Predictive BCI Speller based on Motor Imagery

Tiziano D'Albis
tiziano.dalbis@gmail.com

Dipartimento di Elettronica e Informazione
Politecnico di Milano
Milano (Italy)

24, Ottobre 2008

Rossella Blatt
blatt@elet.polimi.it
Politecnico di Milano

Licia Sbattella
sbattella@elet.polimi.it
Politecnico di Milano

Capitolo 1

Analisi preliminare

1.1 Introduzione

L'obiettivo della tesi é quello di realizzare un sistema BCI che permetta a persone con disabilità motorie di comunicare in maniera efficace. A tal fine si vuole utilizzare il paradigma della *tastiera virtuale*, in cui una serie di simboli vengono successivamente selezionati dall'utente con l'ausilio di una interfaccia BCI.

Questo tipo di comunicazione, che rientra nell'ambito della Comunicazione Alternativa e Aumentativa (AAC), presenta come problematica peculiare la necessità di ridurre al minimo il numero di scelte a cui l'utente viene sottoposto durante l'utilizzo del sistema. Infatti la selezione di una scelta può risultare alquanto costosa sia in termini di tempo che di fatica per il particolare tipo di utenza a cui questi sistemi sono rivolti.

L'utilizzo di una interfaccia BCI, inoltre, impone anche un limite al numero di alternative disponibili per ciascuna scelta. Tale limite è legato al paradigma BCI utilizzato ed alla modalità con cui avviene il processo di selezione. Utilizzando un approccio a scansione, ad esempio, è possibile ovviare a questo tipo di limite a scapito però di allungare notevolmente i tempi necessari al processo di scelta.
Con un approccio a selezione diretta è invece possibile discernere efficacemente un massimo due, tre o quattro stati, a seconda del paradigma BCI adottato. Fanno eccezione alcuni sistemi basati su Motor Imagery che permettono di controllare il movimento di un cursore in una o due dimensioni per i quali sarebbe possibile associare una alternativa ad un target, ammettendo quindi un numero di stati potenzialmente superiore. In ogni caso anche questi sistemi non sono in grado di garantire una precisione nel movimento del cursore tale da discernere più di sei o otto stati.

L'idea che sta alla base della tesi è quella di utilizzare un approccio di tipo predittivo per agevolare il più possibile l'utente nella selezione dei simboli e minimizzare quindi sia il numero di scelte che il numero di alternative per ciascuna scelta.

1.2 Scelta dell'alfabeto

La scelta dell'insieme dei simboli da utilizzare per la comunicazione è di fondamentale importanza per la progettazione di tutto il sistema. Le alternative disponibili possono

essere ridotte a due categorie: l'alfabeto tradizionale (italiano o inglese) oppure un alfabeto iconico (come PCS o Blissymbolics). In entrambi i casi si hanno vantaggi e vantaggi.

La peculiarità di un alfabeto iconico è quella di riuscire ad esprimere con un solo simbolo un concetto semplice associabile a quello di una parola in un alfabeto tradizionale. Il vantaggio è quindi quello di ridurre notevolmente il numero di simboli che devono essere scelti durante la comunicazione. D'altro canto però il numero di simboli a disposizione è dell'ordine delle migliaia e risulterebbe quindi impossibile utilizzare l'intero alfabeto nell'applicazione che si vuole realizzare. Chiaramente sarebbe anche possibile utilizzare solo un sottoinsieme di simboli scelti dall'utente, limitando però le capacità espressive del sistema.

Un altro problema degli alfabeti iconici è legato alla difficoltà nel reperire corpora sufficientemente ampi che permettano l'applicazione di algoritmi predittivi con una fase di apprendimento. Non ultima è la considerazione che gli utenti a cui l'applicazione si rivolge conoscono l'alfabeto tradizionale e risulterebbe per loro alquanto gravoso l'apprendimento di un alfabeto iconico che comunque non sarebbe soddisfacente rispetto all'articolazione del linguaggio tradizionale a cui sono stati abituati.

La scelta dell'alfabeto tradizionale ha invece il vantaggio di avere un insieme limitato di simboli (26 più lo spazio per l'inglese), di poter applicare tutte le tradizionali tecniche di predizione sviluppate nell'ambito del Natural Language Processing e di avere a disposizione un discreto numero di corpora da utilizzare per l'apprendimento. Nel proseguo del documento viene quindi considerata valida questa seconda opzione.

1.3 Interfaccia utente

La progettazione dell'interfaccia utente è anch'essa di particolare rilevanza per lo sviluppo del progetto. Da essa dipende infatti il processo di selezione dei simboli e, insieme agli algoritmi di predizione, è un fattore determinante per l'efficienza del sistema.

L'idea generale è quella di raggruppare le lettere in un numero limitato di insiemi che vengono poi successivamente espansi fino alla selezione del singolo simbolo. La dimensione dell'insieme determina quindi il numero di scelte che sono necessarie per la selezione di un simbolo. Gli insiemi hanno dimensioni differenti in modo da agevolare la selezione delle lettere che più probabilmente verranno scelte dall'utente.

1.3.1 Suddivisione dei gruppi di lettere

La scelta del numero di gruppi è chiaramente determinata dal numero di stati che si hanno a disposizione. Dati i vincoli esposti in precedenza viene presa in considerazione la suddivisione in due, tre o quattro gruppi.

Una suddivisione dicotomica in due soli gruppi non sembra particolarmente adatta allo scopo. Infatti se utilizzassimo insiemi con egual numero di elementi avremmo bisogno di cinque scelte per la selezione di un simbolo, mentre in caso di sbilanciamento degli insiemi (necessario se si vuole utilizzare un approccio predittivo) saremmo costretti ad ammettere sei o più scelte per un simbolo che non sia tra quelli più probabili. Un tale numero di scelte sembra essere eccessivo rispetto alle esigenze del sistema.

D'altro canto una suddivisione in quattro gruppi permetterebbe sì un ridotto numero di scelte ma necessiterebbe di almeno cinque stati alternativi se si considera anche la necessità di codificare alcuni comandi aggiuntivi come la scelta di una intera parola predetta, la funzione di correzione degli errori o di sintesi vocale. Ciò non toglie che sarebbe comunque possibile avere tre gruppi di lettere al primo livello (lasciando il quarto alle funzioni) e poi quattro ai livelli successivi. Questa scelta potrebbe però essere critica per quanto riguarda la funzione di correzione degli errori che non sarebbe più accessibile durante il processo di espansione dei gruppi di lettere.

Una suddivisione in tre gruppi sembra quindi essere quella più adatta alla scopo in quanto rappresenta un giusto compromesso tra numero di stati alternativi e numero di scelte necessarie alla selezione di un simbolo. Si noti inoltre che se si considera un insieme di 27 simboli (che corrisponde a 3^3) esistono diverse modalità di suddivisione che permettono di utilizzare sempre tutti e tre i gruppi disponibili senza lasciarne alcuno inutilizzato.

Se volessimo utilizzare la suddivisione ottima in un alfabeto equiprobabile avremmo inizialmente tre gruppi da nove, ciascuno dei quali si espanderebbe in tre gruppi da tre che a loro volta si espanderebbero in tre gruppi da uno. Si raggiungerebbe quindi qualsiasi simbolo dell'alfabeto con sole tre scelte. Volendo invece utilizzare uno sbilanciamento tra gli insiemi che agevoli la scelta delle lettere più probabili dovremmo ammettere almeno quattro scelte per le lettere meno probabili.

Poiché esistono moltissimi modi per effettuare la suddivisione ricorsiva di 27 elementi in tre gruppi sono stati posti due limiti nella ricerca della suddivisione migliore: che il numero di scelte necessarie nel caso peggiore sia quattro e che non vengano mai lasciati gruppi vuoti. Posti questi due vincoli esistono 217 possibili suddivisioni che possono però essere raggruppate in 12 classi equivalenti. Ciascuna classe è determinata dal numero di lettere raggiungibili con una, due, tre o quattro scelte.

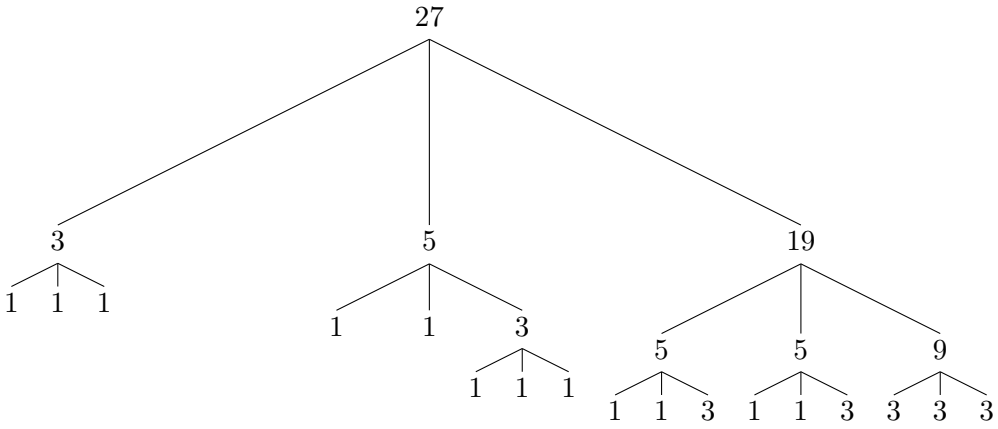
Classe	Una scelta	Due scelte	Tre scelte	Quattro scelte
1	2	0	1	24
2	1	3	2	21
3	1	2	6	18
4	1	1	10	15
5	1	0	14	12
6	0	6	3	18
7	0	5	7	15
8	0	4	11	12
9	0	3	15	9
10	0	2	19	6
11	0	1	23	3
12	0	0	27	0

Supponendo di ordinare le 27 lettere in base alla propria probabilità di occorrenza nel contesto, per determinare quale sia la classe migliore bisognerebbe stimare la probabilità con cui la lettera corretta appaia tra le prime x lettere predette. Se ad esempio la probabilità che la lettera corretta sia tra le prime due predette fosse molto alta sarebbe forse da valutare l'ipotesi di utilizzare una suddivisione di classe uno.

Naturalmente più l'algoritmo di predizione delle lettere si rivelerà accurato e più la scelta si indirizzerà su una classe che minimizza il numero di scelte per le prime lettere. Bisogna

comunque tenere presente che per avere un qualche vantaggio dalla predizione il numero di scelte necessario deve essere mediamente inferiore a tre.

Alla luce di queste considerazioni sembra più plausibile che la classe migliore possa trovarsi tra la sette, la otto e la nove. Poniamo ad esempio di scegliere la classe sette, una possibile suddivisione all'interno di questa classe è la seguente:



1.3.2 Comandi ausiliari

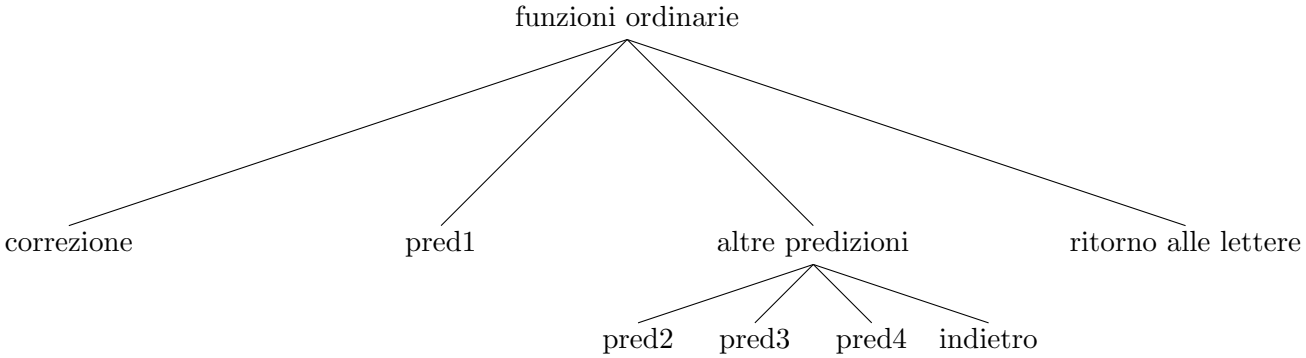
Oltre alla selezione delle lettere il sistema dovrà essere in grado di fornire dei comandi ausiliari per l'utilizzo dell'applicazione. Tali comandi potrebbero essere:

- annullamento dell'ultima scelta in caso di errore;
- scelta di una o più parole predette;
- ritorno alla scelta delle lettere dal gruppo delle funzioni;
- attivazione della funzione di sintesi vocale;
- funzione di chiusura dell'applicazione;

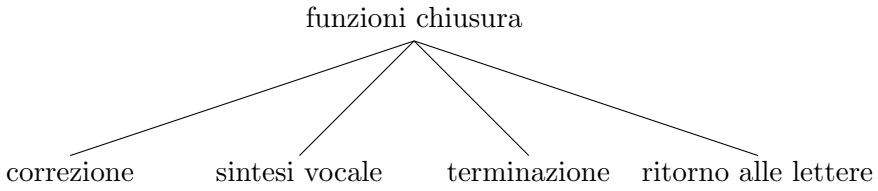
Se si volesse limitare l'applicazione a quattro stati sarebbe possibile avere un gruppo delle funzioni operante in due modalità:

- *modalità ordinaria*: durante la composizione di una lettera o di una parola;
- *modalità chiusura*: al termine della composizione di una parola.

Il gruppo delle funzioni durante la modalità ordinaria potrebbe essere strutturato in questo modo:



Mentre il gruppo delle funzione della modalità di chiusura potrebbe essere strutturato in questo modo:



1.4 Predizione

Il sistema dovrà disporre di algoritmi di predizione a due livelli:

1. *livello di lettera*: per poter assegnare le lettere più probabili ai gruppi meno numerosi;
2. *livello di parola*: per proporre una lista di parole predette da poter scegliere.

In generale i due algoritmi funzioneranno in maniera indipendente, anche se saranno comunque possibili delle interazioni. Per esempio l'algoritmo di predizione della parola escluderà tutte le parole che non inizino con le lettere già scelte, ed è possibile anche pensare ad un raffinamento della predizione delle parole anche durante la scelta dei gruppi di lettere. Inoltre la predizione delle lettere potrebbe essere anche pesata sulla base di una predizione a livello di parola (con una lista di parole più lunga rispetto a quella proposta all'utente) o basata solo su questa quando ci si trovi all'inizio di una parola.

1.4.1 Predizione delle lettere

Un primo approccio alla predizione delle lettere potrebbe essere quello di utilizzare semplicemente un ordinamento sulla base della frequenza assoluta all'interno del vocabolario, senza quindi tenere conto del testo già scritto. Questa tecnica ha il vantaggio di mantenere fissi i gruppi di lettere, che possono essere quindi imparati facilmente dall'utente con l'utilizzo dell'applicazione.

D'altro canto la possibilità di utilizzare il testo scritto in precedenza sembra poter dare vantaggi maggiori nella predizione. Esistono infatti successioni di lettere, anche tra quelle a frequenza più alta, che non potranno mai comparire. Supponiamo infatti che l'utente voglia scrivere la parola "gatto" ed abbia già scritto "ga", sarebbe a questo punto inutile

proporre la lettera “a” tra i gruppi più probabili perché in italiano non esistono parole contenenti una doppia “a”.

Un riarrangiamento dinamico dei gruppi ad ogni lettera potrebbe però essere d'ostacolo per l'utente che dovrà ogni volta cercare il gruppo contenente la lettera desiderata. Se dovesse comunque essere percorsa questa strada sarà allora necessario studiare le suddivisioni in modo da avere sempre almeno due gruppi con un numero limitato di elementi, permettendo all'utente di fare velocemente una scelta per esclusione.

Le tecniche tipicamente usate per la predizione a livello di lettera sono quelle che utilizzano *n-gram* e *k-gram*. Mentre con gli n-gram la predizione avviene sulla base delle $n-1$ lettere precedenti (con n fissato), con i k-gram la predizione avviene sulla base delle prime $k-1$ lettere di una parola (con k dipendente dunque dalla lunghezza della parola).

La predizione basata su k-gram porta tipicamente risultati migliori, ma essendo basata sulle parole non è in grado di fornire alcuna informazione per parole nuove non contenute nel dizionario di training. In questo caso sarà necessario utilizzare un meccanismo di backoff basato su n-gram, frequenza assoluta oppure un meccanismo a regole che penalizzi le sequenze di lettere impossibili.

1.4.2 Predizione delle parole

La predizione delle parole è anch'essa tipicamente basata su n-gram. In questo caso, se è già stato composto un inizio di parola, verrà dapprima filtrato il vocabolario con sole le parole con l'inizio scelto e poi si proporrà la parola più probabile date le $n-1$ parole precedenti.

Algoritmi più avanzati utilizzano anche tecniche di Part Of Speech (POS) tagging, includendo anche una analisi sintattica della frase che si sta componendo. Questo tipo di approccio necessita quindi di avere a disposizione corpora taggati o di utilizzare algoritmi per il tagging automatico. Prima di intraprendere questa strada bisognerebbe però valutare quale sia il reale beneficio del POS tagging rispetto al sistema che si vuole realizzare.

Sono da valutare inoltre tecniche si smoothing, per il probema delle frequenze nulle, e tecniche che permettano di adattare il modello linguistico durante l'uso dell'applicazione. Queste ultime potrebbero prevedere l'inserimento di una nuova parola nel dizionario (con conseguente aggiornamento delle frequenze) ed una predizione più facile per le parole più frequentemente o recentemente utilizzate da un utente.

1.4.3 La scelta dei corpora

Una difficoltà tipica nell'utilizzo degli n-gram per applicazioni di AAC sta nel reperire i corpora adatti per questo particolare tipo di comunicazione. E' infatti noto che le prestazioni di tali algoritmi dipendono notevolmente dalla pertinenza dei corpora. L'ottimo sarebbe l'utilizzo di testi prodotti direttamente dall'utente o comunque da persone con le medesime esigenze comunicative. Solitamente vengono utilizzati corpora di conversazioni telefoniche che più si avvicinano al tipo di comunicazione che si vuole agevolare.

Per la scelta dei corpora si potrebbe anche utilizzare una analisi di complessità, privilegiando testi che per sintassi, semantica o struttura generale dei periodi siano adeguati allo

scopo. Alternativamente sarebbe anche possibile definire diversi livelli di complessità e permettere all'utente di utilizzare quello che meglio rispecchi le proprie esigenze.

1.5 Interfaccia BCI

In questa sezione vengono valutati alcuni paradigmi BCI da utilizzare nell'acquisizione degl'input utente. In particolare si considera come requisito quello di poter effettuare una scelta tra quattro alternative, che corrispondono ai quattro gruppi di simboli identificati.

1.5.1 Approcci alternativi

Motor Imagery

Una prima possibilità sarebbe quella di utilizzare il paradigma Motor Imagery. Questo paradigma si basa sulla classificazione di un certo numero di stati mentali prodotti dall'immaginazione di un corrispettivo movimento da parte dell'utente. Se si riuscisse a discernere in maniera efficace quattro stati (escluso quello di riposo) sarebbe possibile associarli al movimento di un cursore in quattro direzioni: sinistra, destra, alto, basso. A questo punto un gruppo verrebbe selezionato nel momento in cui il cursore colpisce il target corrispondente. Si utilizzerebbe quindi un approccio di tipo asincrono, in cui i tempi di selezione dipendono esclusivamente dall'utente.

Alternativamente si potrebbe utilizzare un approccio di tipo sincrono in cui è invece il sistema a scandire i tempi di inizio e di fine di una selezione. In questo caso basterebbe riuscire a classificare correttamente solo tre movimenti ed associare lo stato di riposo alla quarta alternativa. La decisione dell'input potrebbe anche prescindere dal movimento continuo di un cursore ma basarsi su una classificazione discreta del segnale acquisito durante la finestra temporale di selezione. Ciò non toglie che un qualche tipo di feedback sarebbe comunque d'aiuto all'utente per un miglior controllo del segnale.

Rimanendo nell'ottica dell'approccio sincrono, un'altra possibilità sarebbe quella di classificare solo due movimenti (ad esempio L e R) e codificare i quattro gruppi sulla base di questi (LL, LR, RL, RR). In questo caso però si andrebbe a suddividere in due parti il processo di selezione aumentando sia i tempi necessari che il potenziale rischio di errore.

P300

Il paradigma P300 prevede l'identificazione all'interno del segnale EEG di un particolare potenziale che si manifesta quando il soggetto riceve uno stimolo "interessante" all'interno di un insieme di stimoli "poco interessanti". Evidenziando quindi i quattro gruppi in maniera casuale sarebbe possibile identificare il picco P300 nel momento in cui il gruppo evidenziato sia quello scelto dall'utente. Se questa dovesse essere la scelta sarebbe possibile utilizzare il sistema P300 già sviluppato presso l'AirLab.

SSVEP

Il paradigma SSVEP si basa invece sulla classificazione di potenziali visivi che si manifestano nell'osservare una luce che lampeggia ad una determinata frequenza. Esistono sistemi

SSVEP in grado di discernere efficacemente quattro stati alternativi (con frequenze di stimolo differenti) che verrebbero quindi associati ai quattro gruppi di simboli presenti nell'applicazione. Una possibilità potrebbe essere quella di utilizzare il sistema SSVEP realizzato presso il SensiLab del Polo di Lecco.

Dopo una prima analisi degli approcci BCI alternativi sopra descritti è stato stata scelta la strategia basata su Motor Imagery asincrono in quanto:

- non impone limiti teorici al numero di stati che possono essere utilizzati;
- fornendo un feedback continuo si pongono le basi affinché ci sia un adattamento anche a livello di utente;
- sono documentati in letteratura diversi studi che adottano questo approccio ottenendo buoni risultati;
- Motor Imagery può essere utilizzato anche dai disabili più gravi che hanno un controllo approssimativo o assente della direzione del proprio sguardo.

Capitolo 2

Interfaccia BCI

2.1 Algoritmo per il controllo del movimento del cursore

Il movimento del cursore è determinato da due equazioni lineari che ne determinano rispettivamente il movimento orizzontale e verticale:

$$\begin{cases} m_H(t,\tau) = a_H(t)(L_H(t,\tau)w_{HL} + R_H(t,\tau)w_{HR} + b_H(t)) \\ m_V(t,\tau) = a_V(t)(L_V(t,\tau)w_{VL} + R_V(t,\tau)w_{VR} + b_V(t)) \end{cases} \tag{2.1}$$

In cui per ciascuna delle due direzioni (H = orizzontale, V = verticale):

- $m(t,\tau)$ indica lo spostamento [1] del cursore associato al campione τ della trial t;
- $a(t)$ e $b(t)$ sono fattori di normalizzazione ricalcolati ad ogni trial sulla base di media e varianza dei segnali acquisiti;
- $L(t,\tau)$ e $R(t,\tau)$ sono le potenze (a frequenze specifiche) del segnale EEG acquisito rispettivamente da C3 e C4;
- w_L e w_R sono i pesi applicati alle potenze acquisite.

2.1.1 Adattamento dei pesi

L'algoritmo per il controllo del movimento del cursore prevede che, al termine di ogni trial, i pesi w_L e w_R vengano modificati in modo da utilizzare per la trial successiva i pesi che avrebbero dato le migliori performance per le trial passate. Se si considerano come segnali discreti:

- $TAR(t)$ che indica la posizione del target visualizzato nella trial t
- $M(t)$ che indica la posizione del cursore al termine della trial t

[1] espresso in un valore reale compreso tra -1 e 1

vogliamo trovare i parametri $\hat{w_L}$ e $\hat{w_R}$ che minimizzino lo scarto quadratico tra $TAR(t)$ e $M(t)$ per tutte le trial passate. Formalmente:

$$(\hat{w_L}, \hat{w_R}) = \underset{w_L, w_R}{\operatorname{argmin}} \sum_{t=1}^{N} (TAR(t) - M(t))^2 \tag{2.2}$$

Fissato T il numero di campioni per trial, esplicitiamo ora la dipendenza di $M(t)$ dai due parametri:

$$\begin{aligned} M(t) &= \sum_{\tau=1}^{T} m(t,\tau) \\ &= \sum_{\tau=1}^{T} a(t)(L(t,\tau)w_L + R(t,\tau)w_R + b(t)) \\ &= a(t)(w_L \sum_{\tau=1}^{T} L(t,\tau) + w_R \sum_{\tau=1}^{T} R(t,\tau) + Tb(t)) \end{aligned}$$

Se definiamo:

$$S_L(t) = \textstyle\sum_{\tau=1}^{T} L(t,\tau), \quad S_R(t) = \sum_{\tau=1}^{T} R(t,\tau)$$

otteniamo quindi:

$$\begin{cases} M(t) = a(t)(w_L S_L(t) + w_R S_R(t) + Tb(t)) & \textit{modello} \\ TAR(t) & \textit{dati} \end{cases} \tag{2.3}$$

Per semplificare i calcoli successivi possiamo considerare nel modello lo spostamento non normalizzato ed applicare sui dati la trasformazione inversa. Abbiamo quindi:

$$\begin{cases} \tilde{y}(t) = w_L S_L(t) + w_R S_R(t) & \textit{modello} \\ y(t) = \frac{TAR(t)}{a(t)} - Tb(t) & \textit{dati} \end{cases} \tag{2.4}$$

Possiamo dunque esprimere il modello in forma matriciale:

$$\tilde{y}(t) = \phi^T(t)\theta \tag{2.5}$$

Con:

$$\phi(t) = \begin{vmatrix} S_L(t) \\ S_R(t) \end{vmatrix}, \quad \theta(t) = \begin{vmatrix} w_L \\ w_R \end{vmatrix}$$

Dobbiamo quindi calcolare $\hat{\theta}(t)$ tale che:

$$\hat{\theta}(t) = \underset{\theta}{\operatorname{argmin}} \sum_{t=1}^{N} (y(t) - \tilde{y}(t))^2 \tag{2.6}$$