Università degli Studi di Pisa

Facoltà di Scienze Matematiche Fisiche e Naturali
Corso di Laurea in Matematica

TESI DI LAUREA

Tecniche di cepstrum per l'elaborazione del segnale vocale

Candidato
Fabio Natali

Relatore
Prof. Dario Bini

Controrelatore
Prof. Luca Gemignani

Anno Accademico 2003/2004

Introduzione

La tecnologia ha un ruolo sempre più importante nella nostra vita quotidiana. Usiamo computer nei lavori di ufficio, telefoni sofisticati per comunicare, sportelli automatici per versamenti o prelievi di denaro, viviamo in case e guidiamo auto sempre più "intelligenti".

Affinché si realizzi un reale miglioramento della nostra vita, è necessario studiare dei modi per far comunicare l'uomo con queste macchine in modo sempre più sicuro, efficace e comodo.

La voce, il modo più naturale per noi uomini di comunicare, sarebbe anche lo strumento migliore per interagire con una macchina. La *speech processing* è un settore dell'analisi dei segnali che si occupa proprio di come il segnale vocale possa essere elaborato in automatico da un calcolatore: di come una macchina possa riprodurre la voce umana, di come possa riconoscere la persona che pronuncia un certo discorso o le parole che compongono quel discorso; ancora, di come un segnale vocale possa essere codificato, compresso o di come ne possa essere migliorata la qualità. La Figura 1 mostra l'articolata ramificazione di questo campo di studi.

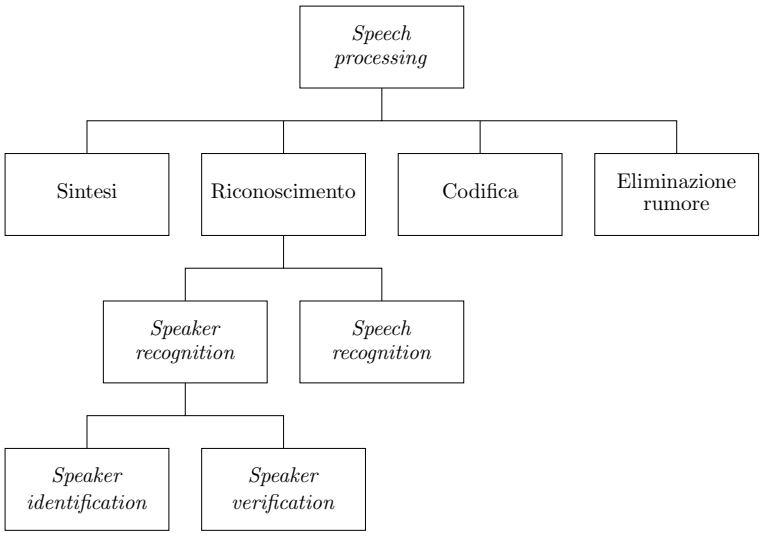


Figura 1: Principali settori di studio della *speech processing*

Sintesi vocale Studia il segnale vocale col fine di riprodurne le caratteristiche in modo artificiale, cioè avvalendosi di un computer e di un altoparlante.

Speech recognition Lo scopo è quello di riconoscere le parole che compongono un discorso dall'analisi del segnale audio. Ci si serve spesso di tecniche di intelligenza artificiale come le reti neurali.

Speaker recognition Un altro compito che risulta molto naturale per noi uomini, ma tuttaltro che facile per una macchina, è il riconoscimento di una persona dall'ascolto della sua voce. Questo settore a sua volta si suddivide in *speaker verification* e *speaker identification*. Nel primo caso l'utente

dichiara la sua identità e il problema è convalidare o meno questa dichiarazione (è il caso degli sportelli bancari, dove io dichiaro di essere Mario Rossi e il computer deve stabilire se è vero). Nel secondo caso, senza che ci sia una dichiarazione specifica, si vuole riconoscere l'utente tra un certo numero di individui registrati.

Codifica della voce In questo settore si cerca di ottenere dal segnale vocale analogico una rappresentazione digitale che, pur preservandone la qualità, permetta una consistente riduzione delle ridondanze e quindi del volume dei dati.

Per ciascuno di questi settori ci sono molte immediate applicazioni: la sintesi vocale e la *speech recognition*, tra l'altro, offrono aiuto ai disabili; la *speaker recognition* si offre come strumento per la sicurezza, permettendo l'autenticazione delle persone in base a parametri caratteristici della propria voce; la codifica della voce (in inglese, *speech coding*) serve nel settore delle telecomunicazioni per comprimere la banda di trasmissione: è alla base ad esempio del noto sistema GSM (*Global System for Mobile comunications*), lo standard per la telefonia mobile. Una breve rassegna delle possibili applicazioni della *speech processing* si può trovare in [9], in [10] e in [12].

1 Come viene prodotta la voce

In questo paragrafo daremo un breve resoconto di come funziona il nostro apparato vocale. Ne presenteremo poi un modello semplificato su cui si basano diverse tecniche di *speech processing*.

Le Figura 2 e la Figura 3 mostrano il nostro apparato vocale, rispettivamente, con un disegno in sezione e con un diagramma semplificato.

Durante il normale processo di produzione della voce, la cavità toracica si contrae forzando il passaggio dell'aria dai polmoni alla glottide, che è lo spazio tra le corde vocali, e da qui al tratto vocale, che è la porzione dell'apparato vocale al di sopra della glottide e comprende laringe, faringe, cavità nasale e cavità orale.

In base alla configurazione delle corde vocali e del tratto vocale vengono prodotti suoni diversi. Una prima distinzione può essere fatta tra i seguenti suoni:

Sonori Se durante il passaggio dell'aria le corde vocali sono in tensione, allora esse entreranno in vibrazione. Il flusso d'aria risulterà modulato in una serie di brevi soffi periodici. Questi impulsi periodici passeranno attraverso il tratto vocale e lo ecciteranno come se fosse una cassa di risonanza. I suoni prodotti da una tale configurazione dell'apparato vocale vengono detti *sonori* (in inglese, *voiced*); ne sono tipico esempio le vocali.

Sordi Se, al contrario, le corde vocali sono rilasciate, allora il flusso d'aria passerà attraverso la glottide inalterato. Si hanno allora due possibilità: suoni *sordi* o suoni *esplosivi*. Nel caso di suoni *sordi* (in inglese, *unvoiced*) ad una certa altezza del tratto vocale viene creata una strettoia attraverso cui l'aria passa a gran velocità producendo turbolenza. Un suono come la *s* di *sibilo*, ad esempio, è prodotto facendo passare con forza dell'aria in un piccolo canale tra lingua e incisivi.

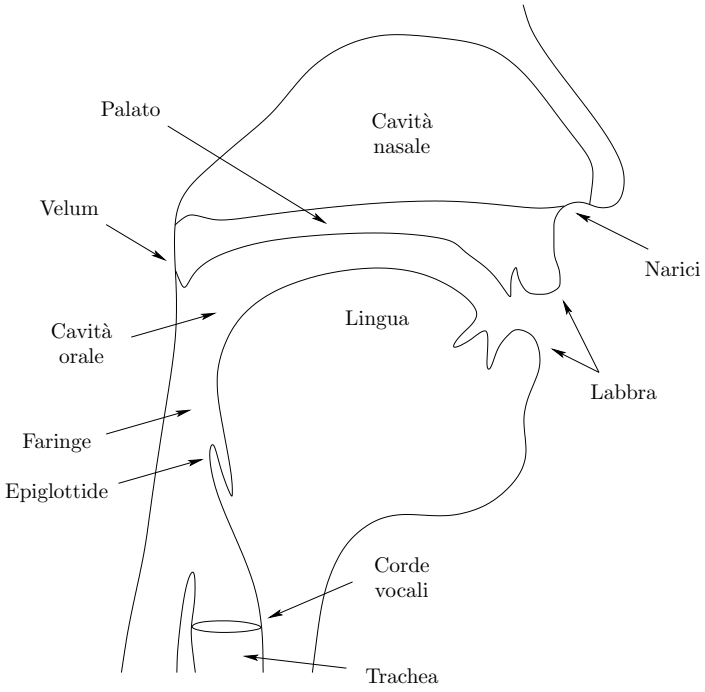


Figura 2: Disegno in sezione del tratto vocale

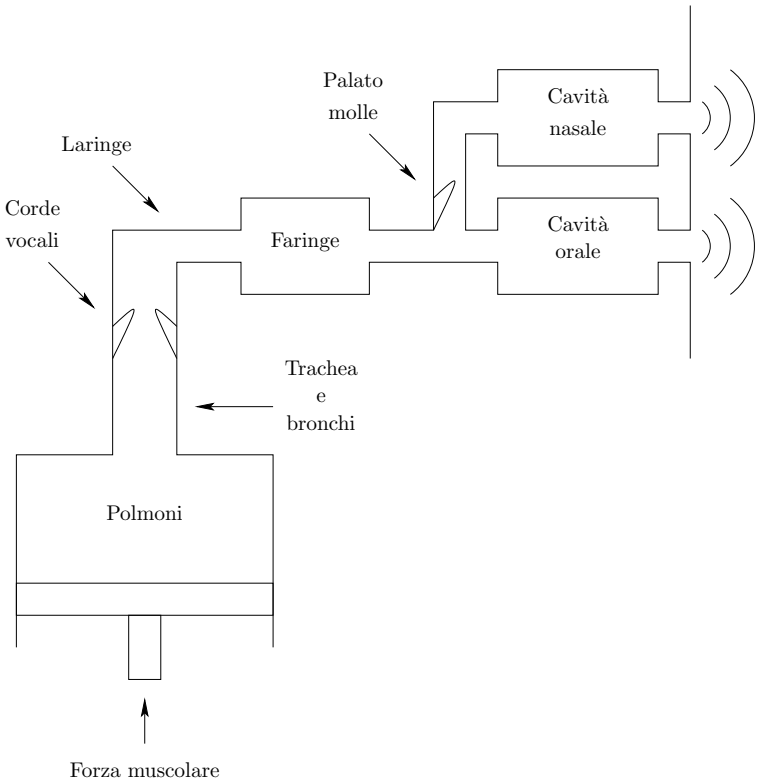


Figura 3: Una prima schematizzazione del tratto vocale

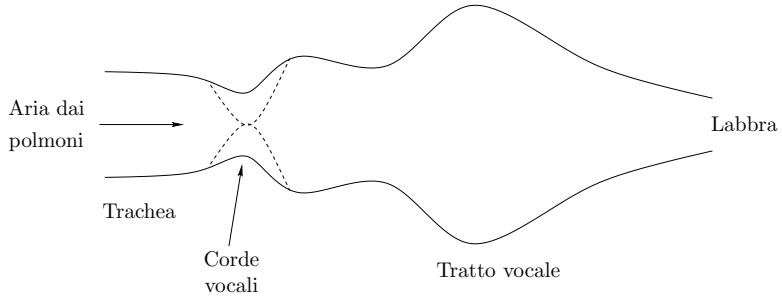


Figura 4: Modello *lossless tube* per il tratto vocale

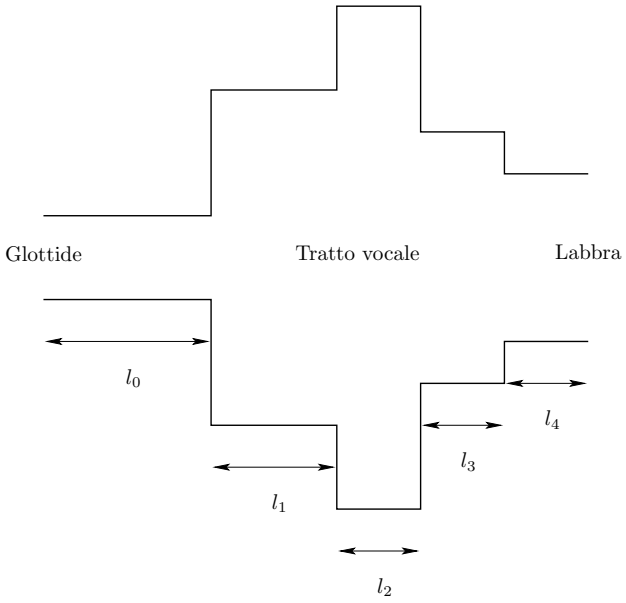


Figura 5: Modello a condotti stazionari, rappresenta un'ulteriore idealizzazione

Esplosivi I suoni *esplosivi* (in inglese, *plosive*) sono invece prodotti chiudendo le labbra per un breve periodo, facendo aumentare la pressione interna al tratto vocale e infine rilasciando velocemente l'aria trattenuta. Come esempio si consideri la *p* di *parola*.

2 Un modello molto semplice

Volendo semplificare grossolanamente possiamo considerare il tratto vocale come un tubo privo di perdite ad una estremità del quale viene immessa aria sottoforma, a seconda della posizione delle corde vocali, di una serie di brevi impulsi periodici o di un flusso costante. (Figura 4 e Figura 5)

Questo modello (in inglese, *lossless tube model*), nella sua essenzialità, rimane sufficientemente accurato per una vasta gamma di suoni, come le vocali, le fricative sorde e le esplosive.

3 Segnali, sistemi e convoluzione

Prima di continuare nella nostra esposizione, è necessario richiamare alcuni concetti base della *digital signal processing*, che ci serviranno per affrontare in modo più matematico il nostro studio.

3.1 Segnali

La serie di impulsi periodici generati dalle corde vocali e la stessa voce sono due esempi di ciò che in *digital signal processing* è chiamato **segnale**. Un segnale, infatti, è la descrizione di come una variabile (ad esempio la pressione dell'aria in corrispondenza della glottide) si modifica in relazione ad una seconda variabile (quest'ultima nella nostra analisi sarà sempre il tempo). Dunque, matematicamente un segnale è una funzione:

$$s : t \longmapsto s(t).$$

Si parla di segnali continui (o analogici) quando continuo è il dominio in cui varia t. In caso contrario, abbiamo un segnale discreto (o digitale) e possiamo scrivere

$$s : \mathbb{Z} \ni n \longmapsto s(n) \in \mathbb{R}.$$

Dato che siamo interessati a sviluppare algoritmi e tecniche gestibili da calcolatori e dalla loro aritmetica finita, ci limiteremo a considerare segnali discreti visti come funzioni da sottoinsiemi finiti di $\mathbb{Z}$ in sottoinsiemi finiti di $\mathbb{Q}$.

Il passaggio da segnali continui a segnali discreti prende il nome di campionamento (in inglese, *sampling*) e consiste semplicemente nel valutare un segnale continuo $s(t)$ ad intervalli regolari di tempo. Questo intervallo T si dice *periodo di campionamento* o *sample rate*. Il logaritmo in base due del numero di valori che abbiamo a disposizione quando valutiamo $s(n)$ si chiama invece *bit rate*. Tanto maggiori sono il sample rate e il bit rate, tanto maggiore sarà la qualità del segnale digitale ottenuto dal campionamento.

Per i segnali vocali che noi considereremo, un sample rate di 16000 Hz e un bit rate di 8 bit saranno più che sufficienti. Per fissare le idee, il segnale digitale registrato su un *compact disc* secondo lo standard ISO9660 ha un sample rate di 44100 Hz e un bit rate di 16 bit.

3.2 Sistemi

Col termine **sistema** si intende un qualunque processo che prende in input un segnale x e ne rende in output un altro $y = S(x)$, come è mostrato in Figura 6.

Il concetto di sistema ci viene in aiuto in una gran quantità di occasioni. Ad esempio, usiamo dei sistemi quando vogliamo attenuare il rumore di un segnale audio o migliorare un'immagine sfocata. Nel nostro caso, noi useremo un sistema come modello matematico per il tratto vocale.

La maggior parte dei sistemi con cui si ha a che fare in *digital signal processing*, e a cui noi stessi siamo interessati per il nostro studio, sono *sistemi lineari stazionari* (SLS). Un sistema si dice *lineare* se è tale che

$$S(\alpha x(n) + \beta y(n)) = \alpha S(x(n)) + \beta S(y(n))$$

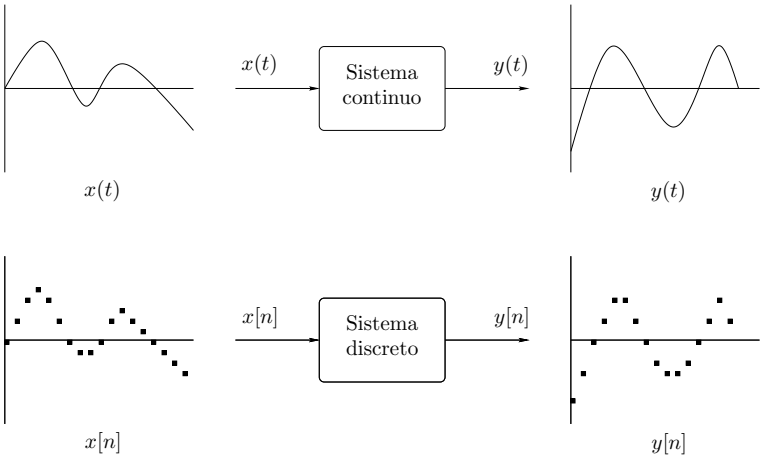


Figura 6: Sistemi continui e sistemi discreti

per ogni coppia di segnali $x(n)$, $y(n)$ e per ogni costante α e β. Un sistema si dice *stazionario* se è invariante per traslazione. Dunque, se

$$S(x(n)) = y(n)$$

allora

$$S(x(n+k)) = y(n+k).$$

3.3 Convoluzione

Dalla definizione stessa discende un aspetto fondamentale dei sistemi lineari stazionari: per caratterizzarne uno è sufficiente definirne la *risposta impulsiva*, ovvero la risposta ad un segnale $\delta(n) : \mathbb{Z} \longrightarrow \mathbb{R}$ definito come

$$\delta(n) = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases}$$

Infatti, per calcolare $S(x(n))$ sarà sufficiente riscrivere

$$x(n) = \sum_{k=-\infty}^{+\infty} \delta(n-k)x(k)$$

e poi sfruttare le proprietà di S, ovvero linearità e invarianza per traslazioni. Precisamente, se $s(n)$ è la risposta impulsiva di un sistema S, allora per ottenere $S(x(n))$ è sufficiente calcolare

$$y(n) = S(x(n)) = \sum_{k=-\infty}^{+\infty} s(n-k)x(k). \tag{1}$$

Questo calcolo è definito **convoluzione** e si indica usualmente con il simbolo $\otimes$, per cui la (1) diventa

$$y(n) = s(n) \otimes x(n). \tag{2}$$

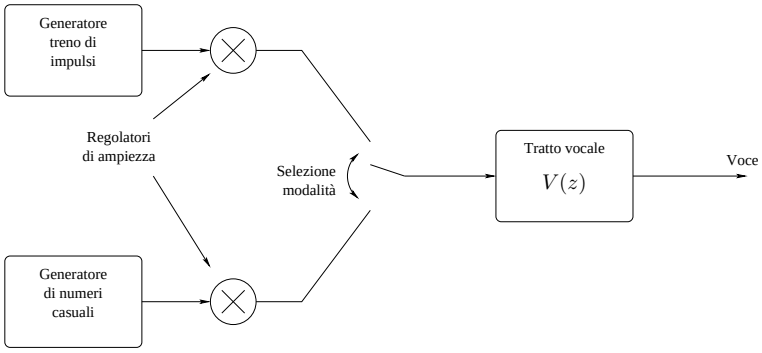


Figura 7: Modello del processo di produzione della voce. Il tratto vocale è un filtro lineare stazionario variabile nel tempo, ma che noi approssimeremo come statico per intervalli di 10-20 ms

Un altro modo spesso utilizzato per caratterizzare un SLS è quello di indicarne la *funzione di sistema*. Più precisamente, se introduciamo le serie formali di Laurent

$$\begin{aligned} S(z) &= \sum_{n\in\mathbb{Z}} s(n)z^n \\ X(z) &= \sum_{n\in\mathbb{Z}} x(n)z^n \\ Y(z) &= \sum_{n\in\mathbb{Z}} y(n)z^n \end{aligned}$$

che chiamiamo le z-trasformate rispettivamente di $s(n)$, $x(n)$ e $y(n)$, allora la relazione (2) che lega i coefficienti di $Y(z)$ con quelli di $S(z)$ e $X(z)$, può essere riscritta come

$$Y(z) = S(z)X(z).$$

Allora il SLS (2) viene univocamente determinato dalla funzione di sistema

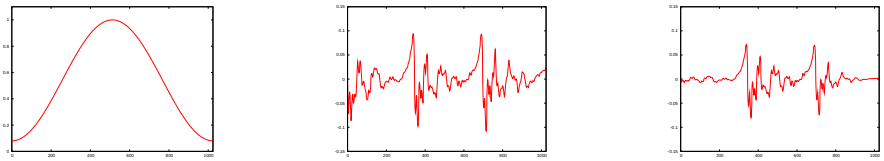
$$S(z) = \frac{Y(z)}{X(z)}.$$

4 Il modello matematico

Torniamo allo studio dell'apparato vocale. Utilizzando i concetti introdotti nelle sezioni precedenti, possiamo considerare il tratto vocale come un sistema lineare stazionario che riceve come input il segnale di eccitazione della glottide e lo trasforma in voce. La Figura 7 illustra questa idea, mettendo in evidenza il diverso comportamento della glottide per suoni sonori o sordi.

Nel caso di suoni sonori, le corde vocali sono in tensione e il tratto vocale viene eccitato da un treno di impulsi. Nel caso di suoni sordi o esplosivi, le corde vocali sono rilasciate e il segnale proveniente dai polmoni, non modulato dalla glottide, non è caratterizzato da particolari frequenze sonore e matematicamente viene ben modelizzato da un generatore di numeri casuali.

La configurazione del tratto vocale cambia istante dopo istante in base al suono che si deve produrre. Per poter considerare il tratto vocale come sistema



(a) La finestra di Hamming

(b) Il segnale vocale

(c) Il segnale vocale dopo la finestratura

Figura 8: Analisi di un segnale vocale tramite finestra di Hamming

lineare stazionario statico dobbiamo condurre la nostra analisi della voce su intervalli di tempo molto brevi. Tipicamente si moltiplica il segnale vocale con una finestra di Hamming di 10-20 ms, che serve a frammentare il segnale e ad attenuare le discontinuità agli estremi di ciascun frammento (vedi Figura 8). La finestra di Hamming è il segnale dato dall'espressione

$$w(n) = \begin{cases} 0.54 + 0.46\cos(\frac{2\pi n}{N-1}) & 0 \leq n \leq N-1 \\ 0 & \text{altrimenti} \end{cases}$$

Matematicamente, quindi il segnale della voce può essere considerato come la convoluzione di due segnali, quello di eccitazione proveniente dalla glottide e la risposta impulsiva del filtro del tratto vocale. Ovvero

$$s(n) = g(n) \otimes v(n) \tag{3}$$

laddove con $s(n)$ si è indicato un frammento di segnale vocale abbastanza piccolo da poter considerare statico il tratto vocale; con $g(n)$ il segnale di eccitazione, treno di impulsi per suoni sonori, successione di numeri casuali per suoni sordi; infine con $v(n)$ la risposta impulsiva del tratto vocale.

5 La deconvoluzione del segnale vocale

Un importante traguardo della *speech processing* è quello di risalire dal segnale vocale alle componenti $g(n)$ e $v(n)$ che l'hanno generato effettuando una deconvoluzione, ovvero invertendo in qualche modo l'operazione (3). In questo paragrafo mostreremo i principali risultati esposti in [4], in [7], in [8] e in [10], dove il problema della deconvoluzione del segnale vocale viene affrontato con una tecnica chiamata tecnica di *cepstrum*.

Le informazioni che riusciamo ad estrarre dal segnale $s(n)$ sono preziose. I valori che riguardano $v(n)$, ad esempio, sono caratteristici sia della persona che parla sia del particolare suono che in un certo istante viene emesso e ci vengono dunque in aiuto sia in problemi di *speaker recognition* sia di *speech recognition*.

La Figura 9 dà un'idea di come realizzare uno speciale filtro D capace di separare dal segnale $s(n) = v(n) \otimes g(n)$ una delle componenti, ad esempio $v(n)$.

In una discussione sommaria, i passaggi sono tre: per prima cosa usiamo un sistema D_* che ci permette di trasformare la convoluzione di $v(n)$ e $g(n)$ nella somma $\hat{s}(n) = \hat{v}(n) + \hat{g}(n)$ delle rispettive immagini $\hat{v}(n) = D_*(v(n))$ e

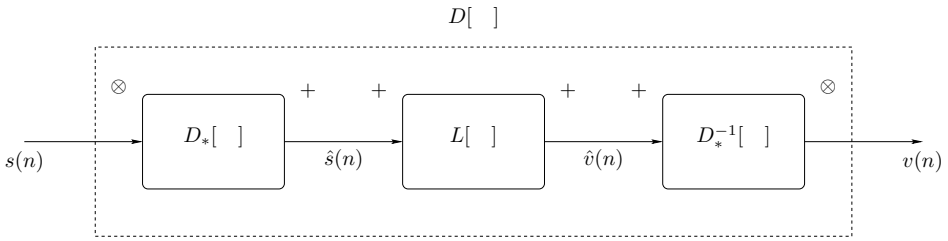


Figura 9: Deconvoluzione omomorfa di un segnale tramite tre opportuni sistemi in cascata. I segni $\otimes$ e $+$ ai lati dei riquadri indicano, stadio dopo stadio, il tipo di operatore con il quale agiamo sui segnali. Per cui se $s(n) = v(n) \otimes g(n)$ allora, dopo l'applicazione di D_*, avremo $\hat{s}(n) = \hat{v}(n) + \hat{g}(n)$.

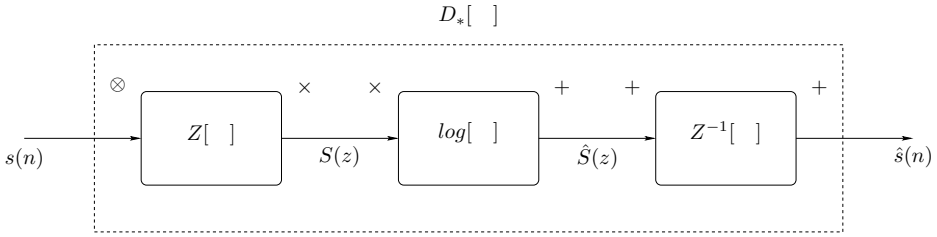


Figura 10: Sistema D_* realizzato tramite cepstrum: z-trasformata, logaritmo e z-trasformata inversa in successione. Prende in input il segnale $s(n)$ e restituisce il suo cepstrum $\hat{s}(n)$.

$\hat{g}(n) = D_*(g(n))$. Proprio in questo punto farà la sua comparsa il concetto di cepstrum.

Il secondo passaggio è un filtro convenzionale, ovvero un SLS che di un certo segnale in ingresso faccia passare solo alcune frequenze. Si tratta qui di sfruttare delle specifiche proprietà del segnale vocale e selezionare da $\hat{v}(n) + \hat{g}(n)$ quelle frequenze che caratterizzano uno dei due segnali, $v(n)$ ad esempio, escludendo $g(n)$.

Infine, il terzo sistema sarà semplicemente l'inverso del sistema iniziale e ci permetterà di riottenere $v(n) = D_*^{-1}[\hat{v}(n)]$.

Adesso dobbiamo occuparci della effettiva natura di questi tre sistemi.

5.1 Il cepstrum

Il primo sistema D_* può essere realizzato applicando in successione ad un segnale una z-trasformata, un logaritmo e una z-trasformata inversa (vedi Figura 10). Questa tecnica è conosciuta col nome di *cepstrum* (pronuncia *kepstrum*) e fu introdotta da Bogert, Healy e Tukey in [1], un articolo in cui viene proposto il cepstrum come strumento per il rilevamento dell'eco in un segnale. Assieme al termine cepstrum, ottenuto da spectrum tramite inversione delle prime quattro lettere, furono coniate anche le espressioni *alanysis*, *quefrency*, *gamnitude*, etc.

Nei paragrafi successivi useremo la notazione $\hat{s}(n)$ per indicare il cepstrum di un segnale $s(n)$.

Come abbiamo visto in 3.3 la z-trasformata di un segnale $x(n)$ è la funzione

di variabile complessa

$$X(z) = \sum_{n\in\mathbb{Z}} x(n) z^{-n}. \tag{4}$$

Adesso vogliamo ottenere l'operazione inversa della z-trasformata, cioè l'applicazione che ad $X(z)$ associa $x(n)$.

Assumiamo che $X(z)$ sia analitica per $z \in \mathcal{R} = \{z \in \mathbb{C} : \quad r < |z| < R\}$ dove R e r sono tali che $r < 1 < R$. Questa condizione è verificata in particolare se $x(n) = 0$ per $n < -N$ e per $n > N$, dove N è un opportuno intero positivo. Questa è ad esempio la situazione che si incontra dopo che il segnale è stato trasformato mediante la finestratura di Hamming.

Moltiplichiamo ambo i membri di (4) per z^{k-1} e calcolando gli integrali di entrambi i membri lungo la circonferenza unitaria si ottiene

$$\oint X(z) z^{k-1} dz = \sum_{n\in\mathbb{Z}} x(n) \oint z^{n-k-1} dz. \tag{5}$$

Adesso osserviamo che col cambiamento di variabile $z = \cos\theta + i\sin\theta$ si ha

$$\begin{aligned} dz &= (-\sin\theta + i\cos\theta)d\theta \\ &= i(\cos\theta + i\sin\theta)d\theta \\ &= iz d\theta \end{aligned}$$

e quindi l'integrale $\oint z^{n-k-1} dz$ si trasforma in

$$\begin{aligned} \oint z^{n-k-1} dz &= i\int_0^{2\pi} z^{n-k} d\theta \\ &= i\int_0^{2\pi} [\cos(n-k)\theta + i\sin(n-k)\theta] d\theta \end{aligned}$$

che vale zero se $k \neq n$ e vale $2\pi i$ se $k = n$. Giungiamo dunque all'espressione

$$x(n) = \frac{1}{2\pi i} \oint X(z) z^{n-1} dz \tag{6}$$

che rappresenta appunto la z-trasformata inversa.

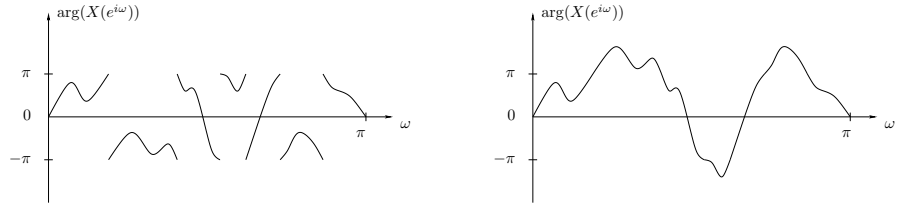
Per definire il cepstrum abbiamo bisogno di utilizzare il concetto di logaritmo di una funzione $X(z)$ di variabile complessa, analitica per $z \in \mathcal{R}$. Ricordiamo che se $X(z) = |X(z)|e^{\arg(X(z))}$, dove $\arg(X(z)) \in [-\pi, \pi[$ è l'*argomento principale* di $X(z)$, allora si definisce [5] logaritmo di $X(z)$ ogni funzione del tipo

$$\log(X(z)) = \log|X(z)| + i\arg(X(z)) + 2\pi ik, \quad k \in \mathbb{Z}.$$

Infatti per ogni valore di k vale $e^{\log X(z)} = X(z)$.

Ricordiamo che [5] ciascuno degli infiniti rami $\log z = \log|z| + i\arg(z) + 2i\pi k$ del logaritmo complesso di z, ottenuti al variare di $k \in \mathbb{Z}$, è una funzione analitica in tutti i punti z in cui essa è definita. Il ramo che si ottiene con $k = 0$ è chiamato *ramo principale* o *logaritmo principale*.

In generale se $X(z)$ è analitica per $z \in \mathcal{R}$, non è detto che il logaritmo principale $\log X(z) = \log|X(z)| + i\arg(X(z))$ sia funzione analitica di z. Ad esempio, se $X(z) = z^2$ e facciamo variare z sulla circonferenza unitaria, cioè $z = e^{i\theta}$, la parte immaginaria del logaritmo principale di $X(z)$ è discontinua per



(a) L'argomento principale **(b)** La curva di fase

Figura 11: Tipica curva di fase per una z-trasformata valutata sulla circonferenza unitaria

$\theta = \pi/2$ e per $\theta = -\pi/2$ e la funzione non è derivabile in questi punti e quindi non è analitica. Invece, scegliendo

$$\log X(z) = \begin{cases} \log|z^2| + \arg(z^2) - 2\pi i & \text{per } \theta > \pi/2 \\ \log|z^2| + \arg(z^2) + 2\pi i & \text{per } \theta < -\pi/2 \end{cases}$$

la funzione logaritmo così ottenuta è analitica in $\mathcal{R}$.

Nella Figura 11 si mostrano le discontinuità dell'argomento principale di $X(z)$ per una generica funzione $X(z)$.

In generale, come è mostrato in [11], se $X(z)$ è analitica in $\mathcal{R}$ è sempre possibile scegliere opportuni multipli di 2π a seconda dei valori di θ, in modo tale che la funzione logaritmo così ottenuta sia analitica su $\mathcal{R}$.

Naturalmente, se $\log X(z)$ è una funzione logaritmo analitica su $\mathcal{R}$ anche $\log X(z) + 2\pi ik$, $k \in \mathbb{Z}$ sono funzioni logaritmo analitiche. Per rimuovere questo ulteriore grado di libertà facciamo la seguente osservazione.

Assumiamo $|z| = 1$, più precisamente poniamo $z = e^{i\theta}$. Poiché $X(z)$ è la z-trasformata di una sequenza reale, la sua parte reale è funzione pari di θ, cioè vale $\mathrm{Real}(X(e^{i\theta})) = \mathrm{Real}(X(e^{-i\theta}))$ mentre la sua parte immaginaria è funzione dispari di θ cioè $\mathrm{Imag}(X(e^{i\theta})) = -\mathrm{Imag}(X(e^{-i\theta}))$. In particolare per $\theta = 0$ è $\mathrm{Imag}(X(1)) = 0$ e quindi $\log(X(1)) = \log|X(1)| + 2\pi ik$. Allora, fra tutte le funzioni analitiche del tipo $\log X(z) + 2\pi ik$ scegliamo quella la cui parte immaginaria si annulla per $\theta = 0$ e che quindi risulta dispari. La disparità della funzione $\log X(z)$ definita in questo modo è importante poiché ci permette di interpretarla come la z-trasformata di una funzione reale.

Osserviamo infine che per ogni scelta di $\log w$, $\log v$ e $\log vw$ vale

$$\log vw = \log v + \log w \qquad (\mathrm{mod}\ 2\pi i)$$

ma in generale è $\log vw \neq \log v + \log w$. Ad esempio se $V(z) = z$, $W(z) = z^2$, allora per $z = e^{i3\pi/8}$ risulta

$$\begin{aligned} V(z) &= e^{i3\pi/8} \\ W(z) &= e^{i3\pi/4} \\ V(z)W(z) &= e^{i9\pi/8} \end{aligned}$$

e quindi

$$\begin{aligned} \log(V(z)) &= i3\pi/8 \\ \log(W(z)) &= i3\pi/4 \\ \log(V(z)W(z)) &= i\pi/8 \\ \log(V(z)) + \log(W(z)) &= i9\pi/8 \end{aligned}$$

Però, come è mostrato in [11], se $X(z) = V(z)W(z)$ è il prodotto di funzioni analitiche $V(z)$ e $W(z)$ su $\mathcal{R}$, è possibile determinare mediante gli opportuni multipli interi di $2\pi i$, le tre funzioni $\log(X(z))$, $\log(V(z))$ e $\log(W(z))$ in modo che valga

$$\log(X(z)) = \log(V(z)) + \log(W(z)).$$

Nel seguito denoteremo con $\log X(z)$ quel logaritmo di $X(z)$ che risulta analitico in $\mathcal{R}$ e la cui parte immaginaria sia una funzione dispari di θ dove $z = e^{i\theta}$.

La definizione di cepstrum data prima a parole può finalmente essere espressa formalmente con la seguente formula

$$\hat{s}(n) = \frac{1}{2\pi j} \oint \log[S(z)] z^{n-1} dz \tag{7}$$

e riassumendo quanto detto finora, il cepstrum ci permette dunque di riscrivere la (3) come

$$\begin{aligned} s(n) = g(n) \otimes v(n) &\longrightarrow S(z) = G(z) \times V(z) \\ &\longrightarrow \log S(z) = \log G(z) + \log V(z) \\ &\longrightarrow \hat{s}(n) = \hat{g}(n) + \hat{v}(n). \end{aligned} \tag{8}$$

5.2 Il calcolo del cepstrum di $v(n)$

Come detto nella sezione 4, possiamo prendere come modello del tratto vocale un tubo a diametro variabile. Si dimostra (vedi ad esempio [3] o [10]) che per un tale modello la funzione di sistema nella sua forma più generale è

$$V(z) = \frac{K \prod_{k=1}^{M_a} (1 - a_k z^{-1}) \prod_{k=1}^{M_b} (1 - b_k z)}{\prod_{k=1}^{M_c} (1 - c_k z^{-1}) \prod_{k=1}^{M_d} (1 - d_k z)} \tag{9}$$

con $|a_k|, |b_k|, |c_k|, |d_k| < 1$, dove a_k con $k = 1, \ldots, M_a$ sono gli zeri di $V(z)$ interni al cerchio unitario e b_k^{-1} con $k = 1, \ldots, M_b$ sono gli zeri di $V(z)$ esterni al cerchio unitario, mentre c_k con $k = 1, \ldots, M_c$ sono i poli di $V(z)$ interni al cerchio unitario e infine d_k^{-1} con $k = 1, \ldots, M_d$ sono i poli di $V(z)$ esterni al cerchio unitario. Possiamo partire da qui per il calcolo di $\hat{v}(n)$.

Il logaritmo complesso di $V(z)$ è

$$\begin{aligned} \log[V(z)] &= \log K + \sum_{k=1}^{M_a} \log(1 - a_k z^{-1}) + \sum_{k=1}^{M_b} \log(1 - b_k z) \\ &\quad - \sum_{k=1}^{M_c} \log(1 - c_k z^{-1}) - \sum_{k=1}^{M_d} \log(1 - d_k z). \end{aligned} \tag{10}$$

I termini logaritmici possono essere riscritti come serie di potenze mediante lo sviluppo

$$\log(1-x) = -x - \frac{x^2}{2} - \frac{x^3}{3} - \dots$$

che è convergente per $|x| < 1$. Ad esempio il termine relativo ad a_k diventa

$$\log(1 - a_k z^{-1}) = -\sum_{n=1}^{\infty} \frac{a_k^n}{n} z^{-n}$$

e dunque

$$\sum_{k=1}^{M_a} \log(1 - a_k z^{-1}) = -\sum_{n=1}^{\infty} \sum_{k=1}^{M_a} \frac{a_k^n}{n} z^{-n}$$

per $|z| > \max|a_k|$.

Procedendo in modo analogo per le altre sommatorie in (10) si ottiene la seguente espressione di $\log[V(z)]$ che è convergente per $\phi < |z| < \psi$, dove $\phi = \max\{|a_n|, |c_n|\}$ e $\psi = \min\{|b_n^{-1}|, |d_n^{-1}|\}$

$$\begin{aligned} \log[V(z)] \quad = \quad & \log K - \sum_{n=1}^{\infty} \sum_{k=1}^{M_a} \frac{a_k^n}{n} z^{-n} - \sum_{n=1}^{\infty} \sum_{k=1}^{M_b} \frac{b_k^n}{n} z^n \\ & + \sum_{n=1}^{\infty} \sum_{k=1}^{M_c} \frac{c_k^n}{n} z^{-n} + \sum_{n=1}^{\infty} \sum_{k=1}^{M_d} \frac{d_k^n}{n} z^n. \end{aligned} \tag{11}$$

Applicando la (7) alla (11) si ottiene

$$\hat{v}(n) = \begin{cases} \log K & n = 0 \\ \sum_{k=1}^{M_c} \frac{c_k^n}{n} - \sum_{k=1}^{M_a} \frac{a_k^n}{n} & n > 0 \\ \sum_{k=1}^{M_b} \frac{b_k^{-n}}{n} - \sum_{k=1}^{M_d} \frac{d_k^{-n}}{n} & n < 0 \end{cases} \tag{12}$$

Il modello di tratto vocale che noi prenderemo in considerazione è un modello semplificato che trascura gli effetti dovuti alla cavità nasale ed ha come funzione di sistema

$$V(z) = \frac{K}{\prod_{i=1}^{M} (1 - a_i z^{-1})(1 - a_i^* z^{-1})} \tag{13}$$

dove, come sopra, si ha $|a_i| < 1$. Questo semplifica la (12), che diventa

$$\hat{v}(n) = \begin{cases} \log K & n = 0 \\ 2 \sum_{i=1}^{M} \frac{|a_i|^n \cos n\omega_i}{n} & n > 0 \\ 0 & n < 0 \end{cases} \tag{14}$$

laddove $a_i = |a_i| e^{j\omega_i}$.

L'equazione (12) e la (14) ci permettono di osservare una proprietà fondamentale: il cepstrum $\hat{v}(n)$ è una successione che assume valori decrescenti man mano che ci si allontana dall'origine. Precisamente, per la (12) possiamo dare la limitazione

$$|\hat{v}(n)| < \beta \frac{\alpha^{|n|}}{|n|} \qquad \text{per} \quad |n| \to \infty \tag{15}$$

dove α è il massimo tra i moduli di a_k, b_k, c_k e d_k e dove β è una costante moltiplicativa.

Per la (14), oltre a valere la limitazione precedente, osserviamo che $\hat{v}(n) = 0$ per $n < 0$. Segnali per cui vale la condizione precedente prendono il nome di segnali a *fase minima*. Per questo tipo di segnali è possibile utilizzare una definizione di cepstrum semplificata rispetto a quella usata finora. Precisamente, finora abbiamo lavorato con quello che di solito si chiama cepstrum *complesso*, in contrapposizione al cepstrum *reale* che indichiamo con $c_v(n)$ ed è dato da

$$c_v(n) = F^{-1} \log |F[v(n)]|$$

dove con F si è indicata la trasformata di Fourier, dunque una z-trasformata sul cerchio unitario $z = e^{i\omega}$. Si osservi che, avendo preso il modulo del valore $F[v(n)]$, il logaritmo usato nell'espressione precedente è un semplice logaritmo reale.

Vogliamo adesso mostrare che per i segnali a fase minima si può utilizzare la variante reale del cepstrum, anziché quella complessa, senza perdere informazioni. Innanzitutto la trasformata di Fourier del cepstrum complesso risulta

$$F[\hat{v}(n)] = \log |V(e^{i\omega})| + i\, arg[V(e^{i\omega})]$$

mentre per il cepstrum reale semplifichiamo con

$$F[c_v(n)] = \log |V(e^{i\omega})|$$

dunque $F[c_v(n)]$ è la parte reale di $F[\hat{v}(n)]$. Ma noi sappiamo che la parte reale della trasformata di Fourier di un segnale è la trasformata della parte pari della successione e dunque abbiamo che

$$c_v(n) = \frac{\hat{v}(n) + \hat{v}(-n)}{2}. \tag{16}$$

Usando il risultato appena ottenuto con la (16) e tornando a considerare la (14), scriviamo

$$\hat{v}(n) = \begin{cases} c_v(n) & \text{per} \quad n = 0 \\ 2c_v(n) & \text{per} \quad n > 0 \end{cases}$$

che conferma quanto annunciato, cioè che per un segnale a fase minima, come la risposta impulsiva $v(n)$ del tratto vocale, possiamo limitarci ad usare il cepstrum reale.

5.3 Il calcolo del cepstrum di $g(n)$

Veniamo ora al calcolo del cepstrum della componente $g(n)$, ovvero del segnale di eccitazione proveniente dalla glottide.

Nel caso di suoni sonori $g(n)$ è un treno di impulsi spaziati di un periodo N_0. Vogliamo dimostrare che anche il suo cepstrum $\hat{g}(n)$ è un treno di impulsi con spaziatura N_0. Cominciamo con lo scrivere

$$g(n) = \sum_{k=0}^{P} \alpha_k \delta(n - kN_0)$$

la cui z-trasformata è

$$G(z) = \sum_{k=0}^{P} \alpha_k z^{-kN_0} \tag{17}$$

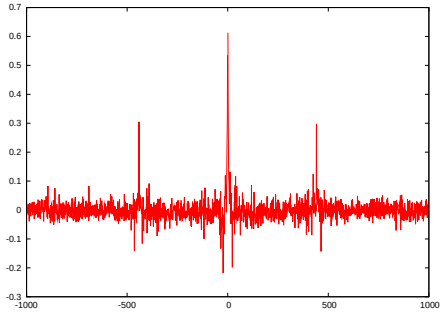


Figura 12: Il grafico del cepstrum per la vocale *a*. Si distinguono evidentemente i due diversi contributi di $\hat{v}(n)$ e di $\hat{g}(n)$, ovvero rispettivamente il picco centrale e i due picchi laterali.

ovvero un polinomio in z^{-N_0}, che può dunque essere espresso come prodotto di fattori della forma $(1 - az^{-N_0})$ e $(1 - bz^{N_0})$ e di un fattore potenza intera di z^{-N_0}. Si mostra perciò facilmente che il cepstrum $\hat{g}(n)$ sarà diverso da zero solo per multipli interi di N_0, ovvero sarà a sua volta un treno di impulsi. Vediamo questo nel caso più semplice in cui

$$g(n) = \delta(n) + \alpha\delta(n - N_0)$$

con $0 < \alpha < 1$. Allora si ha

$$G(z) = 1 + \alpha z^{-N_0}$$

e così, sviluppando il logaritmo in serie di potenze, si ottiene

$$\log G(z) = \log(1 + \alpha z^{-N_0}) = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{\alpha^k}{k} z^{-kN_0}$$

per $|z| > \alpha^{1/N_0}$. Perciò $\hat{g}(n)$ è

$$\hat{g}(n) = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{\alpha^k}{k} \delta(n - kN_0). \tag{18}$$

Nel caso di suoni sordi l'eccitazione proveniente dalla glottide è schematizzabile con una successione di numeri casuali e il cepstrum $\hat{g}(n)$ sarà a sua volta una successione di numeri casuali. Inoltre, data la minore intensità del segnale vocale nel caso di suoni sordi, il modulo degli elementi della successione $\hat{g}(n)$ è molto piccolo rispetto ai valori che si ottengono per i suoni sonori.

5.4 Il filtro lineare

Riassumendo quanto detto finora, ci aspettiamo che il cepstrum di un suono sonoro presenti un picco centrale, relativo al contributo di $\hat{v}(n)$ e altri picchi laterali, equidistanti tra loro e dall'origine, dovuti a $\hat{g}(n)$. Ci aspettiamo invece che nel cepstrum di un segnale sordo siano assenti i picchi laterali.

Nei paragrafi successivi descriveremo in dettaglio il procedimento usato per ottenere il grafico del cepstrum tramite un programma per calcolatore. Intanto,

con la Figura 12 mostriamo il cepstrum di un segnale vocale sonoro. Si osserva chiaramente il contributo di $\hat{v}(n)$, concentrato attorno all'origine, e il contributo di $\hat{g}(n)$, rappresentato dai picchi ad intervalli periodici.

Il filtro che, anche dall'analisi della figura, appare più naturale per la separazione dei due segnali $\hat{v}(n)$ e $\hat{g}(n)$ è

$$l_v(n) = \begin{cases} 1 & \text{per} \quad |n| < N_0 \\ 0 & \text{per} \quad |n| > N_0 \end{cases}$$

per ottenere $\hat{v}(n)$, e

$$l_g(n) = \begin{cases} 0 & \text{per} \quad |n| < N_0 \\ 1 & \text{per} \quad |n| > N_0 \end{cases}$$

per ottenere $\hat{g}(n)$.

Applicando tale filtro ed eseguendo la trasformazione inversa del cepstrum, possiamo finalmente giungere alla separazione delle due componenti $g(n)$ e $v(n)$.

6 Implementazione

Dopo aver passato in rassegna gli aspetti teorici delle tecniche di cepstrum, nei prossimi paragrafi cercheremo un riscontro pratico esaminando un programma per calcolatore appositamente scritto e discutendo i risultati che con questo si sono ottenuti.

6.1 Il programma

In Figura 13 è riportato il codice sorgente di un programma in linguaggio `C` con cui è possibile elaborare frammenti di segnale vocale.

Il programma prende in input i dati del segnale vocale in formato `dat`. Si ricorda che nel formato `dat` i suoni vengono rappresentati con due sequenze di numeri indicanti, la prima, l'istante in cui è effettuato il campionamento e, la seconda, la pressione sonora rilevata. All'inizio di ogni file `dat` è inoltre presente una riga di intestazione o *header* in cui è specificata la frequenza di capionamento. Come esempio, ecco le prime righe di un file `dat`:

```
; Sample Rate 12000
0                  0.0007514134
8.3333333e-05      0.0013812957
0.00016666667     0.00098947156
0.00025           0.00037017185
0.00033333333    -0.00025042519
0.00041666667     -0.0004120511
0.0005            0.00028538797
0.00058333333     0.00068122335
0.00066666667      -3.12021e-05
0.00075           7.3934905e-05
0.00083333333     0.00074160751
0.00091666667    -0.00044059195
0.001            -0.00075502321
0.0010833333      0.00043776073
.
.
.
```

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "misc.h"

#define NAME "mycepstrum"
#define VERSION "0.0"

int main(int argc,char **argv)
{
  //dichiaro
  int i=0,dimframe=128;
  FILE *input=stdin;
  FILE *output=stdout;
  double *complexframe=NULL;
  char buffer[50];

  //assegno a dimframe il valore passato da input (se c'e')
  if(argc>1)dimframe=atoi(*(argv+1));

  //alloco memoria
  complexframe=(double*)calloc(2*dimframe,sizeof(double));

  //assegno a complexframe i dati ricevuti in input
  for(i=0;i<dimframe;i++)
    {
      fscanf(input,"%s",buffer);
      *(complexframe+2*i)=atof(buffer);
      *(complexframe+2*i+1)=0.0;
    }

  //chiamo la function hamming per la finestratura
  hamming(dimframe,complexframe);

  //chiamo la function complexcep per calcolare il cepstrum
  complexcep(dimframe,complexframe);

  //mando in output il risultato tenendo conto della periodicita'
  for(i=1;i<dimframe;i++)
    {
      fprintf(output,"%d %.11f\n",i-dimframe*(i>dimframe/2),*(complexframe+2*i));
    }

  //chiudo gli stream
  fclose(input);
  fclose(output);

  //libero la memoria
  free(complexframe);

  //questo e' un classico
  return 0;
}
```

Figura 13: Codice sorgente del programma `mycepstrum`

I valori della pressione sonora che sono passati in input vengono inseriti nel vettore `complexframe`. Dato che nel corso del programma sarà necessario trattare con numeri complessi, si è adottata la convenzione di creare `complexframe` come vettore di lunghezza `2*dimframe` e ospitare ai posti pari le parti reali dei numeri e ai posti dispari quelle immaginarie.

Al vettore `complexframe` vengono applicate in successione due procedure o `function`, contenute nel file `misc.c` e riportate in Figura 14. La procedura `hamming`, moltiplica per l'omonima finestra il vettore `complexframe`. La procedura `complexcep`, il cuore del programma, esegue il cepstrum complesso. Per eseguire la trasformata di Fourier si è fatto uso delle librerie GSL, GNU Scientific Library, disponibili sotto licenza GPL, GNU General Public License.

Infine il programma rende in output i valori del cepstrum, pronti per essere disegnati.

6.2 Analisi di un segnale

La Figura 15 mostra il segnale vocale relativo alla parola *ascolta*, campionato con un sample rate di 16000 Hz.

Trovano facile riscontro le osservazioni fatte nei primi paragrafi relativamente alla diversa natura dei suoni vocali: i campioni del segnale che vanno dal numero 1000 al numero 2000 circa, corrispondono alla vocale a e appaiono modulati dalle corde vocali in un suono periodico (si veda anche l'ingrandimento di Figura 16(a)); i campioni dal 2000 al 3000 circa, corrispondono invece alla consonante s e mostrano il loro carattere aperiodico (si veda anche l'ingrandimento di Figura 16(b)); in corrispondenza del campione 4000 e del campione 10000 si notano infine delle piccole "discontinuità" dovute ai suoni esplosivi rispettivamente delle consonanti c e t.

Il segnale di Figura 15 può essere suddiviso in frammenti e analizzato tramite il programma `mycepstrum`. Concentriamo la nostra attenzione sui primi due suoni della parola *ascolta*: la vocale a e la consonante s, entrambi frammenti di 512 campioni, ovvero di circa 30 ms essendo il sample rate a 16000 Hz. I risultati ottenuti dal programma sono riportati in Figura 16, sotto le immagini dei rispettivi frammenti.

Si evidenzia la diversa natura del cepstrum per i suoni sonori rispetto a quelli sordi: il cepstrum del suono vocalico presenta, oltre a quello centrale, due picchi laterali che ne indicano il carattere periodico. Tutto ciò conferma quanto visto nel paragrafo 5.

7 Disclaimer e licenza

La parte informatica di questo breve lavoro è stata completamente realizzata con software libero, in prevalenza con software rilasciato sotto licenza GPL, GNU General Public License, si veda `http://www.gnu.org`.

Precisamente, si è usato LaTeX per l'editing, `gnuplot` per i grafici, `sox` per l'acquisizione e l'elaborazione audio, `gcc` come compilatore, le GSL come librerie matematiche; il tutto su un sistema operativo Debian GNU/Linux.

Anche questo breve lavoro viene rilasciato sotto i termini della GNU General Public License.

```
int hamming(int dimframe, double *complexframe)
{
  int i;
  double pi=4*atan(1.0);

  for(i=0;i<dimframe;i++)
    {
      *(complexframe+2*i)=
        (.54-.46*cos(2*pi*i/(dimframe-1)))*
        *(complexframe+2*i);
      *(complexframe+2*i+1)=
        (.54-.46*cos(2*pi*i/(dimframe-1)))*
        *(complexframe+2*i+1);
    }
}

int complexcep(int dimframe, double *complexframe)
{
  //dichiarazioni
  double a,b,pi;
  int i;

  //queste sono delle dichiarazioni per la fft
  gsl_fft_complex_wavetable *wavetable;
  gsl_fft_complex_workspace *workspace;
  //queste invece sono delle dichiarazioni per il logaritmo complesso
  gsl_sf_result rho;
  gsl_sf_result theta;

  //alloco memoria
  wavetable=gsl_fft_complex_wavetable_alloc(dimframe);
  workspace=gsl_fft_complex_workspace_alloc(dimframe);

  //pi greco
  pi=4*atan(1.0);

  //eseguo fft su complexframe
  gsl_fft_complex_forward(complexframe,1,dimframe,wavetable,workspace);

  //eseguo il logaritmo complesso su complexframe
  for(i=0;i<dimframe;i++)
    {
      a=*(complexframe+2*i);
      b=*(complexframe+2*i+1);

      if(a*a+b*b!=0)
        {
          *(complexframe+2*i)=log(sqrt(a*a+b*b));
          *(complexframe+2*i+1)=atan2(b,a);
        }
    }

  //eseguo ifft su complexframe
  gsl_fft_complex_inverse(complexframe,1,dimframe,wavetable,workspace);

  //libero memoria
  gsl_fft_complex_wavetable_free(wavetable);
  gsl_fft_complex_workspace_free(workspace);
}
```

Figura 14: Le due routine principali di `mycepstrum`

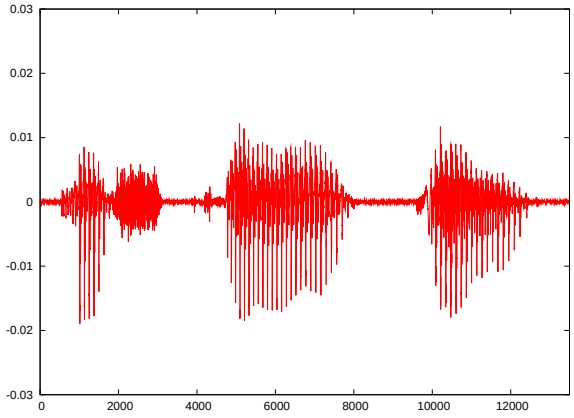
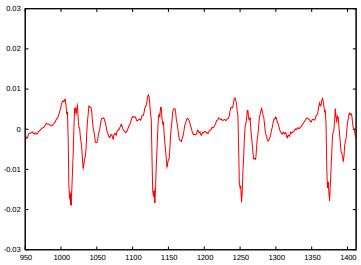
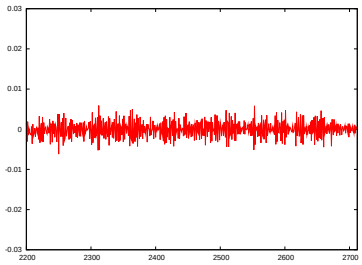


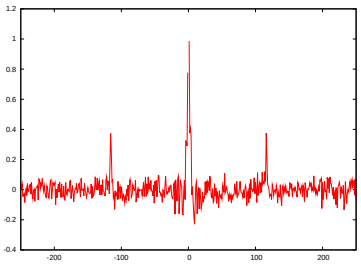
Figura 15: Il segnale vocale relativo alla parola *ascolta*, su cui si sono condotte le successive analisi tramite il programma `mycepstrum`



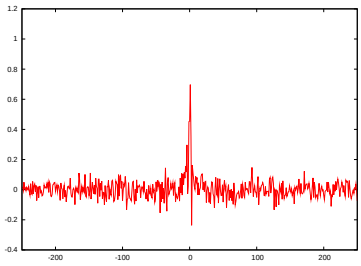
(a) Il frammento di segnale relativo alla vocale *a*



(b) Il frammento di segnale relativo alla consonante *s*



(c) Il cepstrum complesso per la vocale *a*



(d) Il cepstrum complesso per la consonante *s*

Figura 16: Il cepstrum complesso tramite il programma `mycepstrum`: si analizzano due frammenti estratti dalla parola *ascolta*, il primo è un suono sonoro e il secondo è un suono sordo. Si possono osservare i due picchi laterali nel cepstrum di Figura 16(c), dovuti proprio alla natura vocalica del frammento di segnale

Riferimenti bibliografici

[1] B. Bogert, M. Healy, J. Tukey, *The Quefrency Alanisys of Time Series for Echoes*, in M. Rosenblatt, *Proceedings of the Symposium on Time Series Analysis*, Wiley and Sons, 1963

[2] J. P. Campbell, *Speaker Recognition: A Tutorial*, Proceedings of the IEEE, Vol. 85, No. 9, September 1997 - `http://www.ee.columbia.edu/~patricia/papers/tutorials/tutorial.pdf`

[3] J. L. Flanagan, *Speech Analysis, Synthesis and Perception*, Spinger-Verlag, 1972

[4] B. Gold, C. M. Rader, *Digital processing of signals*, McGraw-Hill, 1969

[5] P. Henrici, W. R. Kenan, *Applied and computational complex analysis*, Wiley-Interscience, 1991

[6] A. V. Oppenheim, *A Speech Analysis-Synthesis System Based on Homomorphic Filtering*, J. Acoust. Soc. Am., Vol. 45, 1969

[7] A. V. Oppenheim, R. W. Schafer, *Discrete-time signal processing*, Prentice Hall, 1989

[8] A. V. Oppenheim, R. W. Schafer, T. G. Stockham, *Nonlinear Filtering of Multiplied and Convolved Signals*, Proceedings of the IEEE, Vol. 56, August 1968

[9] L. R. Rabiner, B. Gold, *Theory and application of digital signal processing*, Prentice Hall, 1975

[10] L. R. Rabiner, R. W. Schafer, *Digital processing of speech signals*, Prentice Hall, 1978

[11] R. W. Schafer, *Echo Removal by Discrete Generalized Linear Filtering*, Technical Report 466, Massachusetts Institute of Technology, February 1969 - `http://hdl.handle.net/1721.1/4277`

[12] S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*, California Technical Publishing, 1997 - `http://www.dspguide.com/`

Elenco delle figure

Indice