre l'apparecchio [32]. Inoltre, il chassis può presentare una risposta in frequenza non piatta e quindi distorcere lo spettro del segnale che lo attraversa [52].

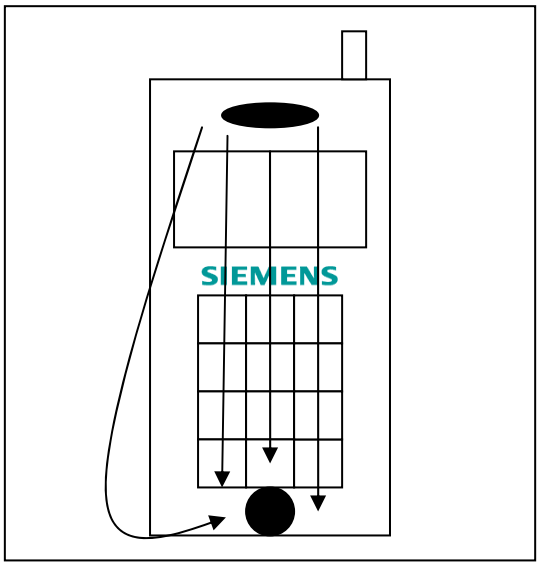


Figura 5.2 Accoppiamento Loudspeaker-Microphone tramite *chassis* e propagazione in aria

Tuttavia, nell'intervallo di frequenza considerato per il parlato rientrante ($80Hz - 4000Hz$) si può approssimare la funzione di trasferimento totale a $h(t) = \alpha\delta(t - \tau_e)$. Dove τ_e rappresenta il ritardo, di pochi millisecondi, della tratta loudspeaker-microphone. Il valore di attenuazione α è direttamente collegato quindi al parametro di *Echo Return Loss (ERL)*, ovvero il rapporto, espresso in decibel, che intercorre tra il livello energetico del segnale $x(t)$ e il livello energetico della sua versione rientrante nell'apparecchio $e(t) = x(t) \otimes \alpha\delta(t - \tau_e) = \alpha x(t - \tau_e)$:

$$ERL = 10\log_{10}\frac{E_x}{E_e} = 20\log_{10}\frac{V_x}{V_e} = 20\log_{10}\frac{1}{\alpha} = -20\log_{10}\alpha \qquad (5.1.1)$$

L'International Telecommmunication Union (ITU) detta gli standard minimi per la qualità del suono , tra i quali il minimo $ERL_{\min}$ consentito nelle comunicazioni radiomobili [21]; al progettista di un sistema di un *Acoustic Echo Canceller*, quindi, spetta il compito di cercare di attenuare l'eco per rispettare questo standard. In particolare lavorerà sul parametro di *Echo Return Loss Enhancement* (*ERLE*). Il parametro di *ERLE* rappresenta il miglioramento apportato dal sistema di cancellazione d'eco; in particolare, si cercherà di avere:

$$ERL + ERLE \geq ERL_{\min} \qquad (5.1.2)$$

Si noti che la richiesta di $ERL_{\min}$ non è fissa ma direttamente collegata con il *round-trip delay* τ_{delay}, infatti maggiore sarà questo, maggiore sarà la necessità di una cancellazione più robusta dell'eco. In figura 5.3 è mostrato il veloce decadimento del *Mean Opinion Score*, un fattore di merito sulla qualità del segnale vocale [23], a seconda del τ_{delay} in due situazioni con *ERL* differente (in rosso $ERL = 55dB$, in blu $ERL = 25dB$) [10]; il caso considerato è riferito a un segnale codificato con PCM a $64\,Kbit/s$ dove il MOS in assenza di disturbi ha il valore di 4.1 su 5 (massimo possibile) [22].

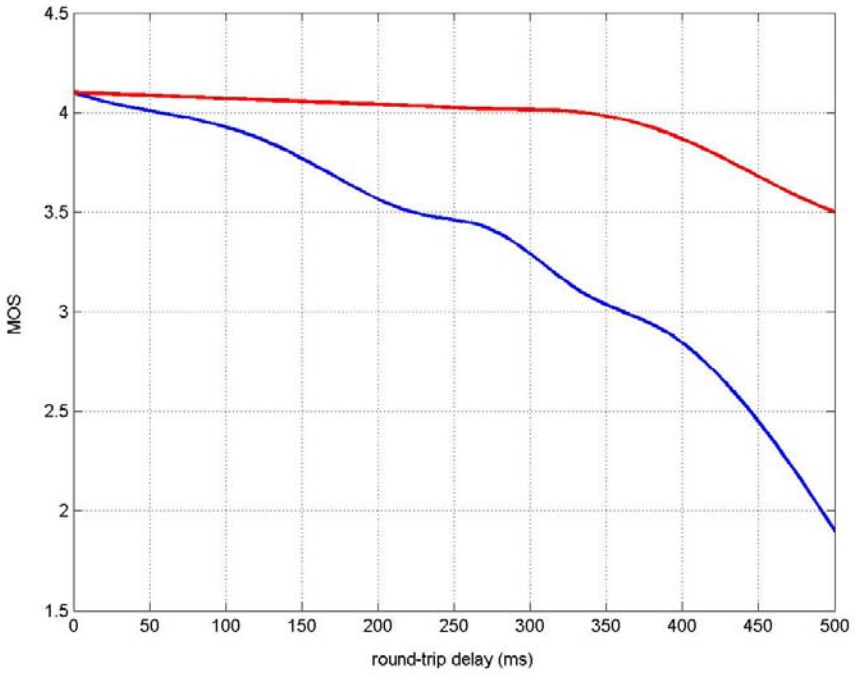


Figura 5.3 Variare del Mean Opinion Score a seconda del round-trip delay con $ERL = 55dB$ (rosso) ed $ERL = 25dB$ (blu)

5.1.2 Lo scenario: i sistemi radiomobili

Dopo aver parlato del fenomeno fisico e dello scopo principale di un cancellatore d'eco, è importante soffermarsi sulla particolarità dello scenario applicativo: il mondo della telefonia mobile UMTS e GSM.

Lo scopo di questa tesi, come viene illustrato nell'introduzione, è risparmiare al segnale il passaggio dal dominio parametrico al dominio PCM per l'attività di Voice Quality Enhancement e la successiva ricodifica. Questo risulta deleterio per il segnale in due modi: il primo è la distorsione da non-linearità introdotta dalla transcodifica, il secondo è il tempo necessario alle operazioni che porta ulteriore ritardo.

L'attività di Acoustic Echo Cancellation verrà svolta all'interno dei *Mobile Switching Center* (MSC) nei casi in cui la comunicazione avvenga tra due terminali radiomobili. Un problema di fondamentale importanza in questo contesto sarà quello di capire dove si trova l'eco; infatti, i ritardi su ciascuna tratta saranno variabili e non noti, necessitando di un "aggancio" tra i due assi temporali dei segnali provenienti dal near-end e dal far-end.

L’architettura di rete UMTS del contesto in cui si opera è mostrata, in versione semplificata, in figura 5.4. Le considerazioni fatte per il sistema UMTS, sono poi facilmente trasferibili al sistema GSM [25].

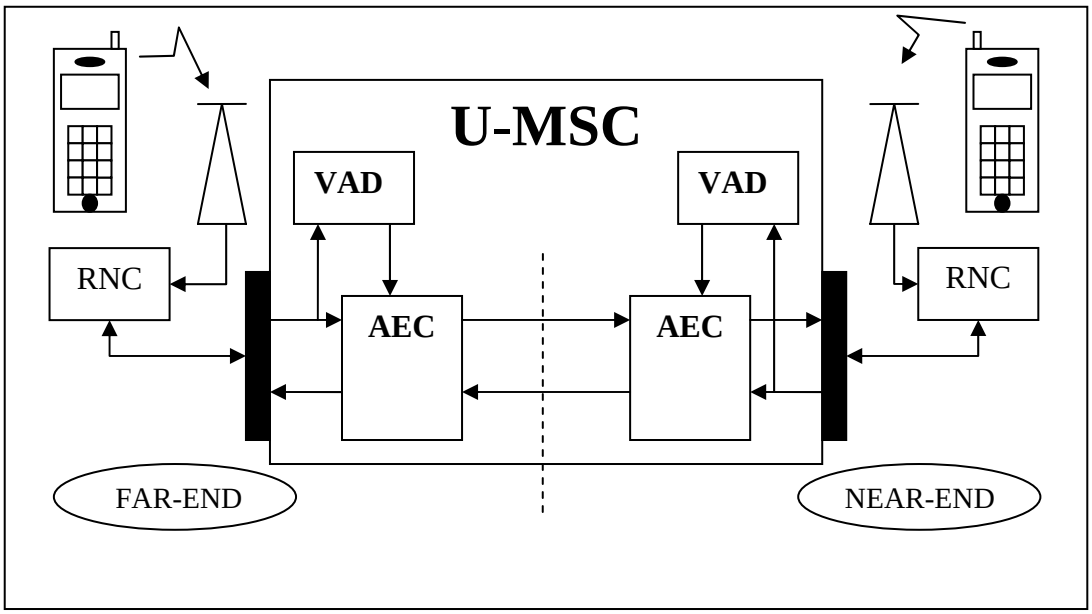


Figura 5.4 Architettura di rete GSM e collocazione degli algoritmi di VAD e AEC

Le operazioni di codifica Adaptive Multi-Rate del segnale vengono svolte sul terminale mobile senza nessuna azione di VQE ed inviate all’antenna del proprio cluster, chiamata *Node-B*. A questo punto il segnale viene inviato al *Radio Network Controller*, responsabile per il controllo delle risorse radio nella propria area. Il segnale vocale parametrico viaggia poi fino al Mobile Switching Center, dove è stato preparato un circuito dedicato alla comunicazione nel quale vengono svolte le operazioni di Acoustic Echo Cancellation e Voice Activity Detection. Ovviamente la cancellazione d’eco avviene su entrambi i lati della comunicazione, sarà comunque sufficiente implementarla sul lato sinistro (near-end) ipotizzando che dall’altra parte l’eco non sia presente per poi implementare lo stesso algoritmo anche sull’altro lato. Semplificando quindi lo schema di figura 5.4, otteniamo il modello su cui andremo a lavorare, mostrato in figura 5.5.

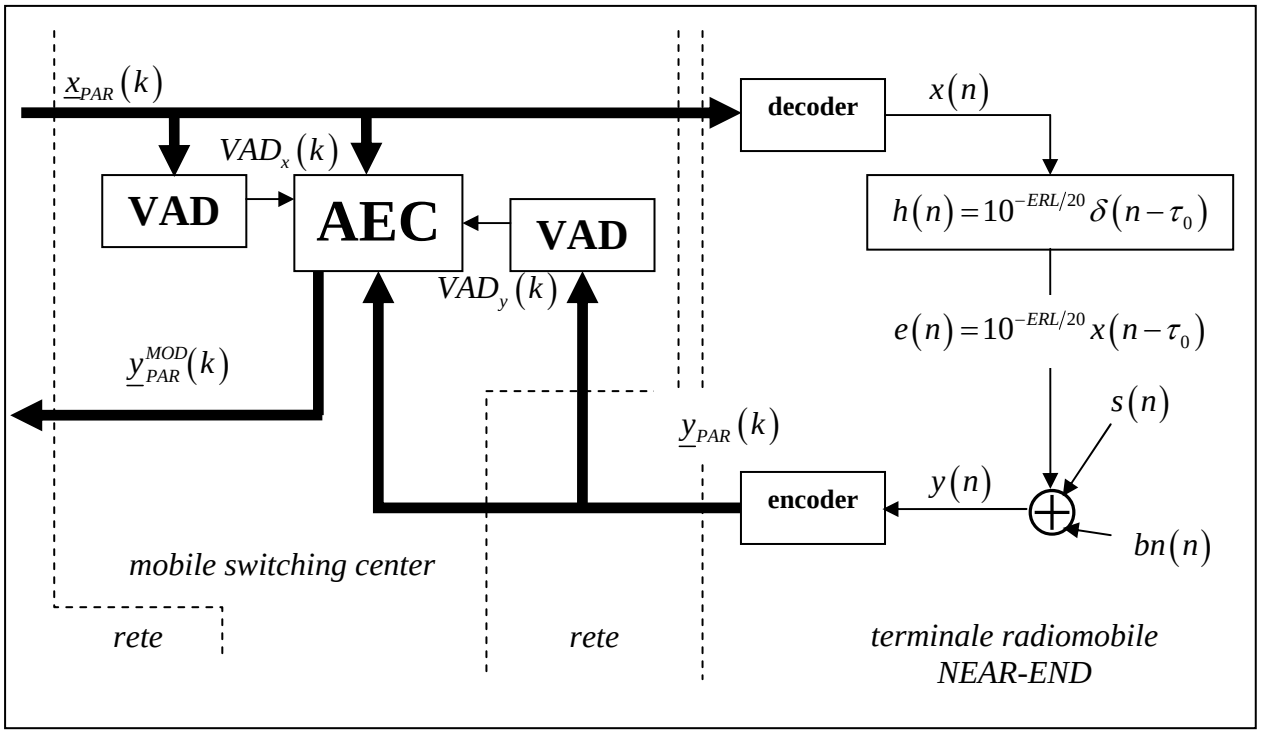


Figura 5.5 Modello in cui opera il cancellatore d'eco

Prima di vedere gli algoritmi implementati è importante fare qualche considerazione sul modello del sistema su cui si lavorerà. Innanzitutto si noti che il segnale in uscita dal decoder, subisce un'attenuazione e un ritardo. Il ritardo τ_0 viene modellizzato come ritardo totale del sistema, ovvero, include il ritardo del fenomeno di accoppiamento microfono-altoparlante, i ritardi di propagazione e i ritardi di codifica. Il parametro τ_0 avrà valori pari agli standard di ritardo per rete UMTS: $30ms \le \tau_0 \le 250ms$, con $\delta\tau_0 = 1/8000Hz = 125\mu s$. L'attenuazione invece sarà direttamente legata al ERL, come accennato in precedenza.

Il rumore di sottofondo $bn(n)$ può essere di vari tipi, già visti durante l'implementazione del VAD: car, street, babble, o, semplicemente, gaussiano bianco [8].

5.2 Analisi preliminari alla cancellazione d'eco

5.2.1 Il rilevatore d'eco

Nello scenario delle comunicazioni radiomobili, la voce deve rispettare dei limiti di ritardo massimo, solitamente attorno ai 300 ms, in modo da poter consentire una comunicazione gradevole tra due utenti. L'attività di cancellazione dell'eco, da quanto visto in precedenza,

non viene svolta dal terminale mobile (il telefono cellulare) ma negli U-MSC ovvero dagli apparati atti allo switching dei segnali in movimento da un terminale radiomobili all'altro. Pertanto, l'eco deve tenere conto dei ritardi delle tratte, approssimabili tra i 30 e i 250 ms. A questo proposito risulta fondamentale l'implementazione di un algoritmo che individui inizialmente il ritardo che intercorre tra il segnale proveniente al loudspeaker dall'utente posto al far-end e il suo eco proveniente dal microfono del near-end. Con questo algoritmo si troverà l'intorno in cui si muove il ritardo in modo da ridurre il range di analisi dell'inseguitore d'eco. Muovendoci nel dominio compresso, il range di ritardo tra i 30 e i 250 ms, diventa compreso tra i 6 e i 50 subframes (ogni subframe corrisponde a 5 ms di parlato). L'analisi quindi parte prendendo il primo segmento di segnale vocale utile (utilizzando quindi le decisioni di VAD) proveniente dal far-end, nel limite di 100 subframes (corrispondenti a 500 ms).

Una volta a disposizione questa porzione di segnale proveniente dal far-end, si cerca di trovare la posizione nel segnale che proviene dal microfono del near-end di pari lunghezza posto nella posizione corrispondente ai possibili ritardi: ad esempio, se il segnale x è compreso nell'intervallo di subframe pari a $[m,\ \ m+100]$, le misure correlative su ciascuno dei parametri verranno calcolate nell'intervallo $[m+6,\ \ m+150]$, un esempio è mostrato in figura 5.6.

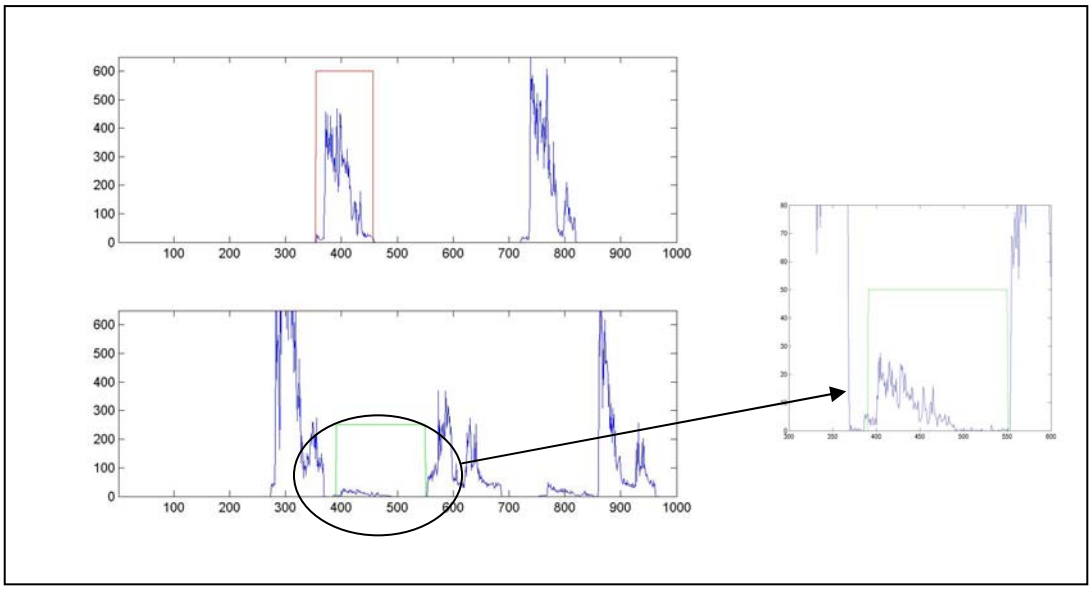


Figura 5.6 In verde è mostrato l'intervallo temporale su cui si effettua la misura correlativa tra i due segnali. In figura è mostrato l'andamento del code-gain con $ERL = 20dB$ ed $SNR = 35dB$

Il caso in considerazione ricorda molto da vicino quello della sincronizzazione di simbolo nelle trasmissioni numeriche. Inoltre, dall'analisi statistica svolta nel terzo capitolo, abbiamo

potuto verificare la gaussianità dei parametri coinvolti nella stima del ritardo: risulta quindi appropriato al tipo di problema la stima a massima verosimiglianza [5].

Come misura di verosimiglianza tra parametri, di cui poi andremo a prendere il massimo, si è scelta la cross-covarianza normalizzata, calcolata per ogni ritardo multiplo di $5ms$ nel *range* di ritardi possibili:

$$r_{xy}(\tau) = \frac{E\left[\left(x(n+\tau)-\mu_x\right)\left(y(n)-\mu_y\right)\right]}{\sqrt{E\left[\left(x(n+\tau)-\mu_x\right)^2\right]E\left[\left(y(n)-\mu_y\right)^2\right]}}, \quad \tau = 6,...,50 \tag{5.2.1}$$

Nell'equazione si sono posti x e y come due generici parametri provenienti dagli encoder rispettivamente del far-end e del near-end. I valori medi μ_x e μ_y vengono considerati costanti, in quanto, pur essendo vero che in 500 ms il parlato non si può più considerare stazionario, i parametri che ne rappresentano l'andamento invece lo sono. Si è scelta questa particolare misura in quanto la media dei due processi corrispondenti ai parametri analizzati è solitamente diversa, quello che più importa sono le fluttuazioni o la *dispersione* dei due processi attorno ai rispettivi valori medi [53].

A questo punto per ogni parametro - le dieci linee spettrali di frequenza, il tempo ed il guadagno di pitch e il guadagno del codebook algebrico - viene calcolata la (5.2.1); dopodiché se ne esegue la somma normalizzata al massimo possibile:

$$r_{xy}(\tau) = \frac{\sum_{i=1}^{10} r_{lsf_x i, lsf_y i}(\tau) + r_{T_x,T_y}(\tau) + r_{g_{pitch,x},g_{pitch,y}}(\tau) + r_{g_{fixed,x},g_{fixed,y}}(\tau)}{13} \tag{5.2.2}$$

Dove cadrà il massimo, questo sarà il ritardo stimato:

$$\hat{\tau}_0 = \arg\max_{\tau} r_{xy}(\tau) \tag{5.2.3}$$

L'errore quadratico medio della stima $E\left[\left(\hat{\tau}_0 - \tau_0\right)^2\right]$ sarà tanto maggiore quanto sarà il rumore presente su entrambi i segnali x e y. L'ulteriore presenza di non-linearità dovute al clipping, crea ulteriori problemi alla ricerca di $\hat{\tau}_0$, vedremo tuttavia che questa strada risulterà percorribile, offrendo risultati particolarmente apprezzabili.

Nella figura 5.7 è mostrato il risultato ottenuto con un ritardo noto a priori di 155 ms (31 subframes) ed $ERL = 20dB$: in blu è mostrato l'andamento con rapporti segnale-rumore nel dominio lineare pari a $SNR_x = 25dB$ e $SNR_y = 25dB$, mentre in rosso è mostrato l'andamento con $SNR_x = 0dB$ e $SNR_y = 0dB$. Si noti che la stima Maximum Likelihood, con la scelta del massimo della funzione $\hat{\tau}_0 = \max_{\tau} r_{xy}(\tau)$, risulta appropriata al tipo di situazione.

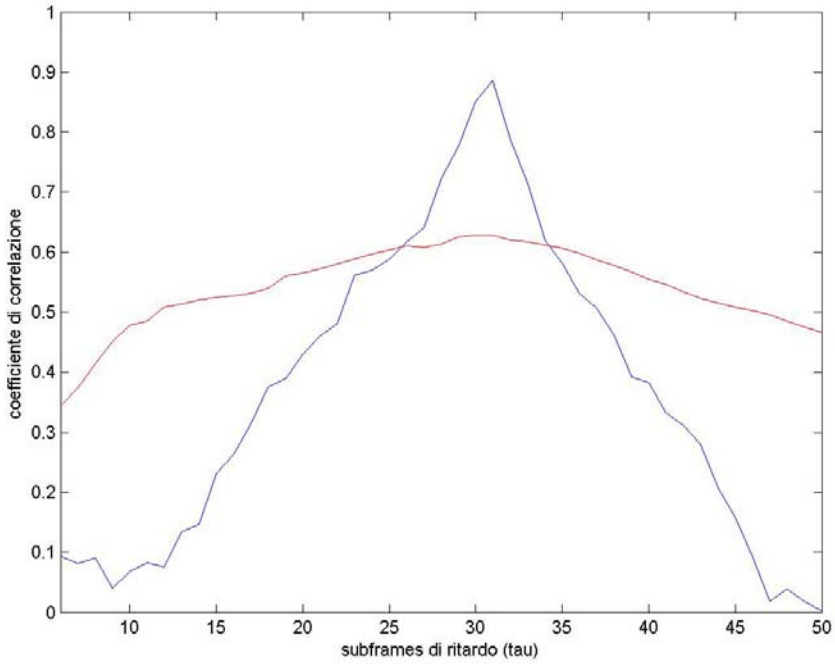


Figura 5.7 Andamento della funzione di cross-covarianza normalizzata $r_{xy}(\tau)$ con $SNR = 25dB$ (blu) ed $SNR = 0dB$ (rosso); le misure sono state effettuate con $ERL = 20dB$

L'uso di tutti e tredici i parametri disponibili per l'analisi rende robusto l'algoritmo. Inoltre, tramite uno studio sulla "qualità" dell'informazione correlativa portata da ciascun parametro, si è potuto dimostrare che, ad alto SNR, tutti i parametri sono validi per l'analisi mentre a basso SNR il movimento delle linee spettrali di frequenza (soprattutto di quelle coinvolte nella rappresentazione delle formanti) risulta ancora piuttosto marcato mentre il guadagno di codebook algebrico risulta quasi del tutto inaffidabile. Nella figura 5.8 è mostrato l'andamento dei coefficienti di correlazione del guadagno di codebook algebrico per vari SNR, con il medesimo echo return loss pari a $20\,dB$; nella figura 5.9, invece, è mostrato l'andamento della correlazione sul movimento delle linee spettrali alle stesse condizioni di SNR ed ERL.

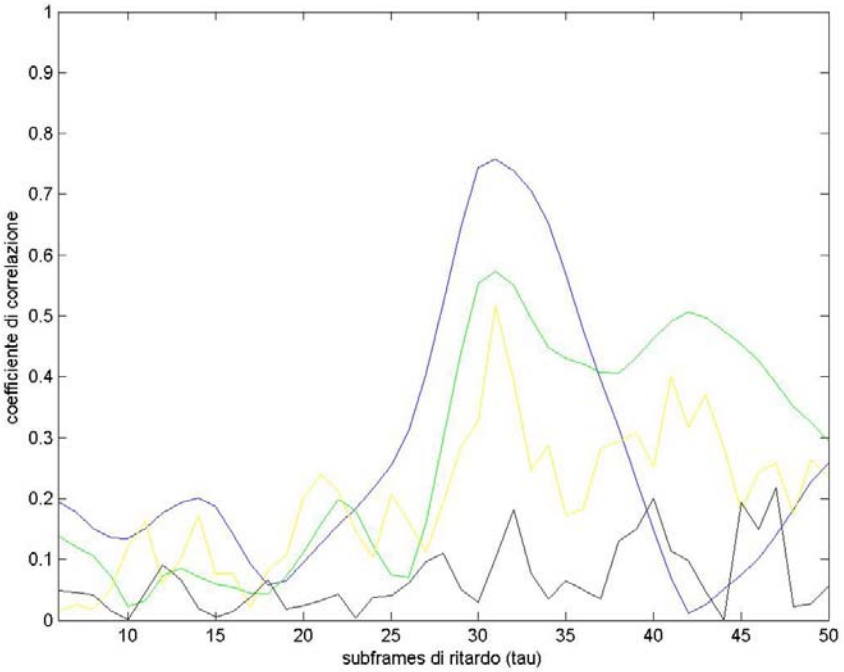


Figura 5.8 Andamento della funzione di cross-covarianza normalizzata $r_{xy}(\tau)$ svolta sul guadagno di codebook algebrico con $SNR = 30dB$ (blu), $SNR = 20dB$ (verde), $SNR = 10dB$ (giallo), $SNR = 0dB$ (nero); le misure sono state effettuate con $ERL = 20dB$

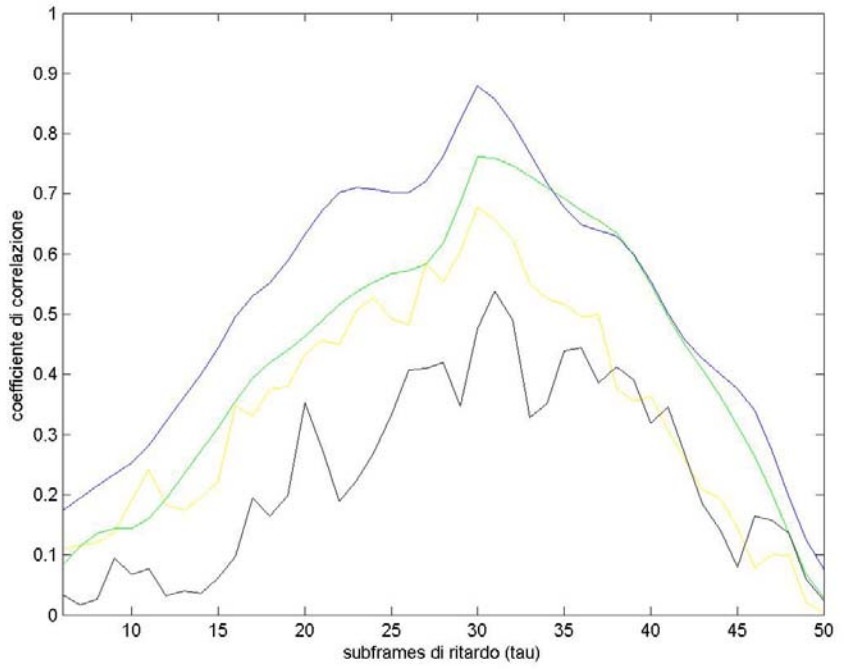


Figura 5.9 Andamento della funzione di cross-covarianza normalizzata $r_{xy}(\tau)$ svolta sui dieci LSF con $SNR = 30dB$ (blu), $SNR = 20dB$ (verde), $SNR = 10dB$ (giallo), $SNR = 0dB$ (nero); le misure sono state effettuate con $ERL = 20dB$

I problemi che incontra la stima del ritardo iniziale possono essere di due tipi: non avere una porzione di segnale proveniente dal far-end abbastanza grande o non avere misure correlative che portino a decisioni affidabili.

Il primo problema, non avere un segmento di segnale abbastanza grande, porta al non riuscire ad effettuare delle correlazioni significative. La questione viene superata imponendo che l'analisi non parte se il segmento è di durata inferiore ai 50 subframes (250 ms). Questo perchè potrebbe produrre risultati di correlazione non apprezzabili e quindi portare a decisioni sbagliate.

Il secondo problema, avere misure correlative inaffidabili, invece è particolarmente significativo infatti può derivare da numerosi fattori, ad esempio la presenza di Double-Talk al segnale rientrante nel microfono del near-end oppure dalla presenza di forte rumore su una tratta o sull'altra (o su entrambe) o un buon disaccoppiamento di altoparlante e microfono che creano un Echo Return Loss molto basso. A questo proposito si è stabilita una regola ad-hoc al problema. Infatti solo se:

$$\max r_{xy}(\tau) > 0.6 \quad \vee \quad \frac{\max r_{xy}(\tau)}{E\left[r_{xy}(\tau)\right]} > 2.5 \tag{5.2.4}$$

si pone $\hat{\tau}_0 = \max_{\tau} r_{xy}(\tau)$; altrimenti si passa ad un segmento successivo, distante 50 subframes (250 ms) dall'inizio di quello correntemente analizzato. La regola creata, in pratica, pretende di avere o un valore di massimo che sia una buona misura correlativa o di trovarsi in presenza di un picco ben definito nella cross-covarianza. In questo modo, si riduce la probabilità di sbagliare la decisione, senza peraltro danneggiare il segnale a livello psicoacustico. Questa situazione, infatti, è dovuta principalmente o alla presenza di Double-Talk, o alla presenza di basso ERL, o anche nel caso di basso SNR, senza escludere la combinazione di tutti e tre i fenomeni: tutti casi in cui l'eco non disturba (o lo fa in maniera minima) il parlatore al far-end.

Per misurare la bontà dell'algoritmo si è deciso di usare la deviazione standard dell'errore quadratico medio dell'errore di stima:

$$std\left(\hat{\tau}_0 - \tau_0\right) = \sqrt{E\left[\left(\hat{\tau}_0 - \tau_0\right)^2\right]} \tag{5.2.5}$$

essendo lo stimatore non polarizzato e quindi con $E\left[\left(\hat{\tau}_0 - \tau_0\right)\right] = 0$. Inoltre, per semplicità, si è considerato il segnale proveniente dal far-end senza rumore e le decisioni di VAD su entrambi i lati senza errori. In figura 5.10 sono mostrate le prestazioni dell'algoritmo.

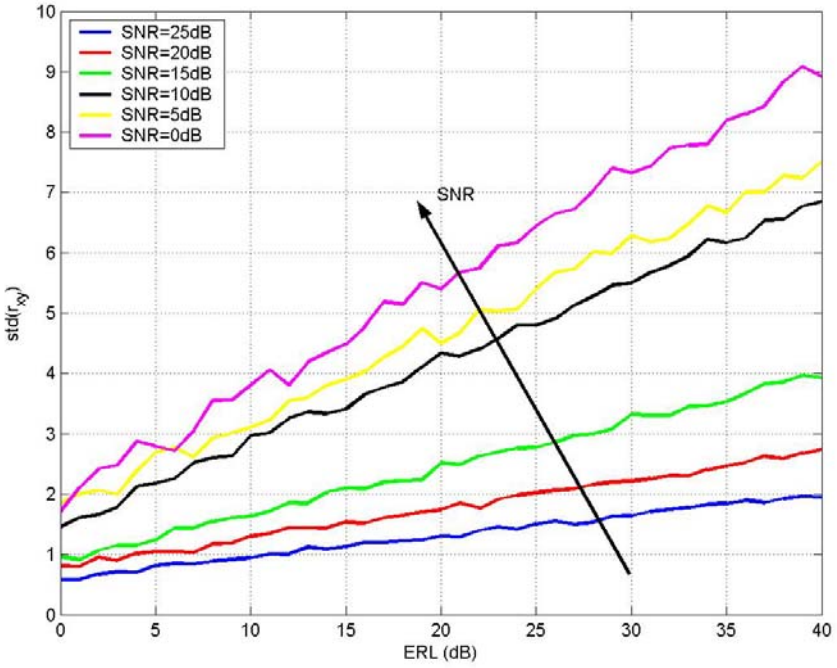


Figura 5.10 Andamento della deviazione standard dello stimatore del ritardo in funzione del ERL per vari valori di SNR al near-end

La deviazione standard, sotto l'ipotesi di gaussianità di $r_{xy}(\tau)$, è proporzionale alla larghezza della campana centrata in $\hat{\tau}_0$ e quindi alla varianza di essa. Nel nostro caso, questa rappresenta all'incirca l'ampiezza del range di valori su cui avviene la scelta della stima del ritardo $\hat{\tau}_0$.

5.2.2 L'inseguitore d'eco

Dopo aver trovato il ritardo che intercorre tra la voce rilevata dalla parte del far-end con la sua versione ritardata e distorta proveniente dal near-end, possiamo sincronizzare le sequenze e procedere con la cancellazione d'eco. Tuttavia durante la conversazione tra utenti radiomobili il ritardo sarà inevitabilmente variabile e quindi dovremo fare in modo di sopperire a questo problema creando un "inseguitore" che tenga aggiornato il tempo di allineamento tra i due assi temporali in modo da rendere il meccanismo di controllo dell'eco sempre efficiente.
A questo scopo risultano particolarmente utili le decisioni provenienti dal Voice Activity Detector, infatti quando questo sarà attivo sia sul segnale proveniente dal far-end sia sul segnale proveniente dal near-end (tenuto conto del ritardo), partirà la cross-correlazione tra i due segnali, ancora una volta eseguita su tutti e tredici i parametri utilizzati per l'analisi.

Quindi se all'n-esimo subframe ci troviamo nella situazione $VAD_x\left(n-\hat{\tau}_0\right)=1 \;\wedge\; VAD_y\left(n\right)=1$, si eseguirà la cross-correlazione tra i rispettivi parametri su un intervallo temporale di 50 subframes, ovvero sull'intervallo di x pari a $\left[n-\hat{\tau}_0-25,\quad n-\hat{\tau}_0+25\right]$ e per y pari a $\left[n-25,\quad n+25\right]$:

$$cc_{xy}\left(n,\tau\right)=\frac{E\left[x\left(m\right)y\left(m+\tau\right)\right]}{\sqrt{E\left[x^2\left(m\right)\right]E\left[y^2\left(m\right)\right]}},\quad \tau=-20,...,20 \tag{5.2.6}$$

Nell'equazione 5.2.6, al solito, si intende con x e y uno dei parametri usati per la rappresentazione del segnale.

Ciascuna funzione di cross-correlazione per ciascun parametro viene poi sommata alle altre e mediata:

$$cc_{xy}\left(n,\tau\right)=\frac{\sum_{i=1}^{10} r_{lsf_x i,lsf_y i}\left(\tau\right)+r_{T_x,T_y}\left(\tau\right)+r_{g_{pitch,x},g_{pitch,y}}\left(\tau\right)+r_{g_{fixed,x},g_{fixed,y}}\left(\tau\right)}{13} \tag{5.2.7}$$

A questo punto si prende il massimo valore di correlazione ottenuto e la sua localizzazione temporale:

$$\begin{aligned} cc\left(n\right)&=\max cc_{xy}\left(n,\tau\right)\\ \delta\hat{\tau}_0\left(n\right)&=\arg\max_{\tau} cc_{xy}\left(n,\tau\right)\end{aligned} \tag{5.2.8}$$

Il ritardo verrà aggiornato se $cc\left(n\right)>0.85$ tramite la semplice regola:

$$\hat{\tau}_0\left(n\right)=\hat{\tau}_0\left(n-1\right)+\delta\hat{\tau}_0\left(n\right) \tag{5.2.9}$$

A questo punto si allineano nuovamente gli assi temporali, mentre il valore di $cc\left(n\right)$ verrà passato al cancellatore d'eco come misura per dosare la velocità di convergenza degli algoritmi Least Mean Square o l'"aggressività" degli altri algoritmi. Questo parametro rappresenterà quindi la "somiglianza" tra segnale al far-end e segnale al near-end, sarà quindi un parametro di *echo likelihood.*

5.2.3 Il Double-Talk Detector

Uno dei problemi principali con cui un progettista di cancellatori d'eco acustico viene a scontrarsi è quello della presenza contemporanea della voce del parlatore al far-end e del parlatore al near-end. In questo caso, infatti, la scelta migliore è quella di un *freezing* delle operazioni. Questo avviene principalmente poiché, il cancellatore d'eco tenderebbe a rimuovere il parlato utile, proveniente dal near-end, che vorremmo recapitato al far-end.

Inoltre, questa situazione potrebbe portare a far divergere gli algoritmi LMS implementati. Precisamente, come vedremo nel seguito, l'aggiornamento del filtro, disturbato dal segnale al near-end, tenderebbe a non muoversi più in direzione inversa al gradiente dell'errore e quindi non avrebbe più significato il segnale in uscita dal filtro.

Nella sezione precedente abbiamo già trovato una misura, l'echo likelihood $cc(n)$, alla base degli algoritmi di double-talk detection più usati e collaudati ovvero quello basato sulla cross-correlazione tra segnale al near-end e segnale al far-end [7]. Infatti, è possibile studiare il comportamento statistico del segnale $cc(n)$ in uscita dall'inseguitore d'eco, per ottenere una discriminazione robusta. Sia in presenza di double-talk, sia in sua assenza, il segnale $cc(n)$ ha mostrato un comportamento gaussiano, con media e varianza abbastanza ben definite. Il comportamento medio con $SNR_y = 3dB \div 20dB$ ed $ERL = 10dB \div 30dB$, è mostrato in figura 5.11, con le densità di probabilità di $cc(n)$ ottenuta nei due casi. Si è posto per semplicità $SNR_x = 20dB$ fisso.

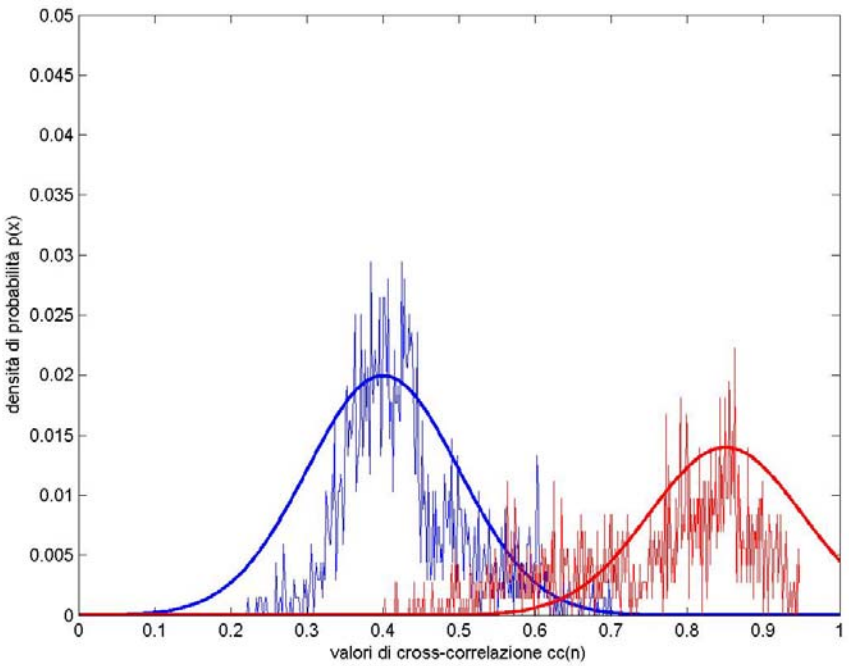


Figura 5.11 Densità di probabilità reali e stimate di $cc(n)$ nel caso di double-talk (blu) e in assenza di double-talk (rosso); le misure sono state effettuate con $ERL = 10 \div 30dB$ ed $SNR_y = 10 \div 30dB$ con $SNR_x = 20dB$ fisso.

Il risultato ottenuto tuttavia, non prende in considerazione il fatto che nella realtà la situazione di double-talk è rara in una conversazione normale e quindi è più realistico imporre una

probabilità a priori $P(D) \ll P(\bar{D})$. Nel seguito si è imposto $P(D) = 0.05$ [17]. Ora possiamo trovare la soglia ottima TH_{cc} per decidere in quale stato ci troviamo semplicemente eguagliando le due densità di probabilità gaussiane trovate. Il risultato ovviamente dipenderà dal livello di ERL e dal livello di SNR; tuttavia, per come sono stati progettati gli algoritmi di cancellazione d'eco, è stato sufficiente definire una soglia media per le situazioni in cui l'eco produce realmente fastidio: $TH_{cc} = 0.42$ è il valore trovato. La soglia infatti può scendere anche sotto TH_{cc}, tuttavia questo potrebbe essere dovuto o ad un ERL molto basso, e quindi un eco difficilmente percettibile, o ad un SNR basso, con un rumore che copre l'eco: entrambi casi in cui viene corrotta in maniera minima la qualità della comunicazione.
Per misurare le prestazioni si è scelto di effettuare le misure con il rapporto NFR, *near-end to far-end ratio*; questo tipo di parametro viene utilizzato solitamente per misurare le prestazioni dei Double Talk Detector. Nel nostro caso, lavorando su segnali con la medesima dinamica, diventa $NFR = ERL$. Le prestazioni si misurano tramite le due probabilità $P_m(E)$ e $P_{fa}(E)$, rispettivamente la probabilità di non rilevare la presenza di double-talk quando questo avviene e la probabilità di rilevare la presenza di double-talk quando questo non è presente. Con la soglia fissa, decidendo di non sbilanciare l'algoritmo (modificando TH_{cc}) per favorire la riduzione di una o dell'altra, risulta $P_{tot}(E) \approx \left(P_m(E) + P_{fa}(E)\right)/2$. I risultati per vari SNR sono mostrati nei due grafici in figura 5.12.

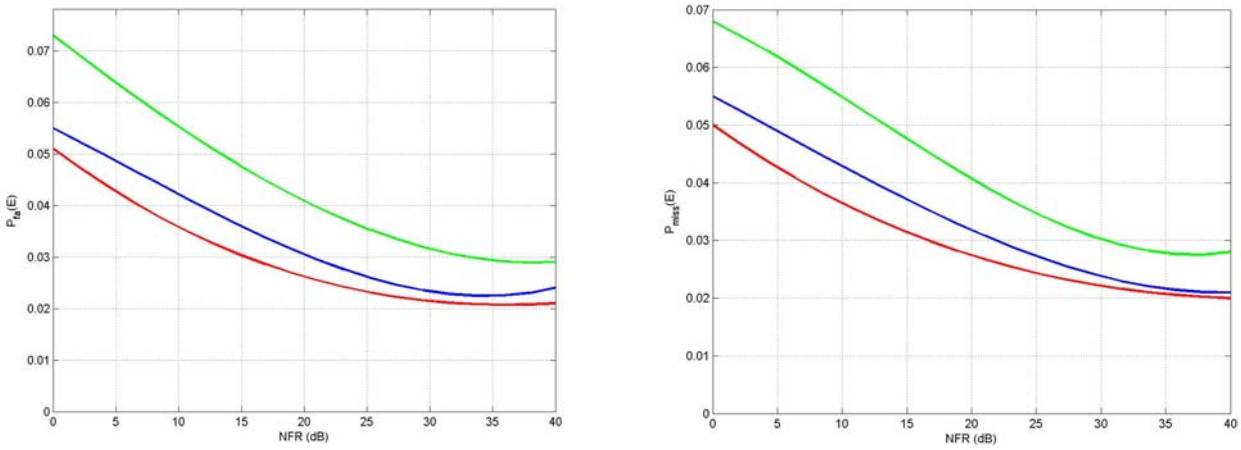


Figura 5.12 Probabilità di false-alarm $P_{fa}(E)$ e probabilità di miss $P_m(E)$ in funzione del rapporto tra segnale di near-end e segnale di far-end. Le misure sono state effettuate con tre differenti livelli di background noise: 5 dB (verde), 10 dB (blu), 20 dB (rosso).

Per valori ragionevoli di ERL, difficilmente inferiore ai 15 dB, quindi l'algoritmo offre prestazioni asintoticamente molto interessanti. L'algoritmo tuttavia tenderà per valori di $NFR < 0dB$ o $NFR > 40dB$ ad accrescere la probabilità d'errore. Questo è dovuto alla scelta fissa della soglia. Tuttavia, ci troviamo fuori dalle condizioni operative reali e quindi soluzioni con soglie adattative non sono state prese in considerazione.

5.2.4 Problemi realizzativi del rilevatore e dell'inseguitore d'eco

Uno dei principali problemi incontrati nella realizzazione degli algoritmi presentati in questa sezione è stato quello di dover lavorare con intervalli temporali di 5 ms. Ricordando quanto visto nel secondo capitolo, questo periodo non è nient'altro che l'intervallo temporale di 40 *samples* presi campionando a $8\,KHz$ il segnale tempo continuo, il corrispettivo di un subframe. A questo proposito si è ritenuto opportuno analizzare il comportamento del rilevatore e dell'inseguitore dell'eco per intervalli temporali variabili, non multipli di 5 ms. In questa sezione, ci riferiremo al ritardo aggiuntivo non multiplo di $5\,ms$ come τ_e.

Questi infatti offrono una stima del ritardo con un $\delta\tau_0 = 5ms$ contro un $\delta\tau_0$ reale pari a $1/8000 = 0.125ms$. Ci aspettiamo quindi che il rilevatore d'eco sbagli la stima iniziale del ritardo e che, in seguito, l'inseguitore d'eco, faccia oscillare la sua stima attorno al valore reale. In realtà vedremo che tenderà a stabilizzarsi sul valore più vicino al vero ritardo, ovvero:

$$\hat{\tau}_0 = \tau_0 + \text{round}(\tau_e) \tag{5.2.10}$$

Dove τ_e è rappresentato in frazioni di subframe cioè $\delta\tau_e = 1/40$.

Testando l'algoritmo di rilevazione d'eco, supponendo nullo il ritardo della rete, il comportamento medio, in diversi casi di SNR ed ERL, a seconda del ritardo τ_e al terminale radiomobili è mostrato in figura 5.13.

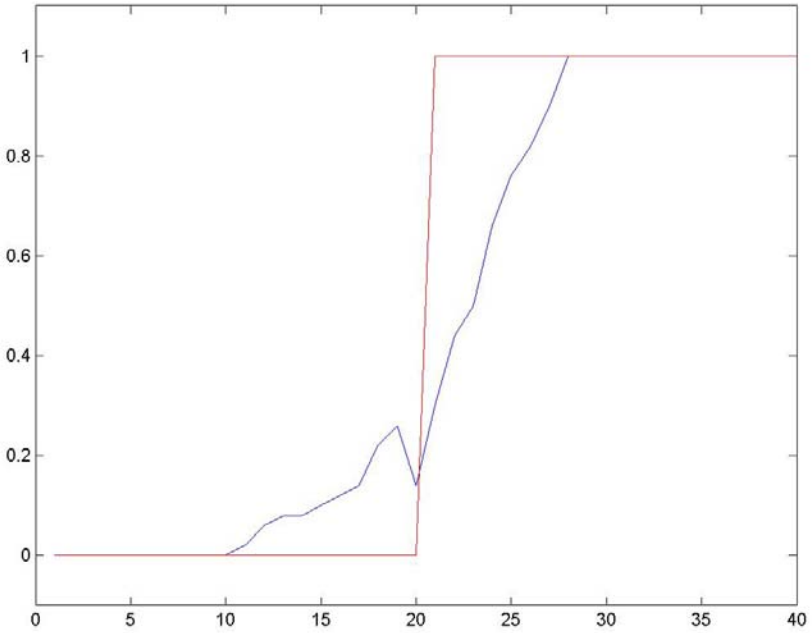


Figura 5.13 Comportamento medio del rilevatore d'eco con un ritardo $0 \le \tau_e \le 5ms \cdot 8000Hz$, ideale (blu), reale (rosso)

La misura correlativa inoltre tenderà ad avere una dispersione più ampia, una conseguenza di questo è anche la non immediatezza del fronte di risalita del rilevatore d'eco. In figura 5.14 è mostrato un esempio del comportamento della misura di cross-covarianza calcolata dal rilevatore d'eco per $\tau_e = 0ms$ (blu), $\tau_e = 1ms$ (verde), $\tau_e = 2ms$ (rosso), $\tau_e = 3ms$ (magenta), $\tau_e = 4ms$ (nero), $\tau_e = 5ms$ (azzurro): in questo caso il ritardo fisso introdotto dalla rete è pari a $120\,ms$, ovvero 24 subframes.

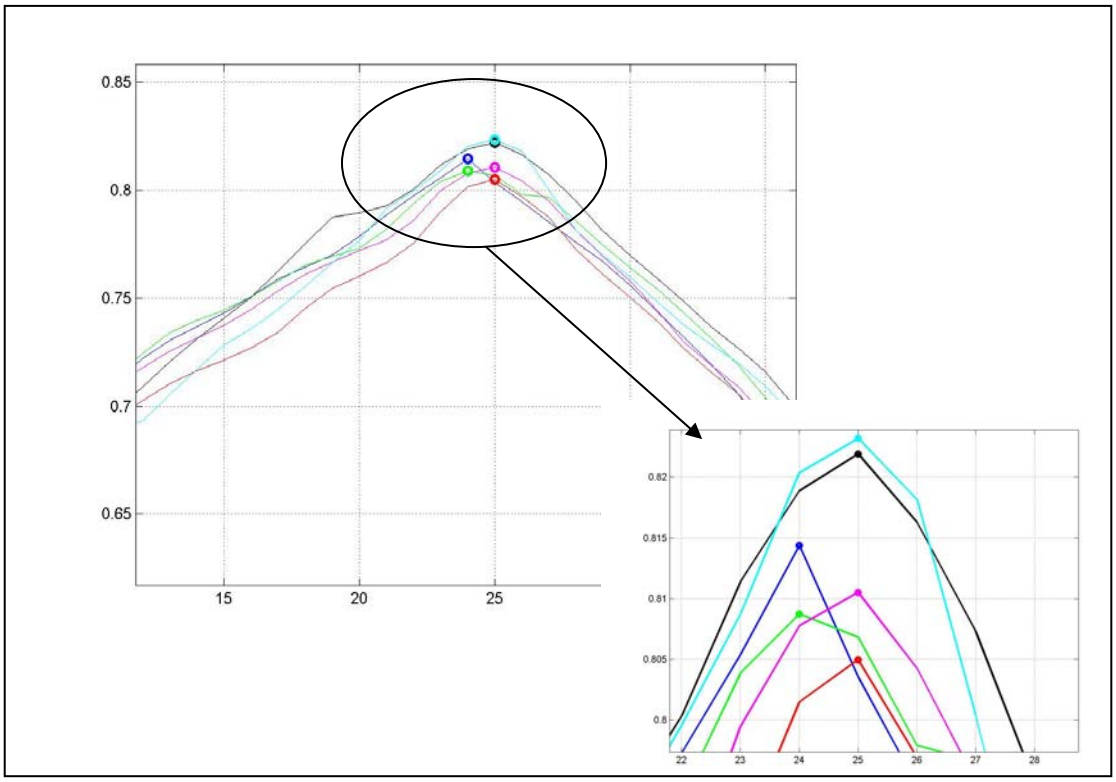


Figura 5.14 Comportamento del rilevatore d'eco con un ritardo di rete pari a $\tau_0 = 120ms$ e ritardi $\tau_e = 0ms$ (blu), $\tau_e = 1ms$ (verde), $\tau_e = 2ms$ (rosso),, $\tau_e = 3ms$ (magenta), $\tau_e = 4ms$ (nero), $\tau_e = 5ms$ (azzurro); $ERL = 20dB$, $SNR = 20dB$.

Ovviamente queste considerazioni sono dipendenti sia dal echo return loss, sia dal rapporto segnale rumore, sia dal tipo di rumore considerato. Infatti, la deviazione standard della funzione di cross-covarianza tendendo ad aumentare, come visto nelle prestazioni del rilevatore d'eco, porterà errori, eventualmente a livello di subframe però, quindi difficilmente relazionabili con il ritardo aggiunto τ_e. È poi evidente che le considerazioni fatte riguarderanno anche l'inseguitore d'eco.

5.3 Algoritmi di cancellazione d'eco

5.3.1 Modifica del guadagno di codebook algebrico

Il guadagno di codebook algebrico $g_{fixed}(n)$, come visto in precedenza, può essere considerato come un fattore moltiplicativo applicato alla funzione di trasferimento dell'n-esimo subframe analizzato:

$$H(z,n) = \frac{g_{fixed}(n)}{\left(1 - g_{pitch}(n) \cdot z^{-T(n)}\right)\left(1 + \sum_{i=1}^{10} a_i(n) \cdot z^{-i}\right)} \tag{5.3.1}$$

Risulta quindi opportuno realizzare un algoritmo robusto in grado di modificarlo nei subframe in cui l'eco è presente, con conseguente riduzione dell'energia del segnale indesiderato. A questo proposito si è scelto di usare, sulla falsariga dei metodi di *echo control* nel dominio lineare [2], l'algoritmo Least Mean Square o algoritmo del gradiente stocastico [54].

L'idea alla base di questo tipo di algoritmo è piuttosto semplice. Supponiamo che $g_y(n)$ sia il guadagno di codebook proveniente dal microfono del near-end, possiamo approssimare:

$$g_y(n) = f\left(g_e(n), g_v(n), g_{bn}(n)\right) \approx g_e(n) + g_v(n) + g_{bn}(n) \tag{5.3.2}$$

Dove $g_e(n)$ rappresenta il guadagno relativo al segnale d'eco, $g_{bn}(n)$ rappresenta il guadagno dovuto alla presenza del rumore di fondo e $g_v(n)$ rappresenta il guadagno dovuto alla possibile presenza di voce dal near-end. La validità dell'approssimazione è esemplificata nella figura 5.15.

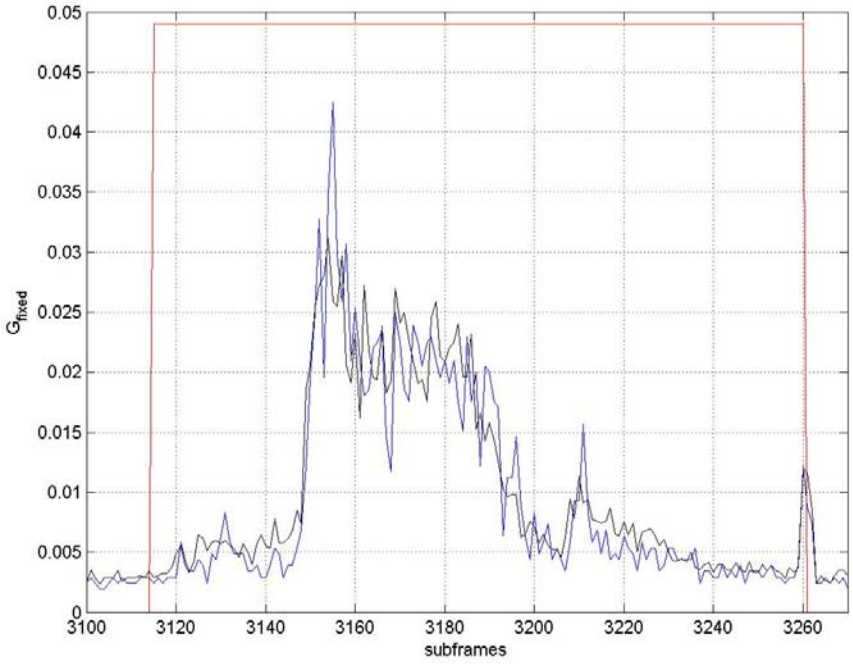


Figura 5.15 Andamento del guadagno di codebook algebrico reale $g_y(n)$ (blu) e ideale (nero) come somma dei guadagni relativi ai vari contributi $g_y(n) = g_s(n) + g_e(n) + g_{bn}(n)$. Misurazione effettuata con $ERL = 10dB$, $SNR = 25dB$ (rumore AWGN)

Supporremo inoltre che sia possibile rappresentare $g_e(n)$ come combinazione lineare di $g_x(n)$:

$$g_e(n) = g_x(n) \otimes \underline{h} = \sum_{l=0}^{L-1} g_x(n) h_l = \underline{h}^T \underline{g}_x(n) \tag{5.3.3}$$

Questo non sarà strettamente vero a causa della presenza di rumore e della numerose distorsioni. Tuttavia il modello risulterà comunque robusto.

Supponiamo quindi di trovare una stima dell'eco $\hat{g}_e(n)$ tramite un filtro $\underline{\hat{h}}$ che si avvicini al filtro $\underline{h}$:

$$\hat{g}_e(n) = \hat{h}^T \underline{g}_x(n) \tag{5.3.4}$$

Quello che vogliamo è quindi implementare un algoritmo che porti a migliorare $\underline{\hat{h}}$ per renderlo sempre più vicino al vettore $\underline{h}$. Essendo $\underline{h}$ sconosciuto, dobbiamo valutare la bontà di $\underline{\hat{h}}$ in maniera indiretta. Il criterio scelto negli algoritmi LMS per valutare l'efficacia rappresentativa di $\underline{\hat{h}}$ è quella di misurare l'errore quadratico medio che intercorre tra il segnale sintetizzato e il segnale reale:

$$E\left[e^2(n)\right] = E\left[\left(g_y(n) - \hat{g}_e(n)\right)^2\right] \tag{5.3.5}$$

Essendo l'obbiettivo l'approssimazione di $\underline{h}$ con $\underline{\hat{h}}$, il modo più naturale è quello di muoversi in direzione opposta al gradiente di $E\left[e^2(n)\right]$. In questo modo si potrebbe pensare di modificare l'algoritmo incrementando $\underline{\hat{h}}$ con il valore:

$$\Delta\hat{h} = -\frac{\mu}{2}\nabla_{\underline{\hat{h}}} E\left[e^2(n)\right] = -\frac{\mu}{2}\nabla_{\underline{\hat{h}}} E\left[\left(g_y(n) - \hat{g}_e(n)\right)^2\right] \tag{5.3.6}$$

Dove μ viene detto *step-size* ed è il parametro che controlla la velocità di adattamento. Il controllo di μ non è affatto banale, e molti studi sono stati realizzati in merito [34].

Quello che l'algoritmo del gradiente stocastico opera è una sostituzione del valore atteso dell'errore quadratico con il suo valore *istantaneo*. Sotto certe condizioni tuttavia anche una stima così "grezza" risulta avere risultati efficaci.

Il gradiente stocastico viene applicato nell'equazione (5.3.6):

$$\Delta\hat{h} = -\frac{\mu}{2}\nabla_{\underline{\hat{h}}} E\left[e^2(n)\right] = -\mu e(n)\nabla_{\underline{\hat{h}}} E\left[e(n)\right] = -\mu e(n)\underline{g}_x(n) \tag{5.3.7}$$

Perciò, al subframe n-esimo, l'aggiornamento dei pesi del filtro adattativo verrà fatto in questo modo:

$$\underline{\hat{h}}(n+1) = \underline{\hat{h}}(n) + \mu\underline{g}_x(n)e(n) \tag{5.3.8}$$

La teoria suggerisce [13] che per un segnale stazionario gaussiano, come è il guadagno di codebook algebrico, l'unica condizione per la stabilità è che:

$$0 < \mu < \frac{2}{\mathrm{tr}\left(\underline{\underline{R}}_x\right)} \tag{5.3.9}$$

Dove al denominatore si trova la traccia della matrice di autocorrelazione del processo osservato $g_x(n)$.

La velocità di convergenza risulta difficile da stimare, tuttavia risulta chiaro che a seconda della struttura della matrice questa può variare molto se gli autovalori di $g_x(n)$ sono compresi in un ampio intervallo di valori. A questo proposito quello che idealmente si vuole fare è "sbiancare" il segnale in modo da rendere tutti gli autovalori uguali e costanti. Il guadagno di codebook, essendo non stazionario, in quanto segue in maniera lineare le variazioni di energia della voce, ha un comportamento spettrale estremamente variabile che rende difficile l'adattamento dello sbiancamento: questa strada risulta quindi difficile e difficilmente applicabile [6]. Tuttavia, una prima "pulizia" della variabilità degli autovalori può essere eliminata facilmente: la variabilità dovuta ai cambiamenti del livello del segnale.

Essendo gli autovalori proporzionali alla varianza (o potenza) del segnale in ingresso, l'obbiettivo può essere conseguito dividendo la parte destra dell'equazione (5.3.7) per la stima locale della potenza. Così, la (5.3.8) diventa:

$$\underline{\hat{h}}(n+1) = \underline{\hat{h}}(n) + \gamma \frac{e(n)}{\underline{g}_x^{\,T}(n)\,\underline{g}_x(n)}\,\underline{g}_x(n) \tag{5.3.10}$$

dove adesso γ è lo step-size normalizzato. E' questo il famoso algoritmo *Normalized Least Mean Square* (NLMS) [19] creato per ovviare ai problemi di processi stazionari in cui il valore dell'errore perde di significato in quanto dipende fortemente dal livello di potenza del segnale analizzato. La regola per la convergenza diventa ora [6]:

$$\gamma < 2 \tag{5.3.11}$$

Una volta affrontata la teoria riguardante l'implementazione dell'algoritmo, giungiamo ad un problema più pratico ma decisamente importante, il numero di prese necessario al filtro $\underline{\hat{h}}$. L'argomento necessita qualche digressione teorica, infatti, la corrispondenza dei segnali usati per l'analisi, mostrata nell'equazione (5.3.3) è solo un'approssimazione: troppe sono le non-linearità e le manipolazioni che il codec compie sul segnale. Questo spingerà l'algoritmo a modellizzare la risposta impulsiva del sistema solo in maniera parziale, ma sufficiente ad avere una riduzione dell'energia del segnale di disturbo. Il numero di *taps* solitamente viene scelto cercando un trade-off tra velocità di convergenza (inversamente proporzionale alla lunghezza di $\underline{\hat{h}}$) e prestazioni a regime (solitamente maggiori con filtri adattativi lunghi); nel nostro caso invece si cercherà di avere un valore tale da avere una varianza dell'errore a regime più bassa possibile.

Nell'algoritmo implementato si è scelta una lunghezza del filtro adattativo $\underline{\hat{h}}$ pari a 5 prese, in modo da poter modellizzare percorsi di eco di lunghezza fino a $25\,ms$, sufficienti ad affrontare il problema in analisi. In linea teorica, sarebbe necessaria solo una presa in quanto il rilevatore ed inseguitore d'eco stimano lo sfasamento e la risposta dell'ambiente è supposta piatta. Tuttavia, a causa della varianza del guadagno, una sola presa può portare ad errori di stima locali del sistema. Inoltre, esiste una dipendenza, data dal modo in cui il segnale viene codificato (lavorando su frames composti da quattro subframes), tra campioni di $g_{fixed}(n)$ adiacenti, della quale con $N = 5$, si riesce a tener conto. Inoltre, con questa lunghezza, si riesce anche a sopperire ai piccoli errori di temporizzazione che possono verificarsi in condizioni di *SNR* ed *ERL* non buone.

Per $N > 5$, il problema riguarda la scarsa abilità del sistema ad adattarsi alle veloci transizioni del $g_{fixed}(n)$, che portano, in casi particolarmente sfavorevoli, anche ad amplificare il segnale d'uscita. In figura 5.16 è mostrata la *system distance* calcolata come:

$$SysDist = 10\log_{10}\left[\left(g_y(n) - \hat{g}_e(n)\right)^2\right] \tag{5.3.12}$$

Nella valutazione degli algoritmi di cancellazione d'eco, questo parametro mostra l'efficienza di questi misurando la differenza che intercorre tra segnale modellizzato e segnale reale.
Si noti come per $N = 5$, la varianza dell'errore sia più bassa che negli altri casi.

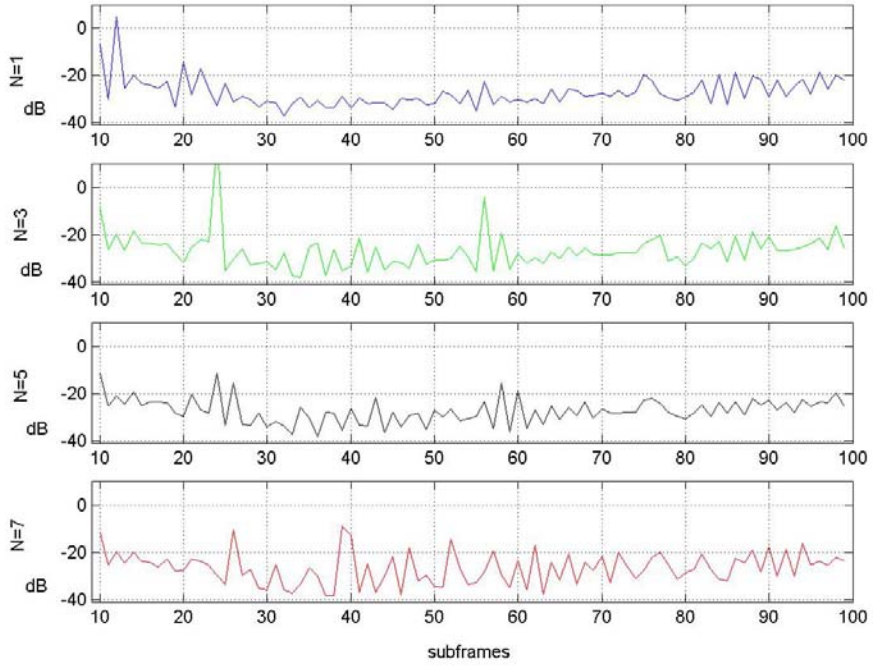


Figura 5.16 Andamento della *system distance* con lunghezza del filtro adattativo pari a $N = 1$, $N = 3$, $N = 5$, $N = 7$. Misurazione effettuata con $ERL = 10dB$, $SNR = 20dB$ (rumore AWGN).

A riprova di quanto detto in precedenza, in figura 5.17 viene mostrato il comportamento della *system distance* con algoritmo NLMS a 1, 5 e 9 prese.

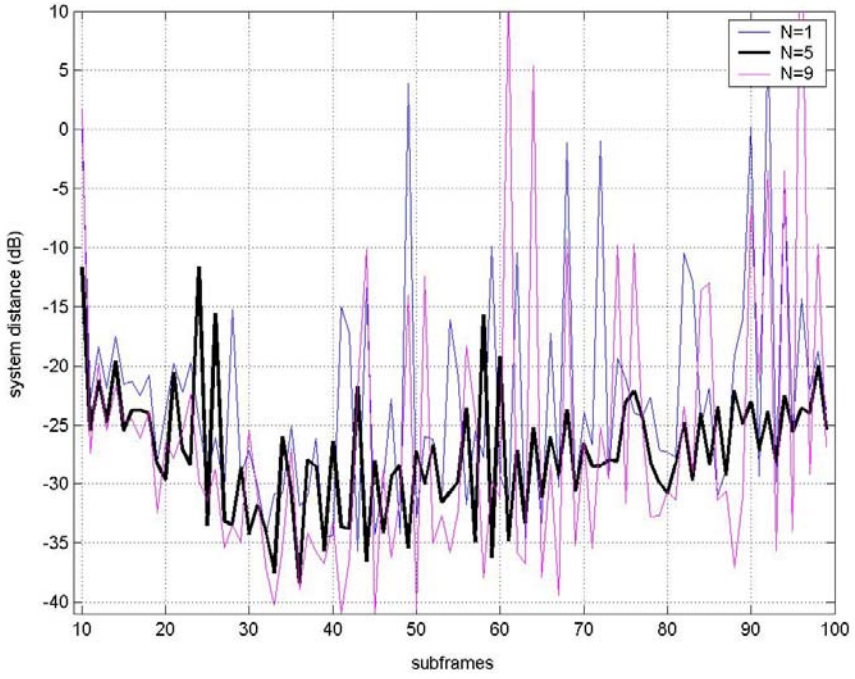


Figura 5.17 Andamento della *system distance* con lunghezza del filtro adattativo pari a $N=1$, $N=5$, $N=9$. Misurazione effettuata con $ERL=10dB$, $SNR=20dB$ (rumore AWGN).

Inoltre, come accennato in precedenza, in presenza di eco, l'inseguitore implementato offrirà in uscita anche un valore di echo likelihood tra i segnali analizzati $cc(n)$ (equazione (5.2.7)), questo consente di poter regolare la velocità di adattamento e convergenza a seconda di quanto i segnali si assomiglino. Ricordando che $0<cc(n)\leq 1$ e ponendo $\gamma=1.5$ così da garantire la stabilità, il criterio di aggiornamento del filtro diventa:

$$\underline{\hat{h}}(n+1)=\underline{\hat{h}}(n)+1.5\cdot cc(n)\frac{\left(g_y(n)-\hat{g}_y(n)\right)}{\underline{g}_x{}^T(n)\,\underline{g}_x(n)}\underline{g}_x(n) \tag{5.3.13}$$

e quindi, il segnale in uscita $g_u(n)$ sarà:

$$g_u(n)=g_y(n)-\hat{g}_y(n)=g_y(n)-\underline{\hat{h}}^T(n)\,\underline{g}_x(n) \tag{5.3.14}$$

Come abbiamo visto in precedenza, per $cc(n)<0.42$ (soglia usata per definire lo stato di double-talk) l'algoritmo si ferma, in questo caso $g_u(n)=g_y(n)$; inoltre, si bloccherà anche l'aggiornamento del filtro.

In figura 5.18 sono mostrate le prestazioni in termini di rapporto tra segnale in ingresso al sistema e in uscita. La misura di riferimento in ascissa è il rapporto tra il guadagno del far-end e il guadagno dovuto al suo eco: molto simile al *Echo Return Loss*. La misura delle prestazioni in termini di *SNR*, sono il risultato medio con diversi tipi di rumore di sottofondo.

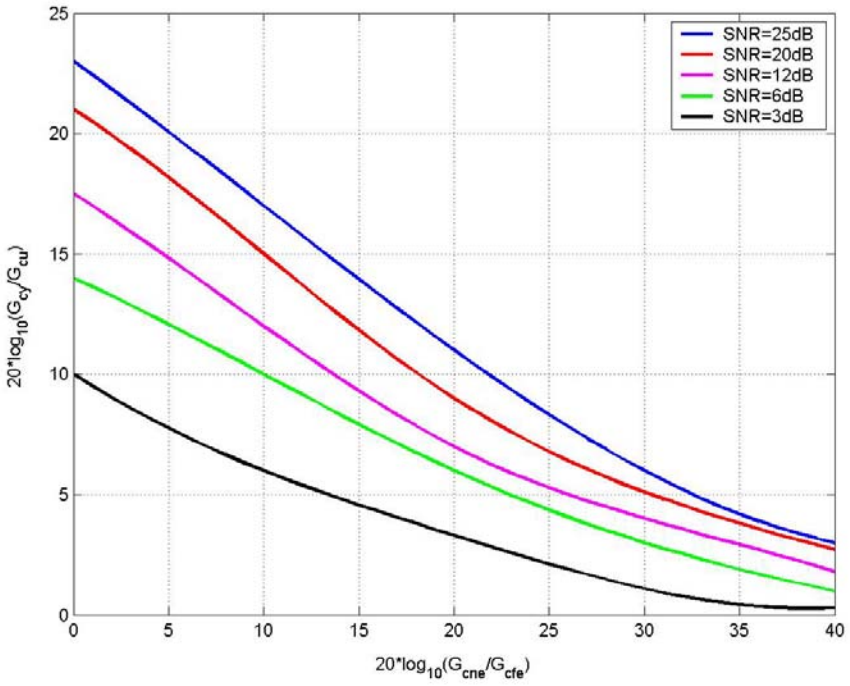


Figura 5.18 Prestazioni dell'algoritmo NLMS sul guadagno di codebook algebrico.

La rapida decadenza delle prestazioni è dovuta principalmente alla sempre più scarsa somiglianza del guadagno di far-end con la sua versione distorta al near-end, al crescere del ERL e del SNR, ma anche al basso valore di $cc(n)$, che porta a una minore cancellazione. Tuttavia, a livello psicoacustico, dove il cancellatore non offre buone prestazioni è anche dove l'eco da meno fastidio: verosimilmente quindi il MOS non avrà variazioni così significative.

5.3.2 Modifica del guadagno di codebook adattativo

Il guadagno di codebook adattativo o guadagno di pitch $g_{pitch}(n)$, come si nota dall'equazione (5.3.1) non influenza il livello energetico del segnale in uscita pesantemente come il guadagno di codebook algebrico ma, come si è visto nel terzo capitolo, mantiene una media piuttosto alta nei segmenti di parlato e quindi anche in presenza di eco. Questo fatto ci fa ritenere importante applicare degli algoritmi di controllo anche su di esso. La scelta cade subito sull'algoritmo Normalized Least Mean Square già implementato per il guadagno di codebook adattativo. Anche in questo caso si è scelta una lunghezza del filtro adattativo $\underline{\hat{h}}$ pari a 5 prese, per gli stessi motivi citati precedentemente. L'algoritmo viene implementato allo stesso modo del precedente, quindi, l'aggiornamento del filtro utilizzerà anch'esso il

valore di echo likelihood tra i segnali analizzati $cc(n)$ (equazione (5.2.7)) che consente di poter regolare la velocità di adattamento e convergenza a seconda di quanto i segnali si assomiglino. Il criterio di aggiornamento del filtro sarà ancora pari a:

$$\underline{\hat{f}}(n+1) = \underline{\hat{f}}(n) + 1.5 \cdot cc(n) \frac{\left(g_{py}(n) - \hat{g}_{py}(n)\right)}{\underline{g}_{px}{}^{T}(n)\,\underline{g}_{px}(n)} \underline{g}_{px}(n) \tag{5.3.15}$$

Dove $g_{px}(n)$ è il segnale proveniente dal far-end e $g_{py}(n)$ è il segnale proveniente dal near-end. Il segnale in uscita $g_{pu}(n)$ sarà:

$$g_{pu}(n) = g_{py}(n) - \hat{g}_{py}(n) = g_{py}(n) - \underline{\hat{f}}^{T}(n)\,\underline{g}_{px}(n) \tag{5.3.16}$$

Come abbiamo visto in precedenza, per $cc(n) < 0.42$, soglia usata per definire lo stato di double-talk, l'algoritmo si ferma e in questo caso avremo $g_{pu}(n) = g_{py}(n)$; inoltre, si bloccherà anche l'aggiornamento del filtro.
Le prestazioni dell'algoritmo, sono estremamente interessanti, infatti, come mostrato in figura 5.19, il guadagno si manterrà piuttosto alto anche in condizioni non buone di eco e di rumore.

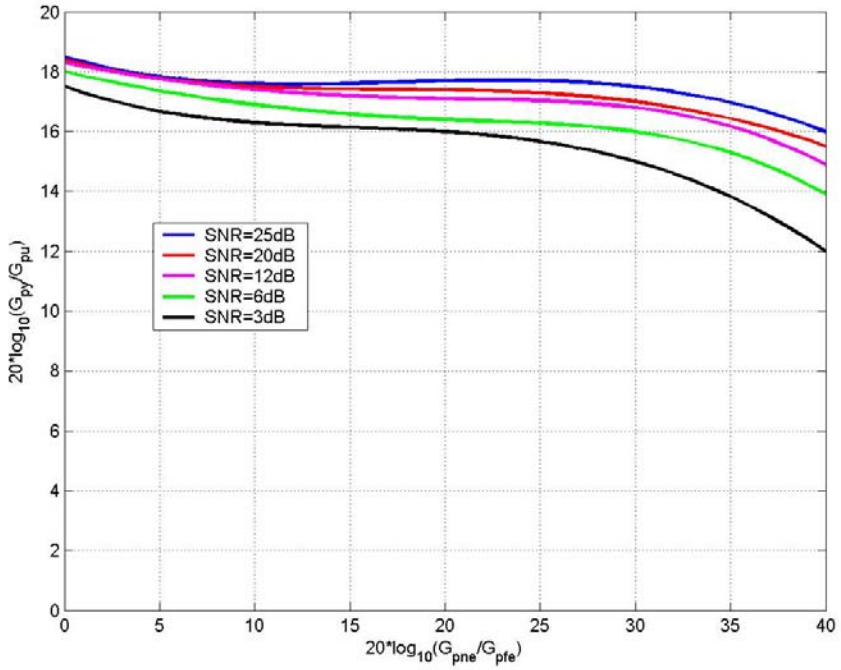


Figura 5.19 Prestazioni dell'algoritmo NLMS sul guadagno di codebook adattativo.

Questo è dovuto principalmente alla statistica del guadagno di pitch, come abbiamo visto al terzo capitolo; esso tenderà ad avere valori che si aggirano attorno al suo massimo in condizioni di voce (anche rumorosa), mentre sarà molto più basso in condizioni di solo

rumore. In figura 5.20 è mostrato un confronto tra NLMS sul guadagno di pitch e NLMS sul guadagno di codebook algebrico. Si noti che la statistica del primo permette una cancellazione massiccia, mentre la forte varianza del secondo, porta a un "limite" di cancellazione oltre il quale sarà difficile scendere.

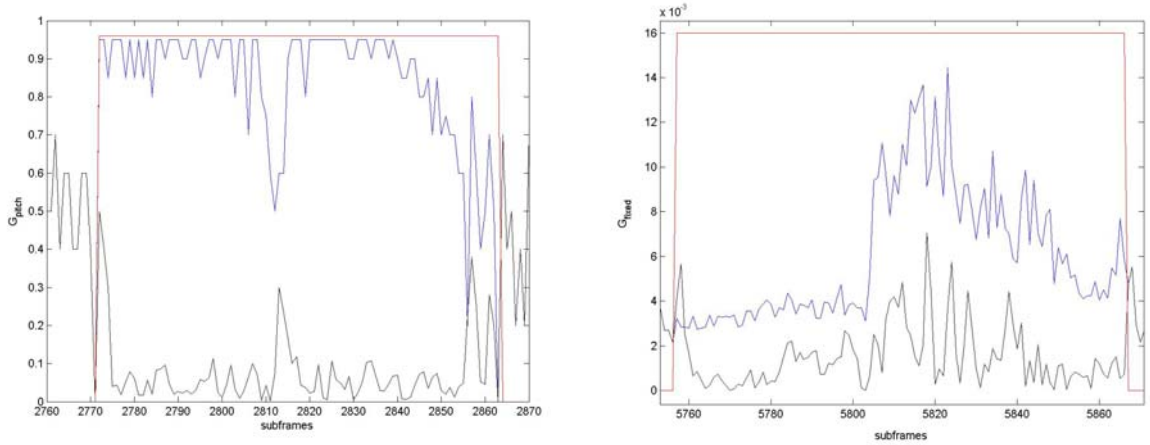


Figura 5.20 Confronto tra funzionamento degli algoritmi NLMS sul g_{pitch} (sinistra) e g_{fixed} (destra); in blu è mostrato l'ingresso ed in nero l'uscita. Misurazione effettuata con $ERL = 20dB$, $SNR = 20dB$ (rumore AWGN).

5.3.3 Modifica del tempo di pitch

Nel contesto in cui questo lavoro di tesi è stato svolto, una corrispondenza perfetta (in qualsiasi tipo di tempo e situazione) del filtro cancellatore d'eco al sistema non è chiaramente possibile, come si è spiegato precedentemente, troppi sono i fattori distortivi del segnale. Questo comporterà la presenza, anche se abbondantemente ridotta, di eco nel segnale errore che torna al parlatore posto al far-end. Il principale motivo è quello che la caratterizzazione spettrale del segmento temporale (subframe) del segnale in uscita dagli algoritmi NLMS applicato ai due guadagni è rimasta praticamente intatta. A questo proposito, si è pensato di andare a modificare il tempo di pitch e le posizioni delle linee spettrali di frequenza: in questa sezione ci occuperemo della modifica del tempo di pitch.

Dall'analisi statistica effettuata sul tempo di pitch nel terzo capitolo si è stabilito che l'andamento del tempo di pitch in presenza di parlato (soprattutto nei suoni vocalici) mantiene un andamento gaussiano con un valor medio ben definito e una varianza molto bassa. Per contro, in presenza di diversi tipi di rumore l'andamento non presenta particolari caratteristiche statistiche e quindi è sembrato giusto considerare la sua densità di probabilità come uniforme.

L'algoritmo implementato quindi, in presenza di eco, cercherà di eliminare l'informazione di lungo termine basata sulle armoniche presenti nel parlato. A questo proposito si cercherà di rendere casuale il tempo di pitch. Viene ancora utilizzata l'informazione proveniente dall'inseguitore d'eco (equazione (5.2.7)) sul grado di likelihood tra x e y (segnale al far-end e segnale d'eco). Infatti se, una volta allineati gli assi temporali, siamo nel caso $VAD_x\left(n-\hat{\tau}_0\right)=1 \quad \wedge \quad VAD_y\left(n\right)=1 \quad \wedge \quad cc\left(n\right)>0.5$ allora:

$$T_{pitch,u}\left(i\right)=T_r \tag{5.3.17}$$

dove T_r è un valore preso da uno spazio di probabilità uniforme $\Omega_{T_r}=\{17,18,19,....,142,143\}$, lo stesso dominio in cui si muove la parte intera del tempo di pitch. Il valore frazionario viene trascurato.

Si noti che la scelta di $cc\left(n\right)>0.5$ non è banale, infatti questo algoritmo è piuttosto potente e quindi nel caso di double talk, dove comunque i suoni vocalici sono presenti, si otterrebbe un fenomeno piuttosto fastidioso di interruzione dell'intelliggibilità del parlato creando un suono particolarmente "metallico"; a questo proposito si è preferito alzare la soglia di non-funzionamento del double-talk detector. Quindi per $cc\left(n\right)<0.5$ si preferisce non usare l'algoritmo. Un esempio della modifica effettuata è mostrato in figura 5.21.

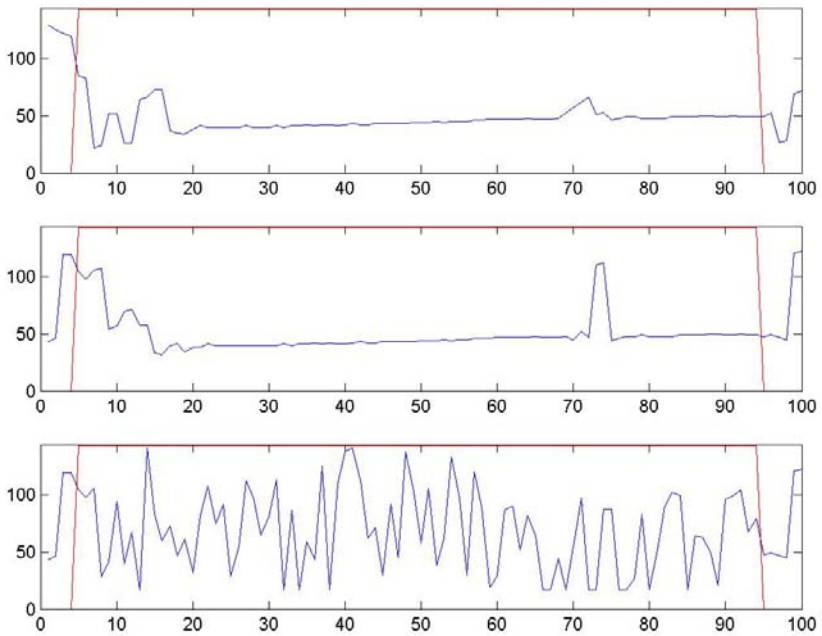


Figura 5.21 Modifica del tempo di pitch. Dall'alto sono rappresentati $T_x\left(n\right)$ e $T_y\left(n\right)$, in basso è mostrato il segnale all'uscita del "randomizzatore" $T_u\left(n\right)$

Risulta difficile misurare le prestazioni di questo algoritmo in modo numerico, come effettuato per le modifiche sui guadagni. In realtà la modifica energetica esiste, seppur minima, ma è praticamente irrilevante. La manipolazione effettuata è a livello psico-acustico e quindi difficilmente quantificabile con parametri oggettivi.

5.3.4 Modifica delle linee spettrali di frequenza

Dopo aver eliminato in parte, con la modifica dei parametri di predizione a lungo termine, la caratterizzazione spettrale della funzione di trasferimento relativa ad un subframe, ci occuperemo nel seguito della modifica della predizione a breve termine ovvero del filtro LPC di ordine 10:

$$H(z,n) = \frac{1}{1+\sum_{i=1}^{10} a_i(n)\cdot z^{-i}} \tag{5.3.18}$$

Come visto in precedenza, i parametri del filtro LPC vengono trasformati dal codec per motivi di robustezza alla quantizzazione in linee spettrali di frequenza LSF. L'algoritmo implementato si propone quindi di andare a toccare le linee spettrali di frequenza, cercando di eliminare il più possibile le informazioni di tipo formantico (i picchi), ovvero di sbiancare il segnale.

Questa operazione risulta particolarmente semplice con il comportamento spettrale del segnale rappresentato dalle linee spettrali di frequenza, in quanto, come dimostrato nel terzo capitolo, in presenza di rumore gaussiano bianco, anche il comportamento delle linee spettrali di frequenza sarà gaussiano. Abbiamo inoltre dimostrato che gli LSF sono tra loro equidistanti in media, ovvero che ciascun LSF presenta un valore medio per l'$i-$esima ($i=1,...,p$) linea spettrale pari a $i\cdot\left(\pi/\left(p+1\right)\right)$, dove p è l'ordine di predizione e anche il numero di LSF, quindi con $p=10$, il vettore dei valori medi sarà:

$$\underline{L}_w = \left[l_{1,w}, l_{2,w}, ..., l_{10,w}\right] = \left[\frac{\pi}{11}, \frac{2\pi}{11}, \frac{3\pi}{11}, \frac{4\pi}{11}, \frac{5\pi}{11}, \frac{6\pi}{11}, \frac{7\pi}{11}, \frac{8\pi}{11}, \frac{9\pi}{11}, \frac{10\pi}{11}\right] \tag{5.3.19}$$

Le ipotesi, rigorosamente dimostrate, ci permettono di affermare che per sbiancare il segnale sarà sufficiente cambiare al valore presente dell'$i-$esima linea spettrale d'eco il suo valore medio nel caso gaussiano, certi che questa sarà la migliore scelta possibile.

Tuttavia, come già in precedenza si diceva per il tempo di pitch, nel caso in cui fossimo in presenza di double talk, sarà necessario impedire all'algoritmo di agire: cancellare

l'informazione formantica di parlato utile sarebbe un fenomeno estremamente fastidioso. A questo proposito si è deciso anche in questo caso di utilizzare il parametro di echo likelihood $cc(n)$ proveniente dall'inseguitore d'eco. Questo permette di addolcire lo sbiancamento del segnale nel caso di presenza di parlato sovrapposto. In particolare, invece di sostituire il valore dell'$i-$esima linea spettrale $l_i(n)$ del subframe $n-$esimo con quello corrispondente nell'equazione (5.3.19), si media tra questo valore e quello attuale:

$$\hat{l}_{u,i}(n) = cc(n) \cdot \frac{i \cdot \pi}{11} + (1 - cc(n)) \cdot l_i(n) \tag{5.3.20}$$

Operando invece direttamente sul vettore $\{a_k\}$, l'operazione appena accennata diventa molto simile a quella di sostituire i valori LPC dell'$n-$esimo subframe $\{a_i(n)\}$ con $\{(1 - cc(n))^i \cdot a_i(n)\}$, diventando:

$$H_u(z,n) = \frac{1}{1 + \sum_{i=1}^{10} (1 - cc(n))^i \cdot a_i(n) \cdot z^{-i}} \tag{5.3.21}$$

dove $(1 - cc(n))$ può essere visto come un fattore di "metamorfosi" del filtro originale. Infatti per $cc(n) = 1$, $H_u(z,n)$ diventa un filtro passatutto, mente per $cc(n) = 0$, $H_u(z,n) = H(z,n)$ rimane del tutto invariato. Si noti che la stabilità del sistema viene mantenuta in entrambe le trasformazioni. In realtà poi la trasformazione non avviene se $cc(n) < 0.42$, valore stabilito dal double-talk detector. Un esempio del funzionamento dell'algoritmo è mostrato in figura 5.22.

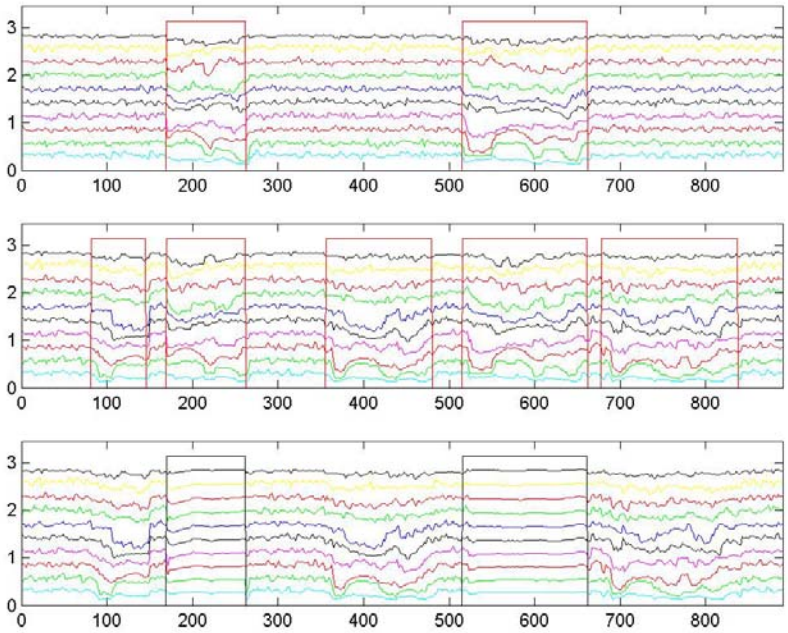


Figura 5.22 Modifica delle linee spettrali di frequenza. Dall'alto sono rappresentate le dieci linee spettrali di frequenza $l_i(n)$ di $x(t)$, le $l_i(n)$ di $y(t)$, in basso, evidenziata in nero, è mostrata l'uscita dall'algoritmo.

Nel caso delle trasmissioni vocali radiomobili, la maggior parte delle conversazioni avvengono in ambienti dove il tipo di rumore di fondo o *background noise* risulta tutt'altro che bianco. A questo proposito si potrebbe modificare la regola di modifica delle posizioni degli LSF, cambiando il vettore di riferimento del rumore gaussiano bianco dell'equazione (5.3.18). Perciò si è deciso di calcolare il vettore in maniera "adattativa", infatti nei momenti in cui $VAD_y(n)=0$, si raccoglieranno le statistiche relative al rumore di fondo, queste saranno poi utilizzate per calcolare il vettore con il comportamento *medio* delle linee spettrali di frequenza $\underline{L}_{bn}$. È importante notare, come visto nello studio statistico, che se si prende un intervallo temporale troppo lungo, qualsiasi sia il rumore di sottofondo avremo $E[\underline{L}_{bn}]=E[\underline{L}_w]$ quindi sarà necessario prendere finestre temporali di pochi subframes $(10\div15)$ e andare a prendere media e varianza su questi. In questo modo la comunicazione risulterà priva di fastidiosi cambi di background noise, ad esempio passando da rumore street a rumore gaussiano bianco.

Anche in questo algoritmo risulta difficile calcolare il miglioramento in termini oggettivi. Provando l'algoritmo in varie situazioni si è ottenuta un'attenuazione della potenza dell'eco di $2\div3\ dB$. Questo risultato di riduzione si affianca a un complessivo miglioramento psico-

acustico legato alla trasformazione in frequenza operata sull'eco residuo. Tale adattamento, come visto in precedenza, è stato realizzato mediante approssimazione delle caratteristiche dell'eco residuo al rumore di fondo precedentemente stimato.

5.4 Noise Injection

Il cancellatore d'eco, come si è visto in precedenza, offre prestazioni decisamente buone considerando l'eliminazione delle parti di segnale non volute dovute all'accoppiamento microfono-altoparlante. Tuttavia, un elevato livello di cancellazione potrebbe non corrispondere a una maggiore gradevolezza della conversazione dopo il passaggio per l'Acoustic Echo Canceller. Infatti, il cancellatore d'eco, in funzionamento a regime, crea alcune situazioni, specialmente a basso SNR, in cui si potrebbe verificare la fastidiosa sensazione di "linea caduta", ovvero una percezione di "quasi-silenzio" in cui l'interlocutore posto al far-end potrebbe pensare che la comunicazione si sia interrotta.
Un esempio di questo è mostrato in figura 5.23, dove il segnale residuo alla cancellazione, a livello energetico, è numerosi decibel sotto l'energia delle parti di segnale in assenza di parlato (ovvero di solo rumore).
A questo proposito, quello che si farà è ricreare rumore laddove l'eco è stato cancellato, ovvero una *Noise Injection* [45].

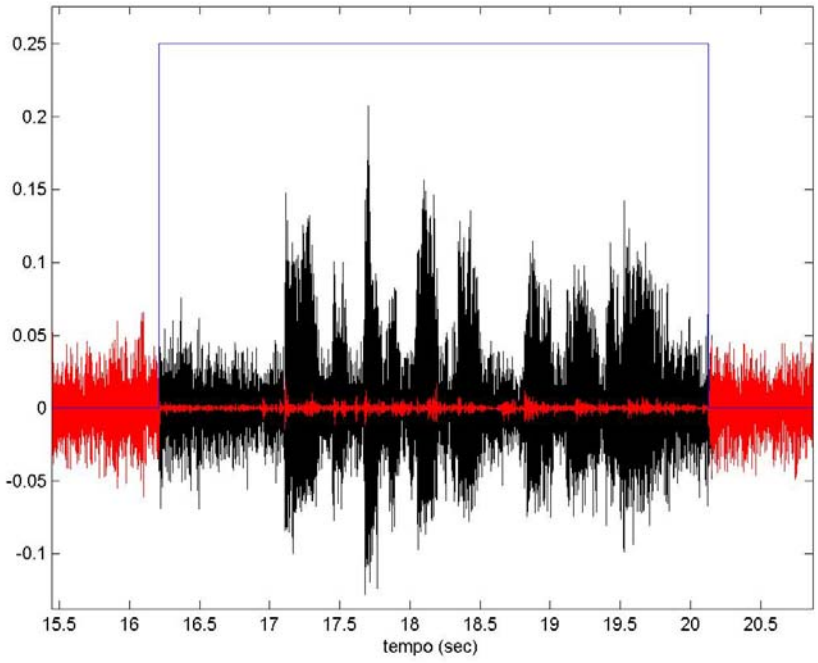


Figura 5.23 Segnale in ingresso al cancellatore d'eco (nero) e in uscita (rosso), in blu è mostrato dove esso agisce. Misurazione effettuata con $ERL = 20dB$, $SNR = 6dB$ (rumore babble) $ERLE = 17dB$.

Nella sezione precedente abbiamo già fatto in modo di ricreare, nelle parti dove agisce l'AEC, una caratteristica spettrale simile al rumore, avendo cambiato la posizione delle linee spettrali di frequenza. Quindi, invece di sommare rumore al segnale in uscita dal cancellatore, sarà opportuno impedire all'AEC di "abbassare" i guadagni di codebook algebrico e di codebook adattativo sotto una certa soglia. Questa soglia sarà semplicemente il valore atteso della statistica del guadagno di pitch e del guadagno di codebook algebrico creata quando $VAD_y(k) = 0$. aggiornabile iterativamente, usando lo stesso stimatore della media creato per aggiornare le soglie di VAD:

$$\begin{aligned} \mu_{gc}(k) &= a_\mu \mu_{gc}(k-1) + \frac{1-a_\mu}{N} \sum_{n=k-N}^{k} g_c(n) \\ \mu_{gp}(k) &= a_\mu \mu_{gp}(k-1) + \frac{1-a_\mu}{N} \sum_{n=k-N}^{k} g_p(n) \end{aligned} \tag{5.5.1}$$

Nella formula (5.5.1) k rappresenta il subframe corrente, mentre il parametro a_μ controlla il tempo della risposta al gradino del filtro, esso determina quindi il tempo di convergenza del filtro. Questo vale $a_\mu = 1 - e^{-5/N_\mu}$ dove $N_\mu = 30$ rappresenta il numero di campioni della risposta al gradino del filtro, *trade-off* ottenuto tra velocità di convergenza e precisione della

stima. Ovviamente la stima si fermerà quando $VAD_y = 1$, ovvero in presenza di parlato al near-end.

A questo punto, se $g_{cu}(n) < \mu_{gc}(n)$, si porrà $g_{cu}(n) = \mu_{gc}(n)$; poi, se $g_{pu}(n) < \mu_{gp}(n)$, si porrà $g_{pu}(n) = \mu_{gp}(n)$. Il risultato, per lo stesso segmento vocale analizzato in figura 5.23, è mostrato in figura 5.24.

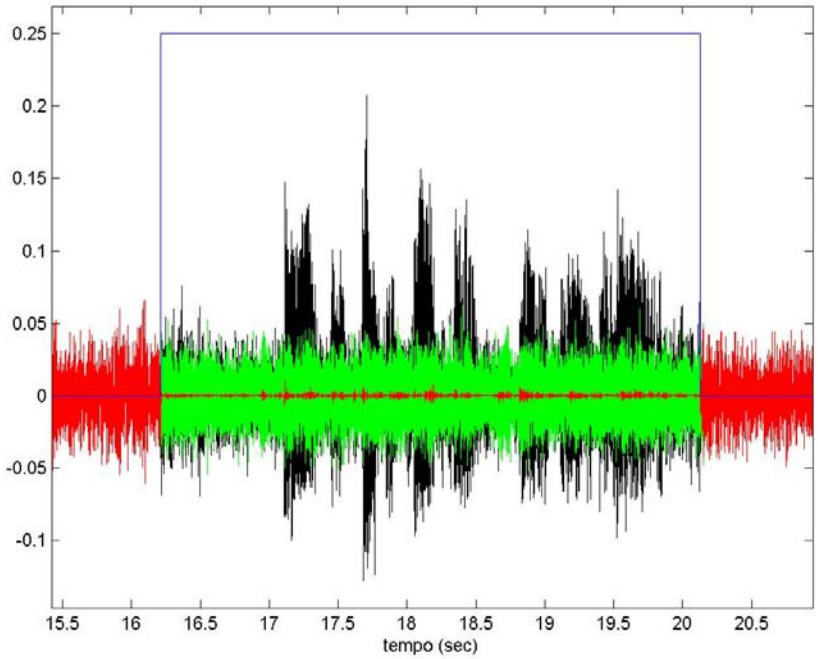


Figura 5.24 Segnale in ingresso al cancellatore d'eco (nero) e in uscita (rosso), in blu è mostrato dove esso agisce. In verde è mostrato il risultato con la *noise injection*. Misurazione effettuata con $ERL = 17dB$, $SNR = 6dB$ (rumore babble).

I risultati ottenuti per rumori stazionari sono soddisfacenti anche se non sono stati sottoposti a intensive sessioni di testing con numerosi ascoltatori, necessarie per validare le prestazioni a livello psico-acustico.

L'algoritmo perde le sue virtù in presenza di rumori fortemente non stazionari, infatti, le statistiche usate per stimare il comportamento del rumore non sarebbero più sufficienti a garantirne una indagine affidabile. Per semplicità, ci siamo limitati ad affrontare il problema in stazionarietà, tuttavia si possono immaginare un numero considerevole di possibili alternative più robuste per l'analisi. La soluzione potrebbe essere, ad esempio, quella di analizzare il movimento degli LSF e dei guadagni a livello temporale di numerosi segmenti di rumore per riprodurli durante l'azione di noise injection.

5.5 Prestazioni del cancellatore d'eco

La misura delle prestazioni totali del cancellatore d'eco implementato, sarà il passo finale del lavoro svolto in questa parte della tesi. Come misura delle prestazioni si è scelto il $ERLE$, Echo Return Loss Enhancement, parametro standard per definire la bontà di un algoritmo di AEC. Sono stati presi a questo proposito numerosi esempi di conversazioni telefoniche "ripuliti" artificialmente da rumori, dopodiché, il segnale proveniente dal far-end è stato combinato con $SNR = 20dB$ (rumore AWGN) fisso. Questo segnale, sommato a quello del near-end con vari valori di ERL $(0dB \div 40dB)$, è stato successivamente aggiunto di rumore di diversi tipi (car, street, wgn, babble, rain) con SNR variabile. I valori ottenuti sono stati poi mediati sui diversi tipi di rumore (con i quali si sono ottenuti prestazioni leggermente diverse tra loro). Il risultato ottenuto è mostrato in figura 5.25.

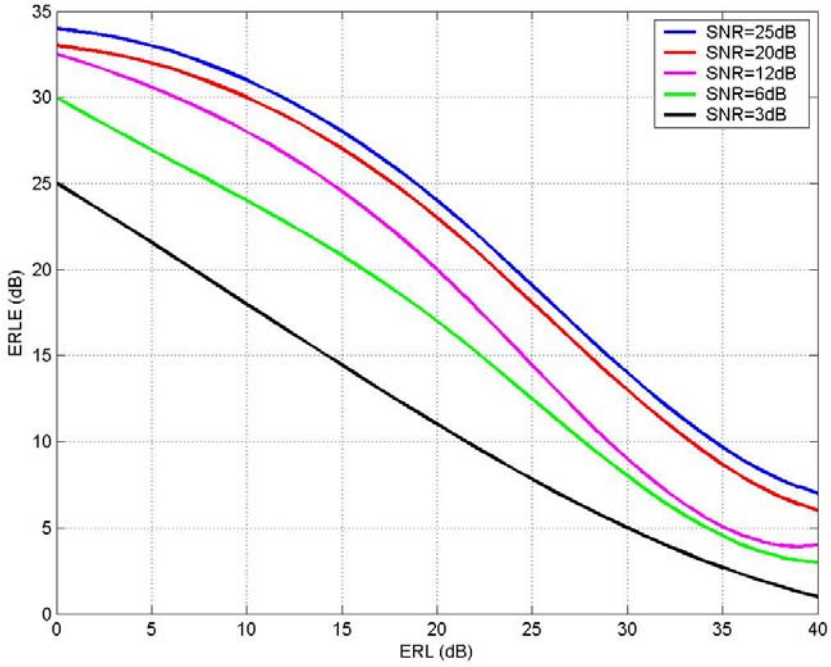


Figura 5.25 Prestazioni totali dell'algoritmo AEC in termini di $ERLE$ per valori diversi di ERL ed SNR.

I risultati ottenuti sono quindi particolarmente buoni, e del tutto paragonabili a quelli che si sarebbero ottenuti con cancellatori d'eco nel dominio lineare [ITU-T G.168].

5.6 Conclusioni

Alla luce dei risultati ottenuti, si può affermare che la cancellazione d'eco nel dominio codificato, pur essendo un campo ancora poco esplorato, presenta dei risultati promettenti.
A questo proposito, nei prossimi anni, è auspicabile una crescita di queste applicazioni, andando ad ottimizzare l'uso degli apparati di transcodifica usandoli solo per le comunicazioni da apparecchio mobile a fisso.
Il lavoro svolto ha mostrato una buona robustezza sia per quanto riguarda gli algoritmi atti alla cancellazione d'eco, applicati sui guadagni e sulle LSF, sia per quanto riguarda gli algoritmi accessori ad essa, come il rilevatore d'eco e il double-talk detector. I limiti principali degli algoritmi realizzati coincidono con quelli degli algoritmi nel dominio lineare [45], riguardano soprattutto i casi di *ERL* molto basso o di *SNR* molto basso, rafforzando la similarità dei cancellatori implementati con quelli canonici.

Conclusioni

Le tecniche di rilevazione dell'attività vocale e di cancellazione d'eco acustico applicate nel dominio codificato si sono dimostrate molto promettenti in termini di precisione ed affidabilità. Inoltre, il loro utilizzo si è dimostrato vantaggioso nell'ambito delle applicazioni nei sistemi di comunicazione per il notevole risparmio computazionale conseguito e la possibilità di operare su segnali compressi senza dover introdurre ulteriori transcodifiche del segnale audio.

Per quanto riguarda l'algoritmo di voice activity detection si è rivelata importante, per la sua affidabilità e robustezza, un'accurata fase di addestramento e di studio statistico dei parametri, in funzione delle particolari situazioni ambientali. Inoltre, i parametri che si sono mostrati particolarmente significativi per la rilevazione dell'attività vocale sono le linee spettrali di frequenza, le quali, comportandosi in modo ben distinto nei casi di voce e rumore, sono fondamentali discriminanti tra i due stati. La combinazione di questi parametri con il guadagno di codebook algebrico, lo studio del ritardo di pitch ed un algoritmo di stima delle caratteristiche del rumore, considerato non stazionario, ha permesso di raggiungere prestazioni soddisfacenti, fornendo, a 12 dB di SNR, una probabilità di corretta decisione del 95% ed una probabilità di errore totale (misdetection e false alarm) del 12%.

L'algoritmo di acoustic echo cancellation, basato sulla robustezza del VAD implementato, ha mostrato caratteristiche di funzionamento estremamente interessanti con prestazioni paragonabili agli algoritmi di cancellazione d'eco nel dominio lineare correntemente presenti nel mercato e sottoposti agli standard di qualità ITU-T. Queste prestazioni non sarebbe state possibili senza gli algoritmi accessori alla cancellazione d'eco quali il rilevatore e il double-talk detector, i quali hanno anch'essi dimostrato notevole robustezza anche in casi non propriamente ideali. Infatti, il rilevatore d'eco e la sua versione adattativa, l'inseguitore d'eco, hanno mostrato interessanti proprietà in termini di errore quadratico medio della stima del ritardo. Inoltre, l'uso dell'inseguitore d'eco, permette di avere in uscita un parametro di *echo likelihood* che comanda

direttamente il grado di cancellazione degli algoritmi di AEC e permette la discriminazione di double-talk detection. Quest'ultima azione, svolta dal double talk detector, ha mostrato probabilità di misdetection e false alarm attorno al 4-6%, paragonabile ai migliori algoritmi basati sulla cross-correlazione.

Il cancellatore implementato agisce su tutti i parametri di rilievo con i quali il codec AMR modellizza un segmento di parlato, in questa chiave risulta positivo che il segnale risulti già scomposto nelle parti di interesse, rendendone più semplice l'analisi. In particolare, agendo sui guadagni dei codebook, algebrico e adattativo, direttamente collegati all'energia del segnale, si può attenuare in maniera considerevole il segnale non desiderato. In più, agendo sui parametri direttamente collegati alla forma dello spettro del segnale, LSF e tempo di pitch, si è potuto sbiancare il segnale, togliendogli ogni residuo di coerenza spettrale. I risultati, in condizioni medie di funzionamento ($SNR = 12dB$ ed $ERL = 20dB$) si è mostrato attorno ai $20dB$ di echo return loss enhancement, attorno agli standard imposti sui cancellatori lineari tradizionali.

Il funzionamento di tutto il sistema è stato poi arricchito dalla presenza di un noise injector, con il quale si è potuta limitare l'azione di cancellazione per adattarsi alle caratteristiche del rumore di fondo, in modo da evitare fastidiose sensazioni di "linea caduta". Inoltre, modificando gli LSF, così da rendere il comportamento spettrale simile a quello del rumore, si è reso ancora migliore il noise injector.

Limiti e sviluppi futuri

Se la possibilità di avere il segnale già scomposto nelle sue caratteristiche principali è stato uno dei punti di forza dell'analisi, è anche vero che è e sarà la sua limitazione maggiore. Questo deriva principalmente dalla perdita di informazione che la compressione necessariamente impone, soprattutto se le potenzialità del codec AMR vengono sfruttate appieno, fino al rate di bit di informazione più basso ($4.75\,Kbit/s$). A questo punto sarebbe opportuno sviluppare algoritmi non lineari, fortemente basati sul riconoscimento dei segnali, e del parlato in particolare, ossia trovare nuove soluzioni nel campo dello speech recognition. Il segnale, in questo modo, potrebbe usufruire dell'informazione supplementare apportata dai modelli di riconoscimento, in modo da aiutare sia la rilevazione di attività vocale, sia la cancellazione d'eco.

Invece, nell'immediato proseguimento di questo lavoro di tesi, sarebbe opportuno sviluppare modelli statistici riguardanti il rumore non stazionario, sia per quanto riguarda il voice activity detector sia per quanto riguarda il cancellatore d'eco e in particolare il noise injector. Infatti, si è notato un leggero degrado delle prestazioni del VAD in presenza di questo tipo di rumore. Questo è spiegabile tramite la misura della coerenza spettrale del rumore colorato, che può far sbagliare gli algoritmi di classificazione basati sugli LSF, pur non presentando un comportamento formantico netto. È appunto in quest'ottica che l'algoritmo può essere migliorato introducendo un modello più raffinato di rumore.

Il noise injector, invece, con un modello di rumore più complesso, potrebbe offrire prestazioni psicoacustiche migliori, riuscendo a modellizzare più efficientemente il rumore, da sostituire al segnale cancellato, anche quando questo è decisamente non stazionario come, per esempio, il rumore babble o street.

Un ulteriore studio riguardante il Voice Quality Enhancement nel dominio codificato, sulla falsariga di questo lavoro di tesi, sarebbe quello di fare lavorare insieme l'algoritmo di cancellazione d'eco e di riduzione del rumore e non come due blocchi posti in cascata, creando importanti sinergie tra di essi, per non sprecare e non eliminare importanti informazioni sulla natura del segnale.

Bibliografia

[1] Adoul J.-P., P. Mabilleau, M. Delprat, S. Morissette, Fast CELP coding based on algebrical codes, *Proc. ICASSP'87*, pp.1957-1960, aprile 1987

[2] Ahgren P., *On system identification and acoustic echo cancellation*, PhD Thesis, Uppsala University, 2004.

[3] Atal B., S. Hanauer, Speech analysis and synthesis by linear prediction of speech waves, *Journal of the Acoustical Society of America*, vol.50, p. 637, 1971

[4] Atal B., M. R. Schroeder, Stochastic coding of speech signals at very low bit rates, *Proc. Int. Conf . on Communications*, maggio 1984, pp. 1610-1613

[5] Bellini S., *Trasmissione numerica*, Edizioni CLUP, 2004, 2a edizione

[6] Benesty J., T. Gansler, D. R. Morgan, M. M. Sondhi, S. L. Gay, *Advances in network and acoustic echo cancellation*, Springer-Verlag, Berlin, 2001, 1a edizione

[7] Benesty J., D. R. Morgan, J. H. Cho, A new class of doubletalk detectors based on cross-correlation, *IEEE Trans. On Speech and Audio Processing*, Vol. 8 pp. 168-172, March 2000

[8] Castiglione P., A. Cremascoli, *Classificazione del rumore in segnali audio codificati con tecniche a predizione lineare (ACELP)*, Tesi di Laurea, Politecnico di Milano, settembre 2005

[9] CCITT Recommendation G.721, 32 $Kbit/s$ adaptive differential pulse code modulation (ADPCM), *Blue Book*, vol. III, fascicolo III.3, ottobre 1988

[10] Clark A., T. Friedman, R. Caceres, Control protocol extended reports - VoIP metrics, *61st IETF Proceedings*, Washington, DC, USA, November 7-12, 2004.

[11] Dudley H., Remaking speech, *Journal of the Acoustical Society of America*, vol.11, p. 169, 1939

[12] Fant G., *Acoustic Theory of Speech Production*, Mounton & Co., Gravenhage, Paesi Bassi, 1960

[13] Feuer A., E. Weinstein, Convergence analysis of LMS filters with uncorrelated gaussian data, *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 33, pp. 222-230, Feb. 1985

[14] Flanagan J., *Speech analysis, synthesis and perception*, Springer-Verlag, New York, NY, 1972

[15] Gersho A., Advances in speech and audio compression, *Proc. IEEE*, Vol. 82, No. 6, giugno 1994

[16] Hagen R., E. Ekudden, B. Johansson, W.B. Kleijn, Removal of sparse-excitation artifacts in CELP, *Proc. ICASSP'98*, pp. I-145-I-148, 1998

[17] Hammer F., P. Reichl, A. Raake, Elements of interactivity in telephone conversations, *ICSLP'04*, Jeju Island, Korea, Oct. 2004

[18] Honkanen T., J. Vainio, K. Järvinen, P. Haavisto, R. Salami, C. Laflamme, J.-P. Adoul, Enhanced full rate speech codec for IS-136 digital cellular system, *Proc. ICASSP'97*, pp. 731-734, 1997

[19] Haykin S., *Adaptive filter theory*, Prentice-Hall, Englewood Cliffs, NJ, 1996.

[20] Itakura F., Line spectrum representation of linear predictive coefficients of speech signals, *Journal of the Acoustical Society of America*, vol.57, S35(A), 1975

[21] ITU-T Recommendation G.168, *Digital network echo cancellers*, 2002

[22] ITU-T Recommendation G.711, *Pulse code modulation (PCM) of voice frequencies*, 1988

[23] ITU-T Recommendation P.800, *Methods for subjective determination of transmission quality*, 1996

[24] Järvinen K., J. Vainio, P. Kapanen, T. Honkanen, P. Haavisto, R. Salami, C. Laflamme, J.-P. Adoul, GSM enhanced full rate speech codec, *Proc. ICASSP'97*, pp. 771-774.

[25] Kaaranen H., Ahtiainen A., *UMTS Networks: Architecture, Mobility and Services*, Wiley, April 2005

[26] Kabal P., R. P. Ramachandran, The computation of line spectral frequencies using Chebyshev polynomials, IEEE Trans. on ASSP, vol. 34, no. 6, pp. 1419-1426, dicembre 1986

[27] Kleijn B.W., T. Bäckström, P. Alku, On line spectral frequencies, *IEEE Signal Processing Letters*, vol. 10, no. 3, pp. 75-77, marzo 2003

[28] Kondoz A.M., *Digital speech: coding for low bit rate communication systems*, John Wiley & Sons, Hoboken, N.J., gennaio 2005, 2a edizione

[29] Levinson N., The Wiener Root Mean Square (RMS) Error Criterion in Filter Design and Prediction, *Journal of Mathematics and Physics*, vol. 25, pp. 214-218, 1947

[30] Liu C., M. Lin, W. Wang, H. Wang, Study of line spectrum pair frequencies for speaker recognition, *Proc. IEEE International Conference On Acoustics, Speech, Signal Processing*, febbraio 1990, pp. 277-280

[31] Lovekin J., R. E. Yantorno, Adjacent pitch period comparison (APPC) as a usability measure of speech segments under co-channel conditions, *Final report for Summer Research*

Faculty Program, Air Force Office of Scientific Research, Speech Processing Lab, Rome Labs, New York, 1998.

[32] Manteuffel D., A. Bahr, D. Heberling, and I. Wolff, Design considerations for integrated mobile phone antennas, in *Proc. 11th Int. Conf. Antennas Propagation*, Apr. 17–20, 2001, pp. 252–256.

[33] Markel J. D., A. Gray, *Linear prediction of speech*, Springer-Verlag, New York, NY, 1976

[34] Messerschmitt D. G., Echo cancellation in speech and data transmission, *IEEE Jour. on Selected Areas in Communications*, Vol. SAC-2 (2), p. 283, March 1984

[35] Nicholls D. F., B. G. Quinn, Random coefficients autoregressive model: an introduction, *Lecture notes in statistics*, Vol. 11, Springer, New York, 1982

[36] Oliver B. M., J. Pierce, C. E. Shannon, The philosophy of PCM, *Proc. IRE*, vol. 36, pp. 1324-1331

[37] Paliwal K. K., A study of Line Spectrum Pair for speech recognition, *Proc. IEEE International Conference On Acoustics, Speech, Signal Processing*, settembre 1988, pp. 485-488

[38] Paliwal K. K., A study of Line Spectrum Pair Frequencies for vowel recognition, *Speech Communication 8*, North-Holland, 1989, 27-33

[39] Paliwal K. K., B. Atal, Efficient vector quantization of LPC parameters at 24 bits/frame, *IEEE Trans. Speech and Audio Processing*, vol. 1, no. 1, pp. 3-14, 1993

[40] Proakis J. G., D. G. Manolakis, *Digital signal processing*, Prentice Hall, Upper Saddle River, NJ, 1996, 3a edizione

[41] Rabiner L., B. H. Juang, *Fundamentals of speech recognition*, Prentice Hall, New York, NY, 1993, 1a edizione

[42] Rabiner L. R., R. W. Schafer, *Digital processing of speech signals*, Prentice Hall, New York, NY, 1978, 1a edizione

[43] Rocca F., *Elaborazione numerica dei segnali*, Edizioni CUSL, Milano, 2003

[44] Schroeder M. R., B. Atal, Code-excited linear prediction (CELP): high quality speech at very low bit rate, Proc. ICASSP-85, aprile 1985, pp. 937-940

[45] Hansler E., G. Schmidt, *Acoustic echo and noise control: a practical approach*, John Wiley & Sons, Hoboken, N.J., 2004, 1a edizione

[46] Schur I., On Power Series Which Are Bounded in the Interior of the Unit Circle, *J. Reine Angew. Math.*, vol. 147, pp. 205-232, 1917

[47] Sohn J., N. S. Kim, and W. Sung, A statistical model-based voice activity detection, *IEEE Signal Processing. Lectures*, gennaio 1999, vol. 6, n. 1, pp. 1–3

[48] Soong F. K., B. H. Juang, Line Spectrum Pairs (LSP) and speech data compression, *Proc. IEEE International Conference On Acoustics, Speech, Signal Processing*, 1984, pp. 1.10.1-1.10.4

[49] Spanias A., Speech coding: a tutorial review, *Proc. IEEE*, Vol. 82, No. 10, pp. 1539-1582, 1994

[50] Tourneret, J. Y., M. Ghogho, Line spectrum Pairs in pattern recognition, *Signal Processing*, maggio 1999, vol. 43, no. 3

[51] Uberti M., *La voce* da *La Nuova Enciclopedia della musica*, Garzanti, Milano, 1996, 2° edizione

[52] Vainikainen P., J. Ollikainen, O. Kivekäs and I. Kelander, Resonator-based analysis of the combination of mobile handset antenna and chassis, *IEEE Transactions on Antennas and Propagation*, Vol. 50, No. 10, October 2002, pp. 1433-1444.

[53] Vaseghi S. V., *Advanced Digital Signal Processing And Noise Reduction*, John Wiley & Sons, Hoboken, N.J., marzo 2000, 2a edizione

[54] Widrow B., M. E. Hoff Jr., Adaptive Switching Circuits, *IRE Wescon Conc. Rec.*, 1960, part 4, pp. 96-104

[55] Wijngaarden, S. J., Communicability Testing for Voice Communications, *IEEE Workshop on Speech Coding*, Ibaraki, Japan, Oct. 2002.

[56] Zheng F., Z. Song, W. Yu, F. Zheng, W. Wu, The distance measure for line spectrum pairs applied to speech recognition, *Journal of Computer Processing of Oriental Languages*, marzo 2000, vol. 11, pp. 221-225

[57] 3rd Generation Partnership Project (3GPP), AMR Speech Codec: general description, *http://www.3gpp.org*, TS 26.071, versione 6.0.0, gennaio 2005

[58] 3rd Generation Partnership Project (3GPP), AMR Speech Codec: transcoding functions, *http://www.3gpp.org*, TS 26.090, versione 6.0.0, gennaio 2005