CAPITOLO 4: Stima spettrale mediante autoanalisi

4.1 Stima frequenziale mediante metodi di autoanalisi (*eigenanalysis methods*)

Supponiamo di avere un segnale come segue:

$$x(n) = \sum_{i=1}^{p} A_i e^{jn\omega_i} + \omega(n) \tag{4.1.1}$$

dove:

- $A_i = |A_i| e^{j\phi_i}$ $\quad \phi_i \in [-\pi\ \pi]$ uniformemente distribuite
- p è il numero di esponenziali complessi di cui è composto il segnale
- $\omega(n)$ rappresenta rumore bianco con densità spettrale di potenza pari a σ_ω^2

Il problema da risolvere è la stima delle frequenze ω_i e delle ampiezze $|A_i|$.
L'approccio seguito è quello di studiare la matrice di autocorrelazione del processo.

La sequenza di autocorrelazione di $x(n)$ è:

$$r_x(k) = \sum_{i=1}^{p} P_i e^{jk\omega_i} + \sigma_\omega^2 \delta(k) \tag{4.1.2}$$

dove $P_i = |A_i|^2$ è la potenza dell'i-esima componente.
La matrice di autocorrelazione MxM è definita come:

$$R_x = E\{x(n)x(n)^H\} \tag{4.1.3}$$

dove

$$x(n) = \begin{bmatrix} x(n) \\ \dots \\ x(n+M-1) \end{bmatrix} \tag{4.1.4}$$

Notare che è possibile scrivere:

$$R_x = R_s + \sigma_\omega^2 \cdot I \tag{4.1.5}$$

Se si seleziona $M > p$, R_s (che ha dimensione MxM) non è di rango pieno, perchè il suo rango è p. Comunque, R_x ha rango pieno perchè $\sigma_\omega^2 \cdot I$ ha rango M.
Infatti, la matrice R_s può essere scritta come:

$$R_s = \sum_{i=1}^{p} P_i e_i e_i^H \tag{4.1.6}$$

dove

$$e_i = \begin{bmatrix} 1 \\ e^{j\omega_i} \\ \dots \\ e^{j(M-1)\omega_i} \end{bmatrix} \tag{4.1.7}$$

ottenendo:

$$R_x = \sum_{i=1}^{p} P_i \begin{bmatrix} 1 & e^{-j\omega_1} & \dots & e^{-j(M-1)\omega_1} \\ e^{j\omega_1} & 1 & & \\ \dots & \dots & \dots & \dots \\ e^{j(M-1)\omega_1} & & & 1 \end{bmatrix} + \sigma_\omega^2 \cdot I \tag{4.1.8}$$

Dal momento che ciascun vettore $e_i e_i^H$ è una matrice di rango 1 e che ci sono p vettori prodotto, la matrice R_s ha rango p. Notare che se le sinusoidi sono reali, la matrice di correlazione R_s ha rango $2p$ (una sinusoide reale è composta dalla somma di due esponenziali complessi).

Calcoliamo adesso autovalori ed autovettori della matrice R_x. Supponiamo di ordinare gli autovalori in ordine decrescente con $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \dots \geq \lambda_M$ e denotiamo con $\{v_i, \text{ i} = 1,\dots, M\}$ i rispettivi autovettori. Assumiamo inoltre che gli autovettori siano normalizzati, cioè che $v_i^H \cdot v_j = \delta_{ij}$. In assenza di rumore gli autovalori $\lambda_1,\dots,\lambda_p$ saranno diversi da zero, mentre saranno nulli $\lambda_{p+1},\dots,\lambda_M$.

La matrice R_s può quindi essere scritta come:

$$R_s = \sum_{i=1}^{p} \lambda_i v_i v_i^H \tag{4.1.9}$$

Dal momento che gli autovettori v_i generano il sottospazio del segnale (come del resto anche i vettori e_i), vengono definiti *autovettori principali* e i corrispondenti autovalori vengono chiamati *autovalori principali*.

In presenza di rumore, la matrice di autocorrelazione relativa al rumore stesso può essere scritta come:

$$\sigma_\omega^2 \cdot I = \sigma_\omega^2 \sum_{i=1}^{M} v_i v_i^H \tag{4.1.10}$$

Riscrivendo, si ha

$$R_x = \sum_{i=1}^{p} \left(\lambda_i + \sigma_\omega^2\right) v_i v_i^H + \sum_{i=p+1}^{M} \sigma_\omega^2 v_i v_i^H \tag{4.1.11}$$

Questo ci porta a decomporre i rispettivi autovettori in due classi: una classe, quella degli autovettori principali, che genera il sottospazio del segnale, l'altra che genera il sottospazio del rumore.
Dal momento che gli autovettori relativi al rumore sono ortogonali a quelli principali, e che i vettori e_i (anch'essi generano il sottospazio del segnale) sono semplice combinazione lineare di questi ultimi, ne consegue che gli autovettori "rumorosi" sono ortogonali anche agli e_i.
Per sinusoidi complesse, se selezioniamo $M = p + 1$ (per sinusoidi reali scegliamo invece $M = 2p + 1$), esiste un solo autovettore nello spazio del rumore, autovettore che sarà associato al minimo autovalore. Quindi si ha:

$$e_i^H v_{p+1} = \sum_{k=0}^{p} v_{\min} e^{-j\omega_i k} = 0 \quad i = 1, ..., p \tag{4.1.12}$$

Le frequenze possono allora essere trovate calcolando gli zeri del polinomio

$$V(z) = \sum_{k=0}^{p} v_{\min} e^{-j\omega_i k} = \prod_{k=1}^{p} \left(1 - e^{j\omega_k} z^{-1}\right) \tag{4.1.13}$$

che giacciono tutti sul cerchio unitario.

Per stimare le potenze $P_i = |A_i|^2$ (A_i sono le ampiezze), si consideri:

$$R_x v_i = \lambda_i v_i \qquad \text{i} = 1, ..., p \tag{4.1.14}$$

con $v_i^H v_i = 1$.

Si ottiene:

$$\lambda_i = V_i^H R_x v_i = v_i^H \left\{ \sum_{k=1}^{p} P_k e_k e_k^H + \sigma_\omega^2 \cdot I \right\} v_i \tag{4.1.15}$$

Possiamo ottenere i P_i dall'insieme di equazioni lineari:

$$\sum_{k=1}^{p} P_k \left| e_k e_k^H \right|^2 = \lambda_i - \sigma_\omega^2 \qquad \text{i} = 1, ..., p \tag{4.1.16}$$

Notare che se definiamo $V_i(e^{j\omega}) = \sum_{l=0}^{p} v_i(l) e^{-jl\omega}$, ossia $V_i(e^{j\omega})$ è l'autofiltro corrispondente all'i-esimo autovalore, si ha:

$$\begin{bmatrix} \left|V_1(e^{j\omega_1})\right|^2 & \dots & \left|V_1(e^{j\omega_{p1}})\right|^2 \\ \dots & \dots & \dots \\ \left|V_p(e^{j\omega_1})\right|^2 & \dots & \left|V_p(e^{j\omega_{p1}})\right|^2 \end{bmatrix} \cdot \begin{bmatrix} P_1 \\ \dots \\ P_3 \end{bmatrix} = \begin{bmatrix} \lambda_1 - \sigma_\omega^2 \\ \dots \\ \lambda_3 - \sigma_\omega^2 \end{bmatrix} \tag{4.1.17}$$

4.2 Metodo di Pisarenko

Il metodo di Pisarenko [1] consiste nello scegliere $M = p + 1$, considerando così un solo autovettore che genera lo spazio del rumore. Due sono i metodi pratici che si possono utilizzare per calcolare le frequenze:

1. analizzare lo pseudospettro del segnale, generato variando la frequenza, e definito come:

$$P(e^{j\omega}) = \frac{1}{\left| e(\omega)^H v_{\min} \right|^2} \tag{4.2.1}$$

$$\text{dove } e(\omega) = \begin{bmatrix} 1 \\ e^{j\omega} \\ \dots \\ e^{j\omega p} \end{bmatrix} \qquad \text{(vettore di lunghezza } p + 1) \tag{4.2.2}$$

2. calcolare gli zeri dell'"autofiltro" in (4.1.13)

4.3 MUSIC

E' una generalizzazione dell'algoritmo di Pisarenko, basata sul porre $M > p + 1$
Lo pseudospettro del segnale si ottiene valutando la funzione:

$$P(e^{j\omega}) = \frac{1}{\sum_{i=p+1}^{M} \left| e(\omega)^H v_i \right|^2} \tag{4.3.1}$$

Generalmente MUSIC [2] dà risultati migliori del metodo di Pisarenko, andando a mediare fra loro tutti gli $M - p$ autovalori relativi al rumore, che in teoria dovrebbero essere tutti uguali, ma che in pratica non lo sono.
Il procedimento pratico segue esattamente quello già delineato per il metodo di Pisarenko, ovviamente con l'aggiunta appena descritta.

4.4 Smyth

L'algoritmo di Smyth [3] è un algoritmo basato su più stadi di calcolo. La prima parte è costituita da una versione modificata dell'algoritmo di Pisarenko (che chiameremo *Pisarenko vincolato*) che permette, in genere, di migliorare le stime prodotte rispetto dal metodo originale. I risultati forniti dall'algoritmo di Pisarenko vincolato vengono poi passati ad un altro algoritmo, l'algoritmo di Osborne/Bresler/Macovsky [4][5][6]. Quest'ultimo è un algoritmo iterativo, inserito allo scopo di produrre risultati per il terzo ed ultimo stadio, l'algoritmo ai minimi quadrati di Osborne e Smyth [7][8].
L'algoritmo di Pisarenko, come già visto, non richiede valori iniziali per poter funzionare; al contrario, l'algoritmo di Osborne ai minimi quadrati (quello utilizzato all'ultimo stadio) richiede, come tutti gli algoritmi di stima frequenziale basati sui minimi quadrati, una stima iniziale molto buona. L'algoritmo di Osborne/Bresler/Macovsky è invece molto meno sensibile ai valori iniziali di quello di Osborne e converge nella maggior parte dei casi se inizializzato con quello di Pisarenko. Da qui l'idea dell'algoritmo di Smyth in tre fasi distinte.

Il problema generale dell'autoanalisi può essere affrontato anche per via iterativa. Chiamiamo c il vettore dei coefficienti del polinomio annullatore; quindi sarà:

$$C(z) = c_1 + c_2 z + \ldots + c_{2p+1} z^{2p} \tag{4.4.1}$$

Si ricordi che il vettore c è l'autovettore associato al più piccolo autovalore di una matrice B opportunamente scelta, che in Pisarenko è la matrice di autocorrelazione del segnale.
Tentando di risolvere c per via iterativa, si considera la seguente espressione:

$$(B(c^k) - \lambda^{k+1} D) c^{k+1} = 0 \tag{4.4.2}$$

in cui c^k è la stima calcolata al passo k-esimo, c^{k+1} è la soluzione da calcolare, D è una matrice simmetrica costante e λ^{k+1} è la più vicina a zero di tali soluzioni.
Il problema (4.4.2) può essere riformulato come un problema di minimizzazione quadratica vincolata:

$$\min_{c^{(k+1)^T} D c^{(k+1)} = 1} c^{(k+1)^T} B\left(c^k\right) c^{(k+1)} \tag{4.4.3}$$

Una scelta comune per D è $D = I$, corrispondente al vincolo $\|c\| = 1$, oppure $D = \mathrm{diag}(0, 0, ..., 1)$ corrispondente al vincolo $c_{2p+1} = 1$.
Una proprietà importante del polinomio (4.4.1) posto uguale a zero è che le sue radici sono simmetriche, cioè se esiste una radice z deve esistere anche z^{-1}. Questa condizione su c è però solo necessaria ma non sufficiente per poter determinare che le radici di $C(z)$ giacciono sul cerchio unitario.
Il vincolo di simmetria può essere incorporato all'interno dell'autoproblema utilizzando il metodo suggerito in [8]. Supponiamo di avere le matrici Q_1 e Q_2 rispettivamente di dimensioni $(2p + 1)$ x $(p + 1)$ e $(2p + 1)$ x p:

$$Q_1 = \begin{pmatrix} I_p/\sqrt{2} & 0 \\ 0 & 1 \\ J_p/\sqrt{2} & 0 \end{pmatrix}, \qquad Q_2 = \begin{pmatrix} I_p/\sqrt{2} \\ 0 \\ -J_p/\sqrt{2} \end{pmatrix} \tag{4.4.4}$$

dove I_p è la matrice identità di dimensioni p x p e J_p è la matrice antidiagonale p x p:

$$J_p = \begin{pmatrix} 0 & & 1 \\ & \iddots & \\ 1 & & 0 \end{pmatrix} \tag{4.4.5}$$

Il vincolo di simmetria può essere allora rappresentato come $c = Q_1\gamma_1 + jQ_2\,\gamma_2$ in cui γ_1 e γ_2 sono vettori reali di dimensioni rispettivamente $p + 1$ e p.
Se supponiamo che il segnale da analizzare sia reale, come lo è nel caso di questa tesi, la matrice B ed il vettore c saranno reali; si deduce quindi che $\gamma_2 = 0$, e l'autoproblema (4.4.3) può essere allora riscritto come:

$$\left(Q_1^T B\left(c^k\right)Q_1 - \lambda^{k+1} Q_1^T D Q_1\right)\gamma_1^{k+1} = 0 \tag{4.4.6}$$

Il metodo dell'autoanalisi vincolata consiste quindi nel risolvere iterativamente la (4.4.6) nella variabile γ_1, quindi calcolare c dall'equazione $c = Q_1\,\gamma_1$. Notare che Q_1 è stata opportunamente scelta per soddisfare la relazione $Q_1^T Q_1 = I$, riducendosi all'autoproblema classico se $D = I$.
Per ragioni di semplicità il corrispondente problema per segnali complessi non viene qui trattato.

4.5 Esempi

In questo paragrafo si forniranno alcuni esempi pratici di utilizzo degli algoritmi di stima frequenziale basati sul metodo di autoanalisi sopra riportati (Pisarenko, MUSIC, Smyth).
Nel Signal Processing Toolbox di Matlab sono già presenti le funzioni *rootmusic* e *pmusic* che implementano MUSIC (e quindi Pisarenko): la prima restituisce frequenze e potenze delle sinusoidi[1] che compongono il segnale, la seconda ne restituisce lo pseudospettro.

Un esempio di chiamata della funzione *rootmusic* è:

$$[frequenze\ potenze] = rootmusic(x,seni)$$

dove:

- *x* è il segnale somma di sinusoidi
- *seni* è il numero di esponenziali complessi di cui è composto il segnale (ricordare che ogni sinusoide reale è somma di due esponenziali complessi)
- *frequenze* è il vettore contenente le frequenze stimate per ciascuna sinusoide in rad/sec
- *potenze* è il vettore delle potenze di ciascuna sinusoide; la potenza di una sinusoide reale vale $A^2/2$, dove *A* rapprenta l'ampiezza: è perciò immediato ricostruirsi le ampiezze a partire dalle potenze

Nei grafici seguenti verranno proposti i risultati ottenuti da ciascuno dei tre algoritmi al variare:

- del numero delle componenti sinosoidali
- del rapporto segnale rumore

In particolare, per ciascun esempio si riporterà:

- il grafico relativo al segnale somma di *N* componenti sinosoidali, con *N* variabile in base all'esempio proposto
- il grafico relativo allo stesso segnale con l'aggiunta di rumore gaussiano bianco a media nulla
- il segnale ricostruito in base ai risultati forniti dagli algoritmi

[1] Nell'help Matlab è erroneamente indicato che le potenze stimate dalla funzione *rootmusic* sono valutate in dB

4.5.1 Esempio (Pisarenko)

Prendiamo in considerazione il segnale

$$x(t) = 2\sin(0.1t)$$

composto da una singola sinusoide immersa in rumore, con SNR = 30 dB. Dopo l'applicazione del metodo di Pisarenko, i grafici ottenuti sono i seguenti:

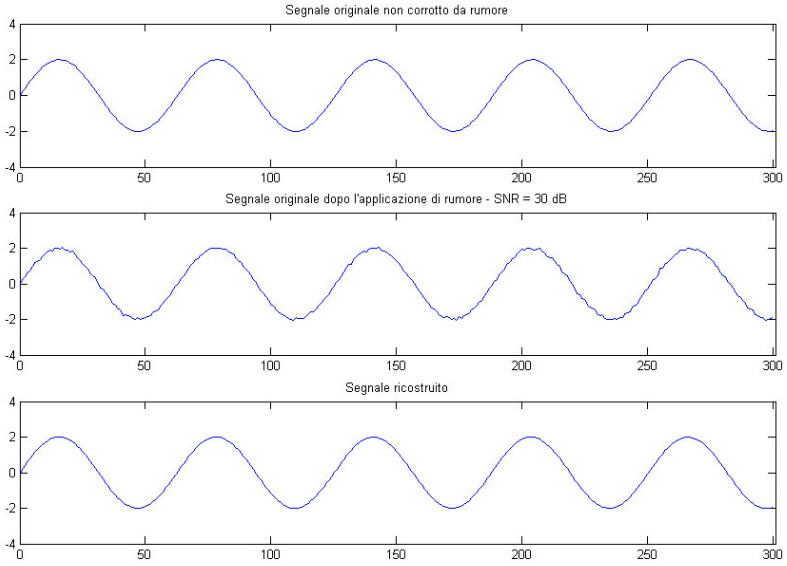

Figura 4.1. Ricostruzione di un segnale con una sola sinusoide e SNR = 30 dB

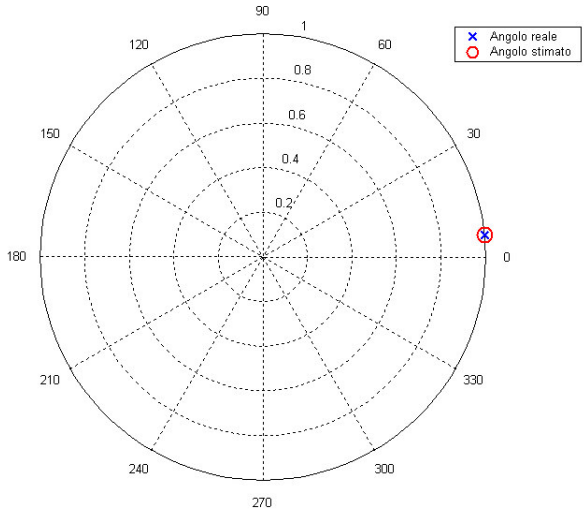

Figura 4.2. Pulsazione (reale e stimata) in gradi/sec sul cerchio unitario

In tabella si riportano i valori delle frequenze ed ampiezze stimate:

Ampiezza		**Frequenza (rad/sec)**	
Reale	**Stimata**	**Reale**	**Stimata**
2	2.0044	0.1	0.1004

Si noti quindi come con un segnale abbastanza pulito il metodo di Pisarenko riesca ad identificare correttamente frequenza ed ampiezza di una sinusoide. L'ampiezza viene infatti stimata con un errore relativo percentuale dello 0.22%, la frequenza con un errore relativo percentuale dello 0.4%.

Diminuiamo ora il rapporto segnale rumore, portandolo da 30 a 15 dB, per cercare di valutare fino a dove il metodo di Pisarenko riesce a fornire risultati attendibili. Dopo le simulazioni, i grafici ottenuti sono:

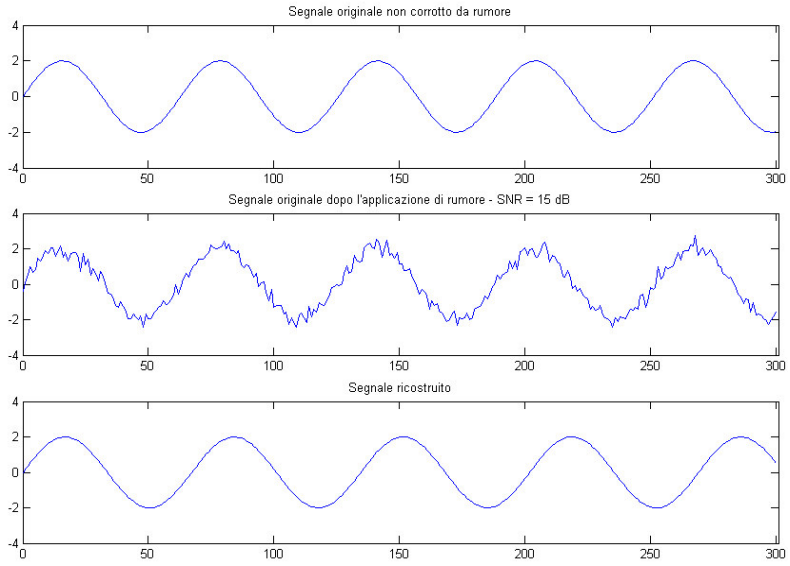

Figura 4.3. Ricostruzione di un segnale con una sola sinusoide e SNR = 15 dB

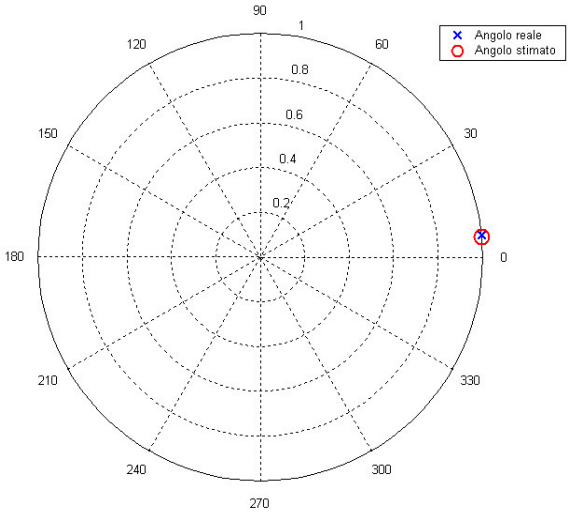

Figura 4.4. Pulsazione (reale e stimata) in gradi/sec sul cerchio unitario

In questo caso si ha un errore relativo in frequenza pari al 16.1%, che porta ad una sovrastima del periodo causando una dilatazione del segnale stesso.

Ampiezza		**Frequenza (rad/sec)**	
Reale	**Stimata**	**Reale**	**Stimata**
2	1.9971	0.1	0.0839

Diminuendo ancora il rapporto segnale-rumore si assiste ad una continua degradazione della stima, non avendo più nessuna informazione utile dal metodo di Pisarenko.

4.5.2 Esempio (Pisarenko)

Scegliamo adesso un segnale con due componenti sinusoidali

$$x(t) = 2\sin(0.1t) + \sin(0.5t).$$

Verifichiamo anche in questo caso la bontà dell'algoritmo.

I grafici che si ottengono sono riportati nella pagina seguente.

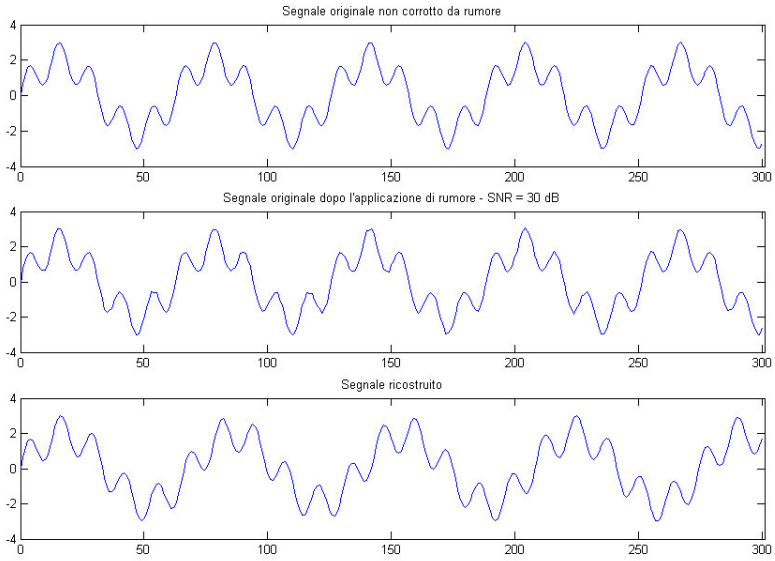

Figura 4.5. Ricostruzione di un segnale con due sinusoidi e SNR = 30 dB

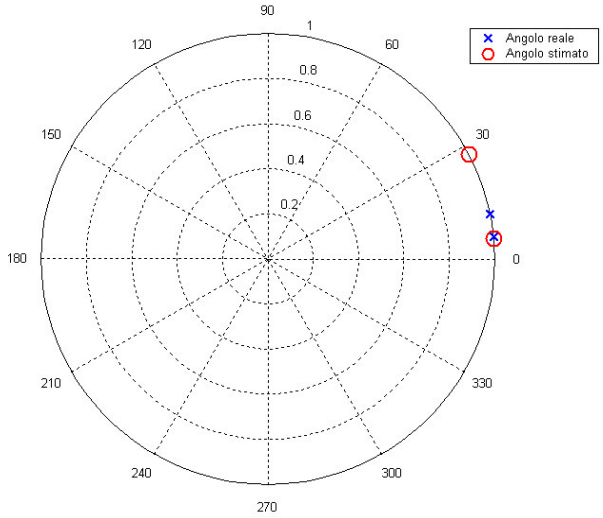

Figura 4.6. Pulsazione (reale e stimata) in gradi/sec sul cerchio unitario

Ampiezza		**Frequenza (rad/sec)**	
Reale	**Stimata**	**Reale**	**Stimata**
2	1.9749	0.1	0.0910
1	1.0405	0.5	0.4820

In tabella si nota come la prima frequenza a 0.1 rad/sec venga stimata a 0.0910 rad/sec con un errore relativo del 9%, mentre la seconda a 0.5 rad/sec venga stimata a 0.4820 rad/sec, con un errore relativo pari al 3.6%. Le ampiezze vengono ricostruite meglio delle frequenze: nella prima (ampiezza pari a 2) si commette un errore del 1.25%, nella seconda (ampiezza pari a 1) l'errore relativo è del 2.02%. Tale precisione non deve comunque fuorviare: infatti questi risultati sono stati ottenuti a partire da un segnale praticamente "pulito", in cui inoltre le frequenze sono state scelte opportunamente distanti.
Se si considera invece la seconda componente sinosoidale con frequenza non più 0.5 ma 0.2 rad/sec, ossia se il segnale considerato è

$$x(t) = 2\sin(0.1t) + \sin(0.2t)$$

si osserva come la stima non sia per niente attendibile. Nel segnale ricostruito, infatti, si osserva una sola frequenza, non due come desiderato. Il motivo di tutto ciò è semplice: durante la costruzione dello pseudospettro, i picchi di frequenza a 0.1 e 0.2 rad/sec non risultano distinti, quindi l'algoritmo stima una frequenza (al posto delle due menzionate) che in pratica è la media delle due (in questo caso la frequenza stimata è 0.1519, che è molto vicino al valor medio tra 0.1 e 0.2).

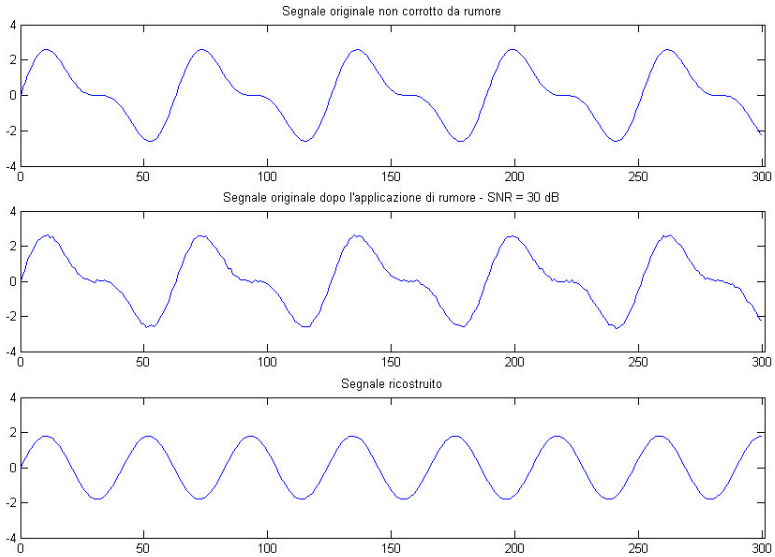

Figura 4.7. Ricostruzione di un segnale con una sola sinusoide e SNR = 30 dB (in questo caso il metodo di Pisarenko fallisce)

Ampiezza		Frequenza (rad/sec)	
Reale	**Stimata**	**Reale**	**Stimata**
2	1.8097	0.1	0. 1526
1	1.2526	0.2	

Quello che si osserva è che aumentando la complessità del segnale (inserendo cioè più componenti sinosoidali), pur avendo un rapporto segnale-rumore molto alto (dell'ordine dei 40 dB), le stime di Pisarenko non sono più attendibili. In effetti, come noto in letteratura, il punto forte del metodo di Pisarenko non sta tanto nell'essere un metodo robosto (ed in effetti non lo è) per identificare sinusoidi immerse in rumore, quanto nell'aver dato un notevole impulso alla ricerca in questo settore.

4.5.3 Esempio (MUSIC)

Cerchiamo di fare subito un raffronto con l'algoritmo di Pisarenko, considerando il segnale

$$x(t) = 2\sin(0.1t)$$

dell'esempio 4.5.1. Cerchiamo di verificare che MUSIC dà stime migliori del metodo di Pisarenko andando a considerare più autovettori associati al rumore e mediando tra loro i rispettivi autovalori. Il metodo di Pisarenko per il segnale considerato iniziava a dare problemi con SNR = 15 dB. Consideriamo allora questo SNR come punto di partenza, ottenendo:

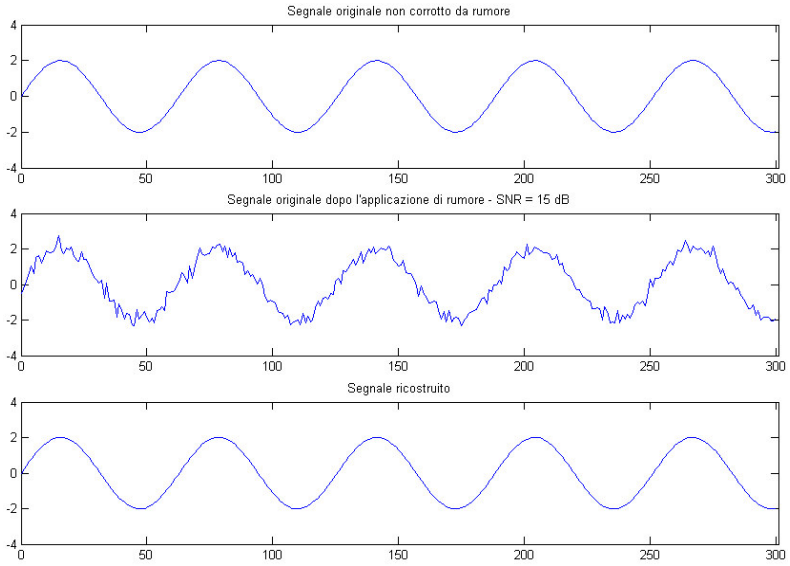

Figura 4.8. Ricostruzione di un segnale con una sola sinusoide e SNR = 15 dB

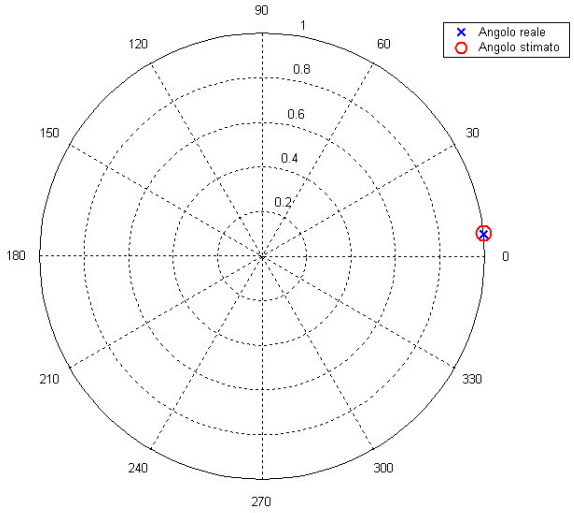

Figura 4.9. Pulsazione (reale e stimata) in gradi/sec sul cerchio unitario

Ampiezza		**Frequenza (rad/sec)**	
Reale	**Stimata**	**Reale**	**Stimata**
2	2.0153	0.1	0.1001

Dai valori in tabella ci si rende conto del vantaggio ottenuto utilizzando MUSIC piuttosto che il metodo di Pisarenko. In questo caso, ponendo per MUSIC $M = 6$ (la relazione $M > p + 1$ è quindi rispettata, essendo p il numero di esponenziali complessi del segnale, in questo caso 2) si ottengono risultati molto simili a quelli ottenuti con il metodo di Pisarenko in cui il segnale presentava però un SNR doppio (30 dB invece che 15 dB come nell'esempio attuale). Portando M a 10, si può diminuire ancora il SNR fino a 3 dB senza problemi.

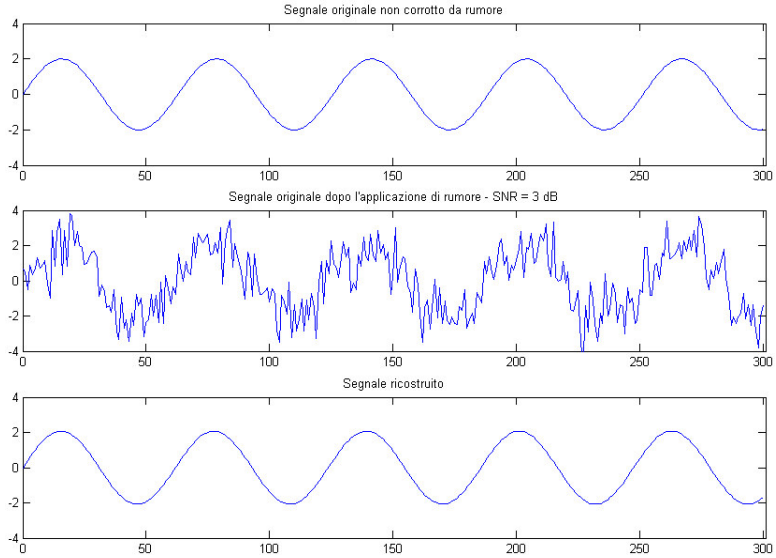

Figura 4.10. Ricostruzione di un segnale con una sola sinusoide e SNR = 3 dB

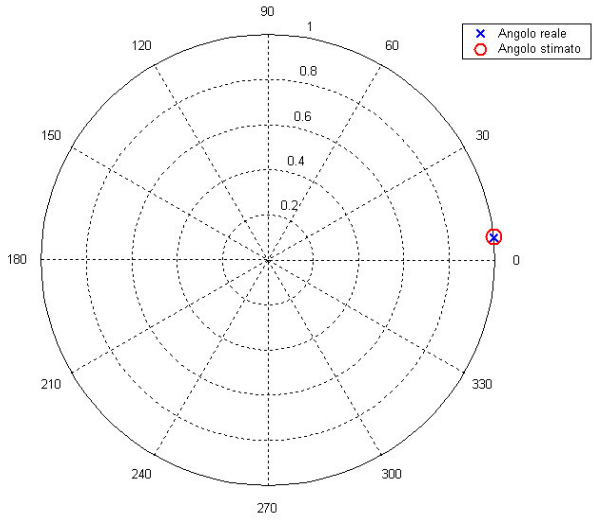

Figura 4.11. Pulsazione (reale e stimata) in gradi/sec sul cerchio unitario

Ampiezza		Frequenza (rad/sec)	
Reale	**Stimata**	**Reale**	**Stimata**
2	2.0749	0.1	0.1016

Già in questo primo esempio si evidenziano le diverse prestazioni tra i due algoritmi considerati.

4.5.4 Esempio (MUSIC)

Consideriamo il segnale

$$x(t) = 2\sin(0.1t) + \sin(0.2t)$$

dell' esempio 4.5.2. Con un SNR pari a 30 dB il metodo di Pisarenko non riusciva ad individuare le frequenze corrette. Scegliendo $M = 20$, con MUSIC si ottiene:

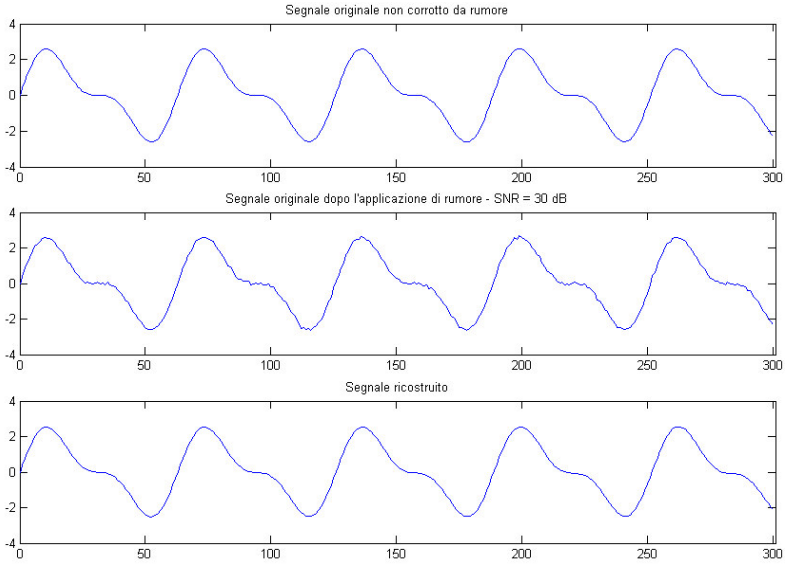

Figura 4.12. Ricostruzione di un segnale con due sinusoidi e SNR = 30 dB

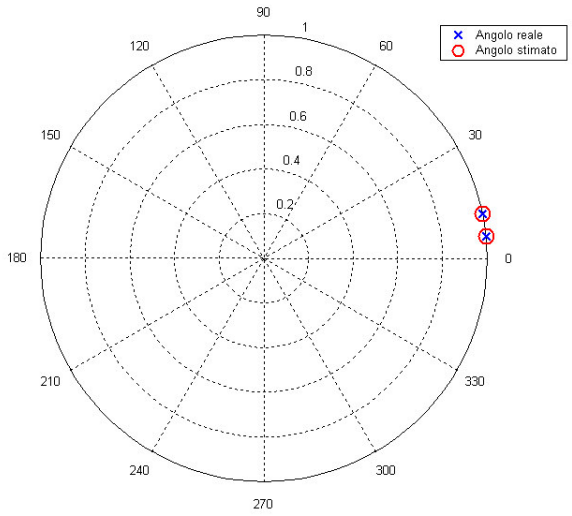

Figura 4.13. Pulsazione (reale e stimata) in gradi/sec sul cerchio unitario

Ampiezza		Frequenza (rad/sec)	
Reale	**Stimata**	**Reale**	**Stimata**
2	1.9801	0.1	0.0999
1	0.9319	0.2	0.1995

Le prestazioni sono evidentemente migliori di quelle fornite dall'algoritmo di Pisarenko. Ma cosa succede se diminuiamo il SNR fino a 5 dB? I risultati sono evidenziati in figura.

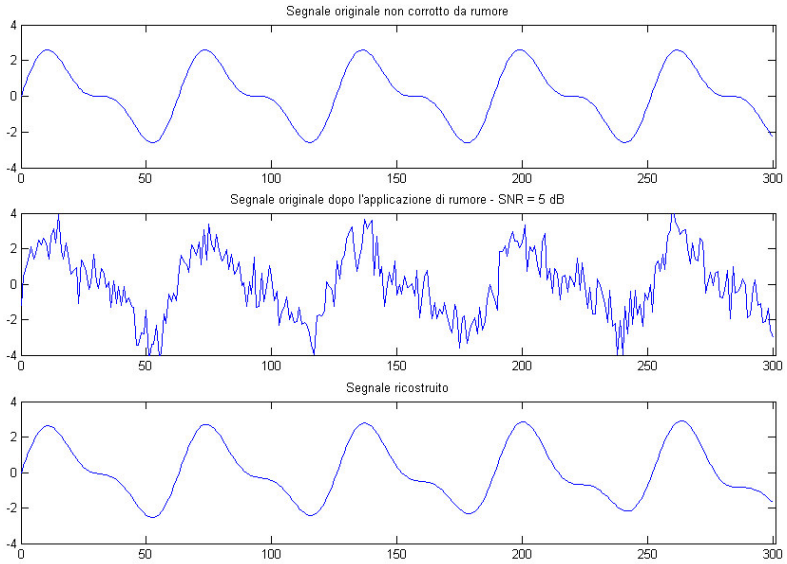

Figura 4.14. Ricostruzione di un segnale con due sinusoidi e SNR = 5 dB

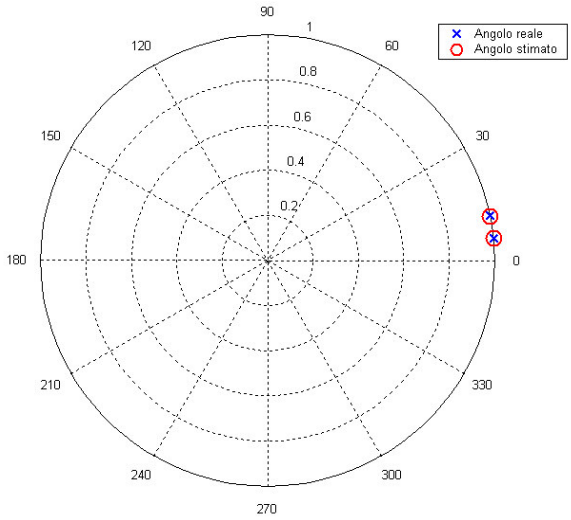

Figura 4.15. Pulsazione (reale e stimata) in gradi/sec sul cerchio unitario

Ampiezza		Frequenza (rad/sec)	
Reale	**Stimata**	**Reale**	**Stimata**
2	2.0753	0.1	0.1004
1	0.9400	0.2	0.1979

In questo caso le ampiezze vengono ricostruite con errori relativi rispettivamente del 3.76% e del 6%, le frequenze con errori relativi dello 0.4% e dell 1.05%,
In pratica, MUSIC continua a dare informazioni utili anche in caso di rumore molto evidente.

4.5.5 Esempio (Smyth)

Si consideri il segnale composto da due sinusoidi

$$x(t) = 2\sin(0.1t) + \sin(0.2t)$$

Come visto sopra, con MUSIC tali componenti sinusoidali vengono ricostruite accettabilmente fino al SNR di 5 dB. Partiamo qui con un SNR di 15 dB e valutiamo gli effetti.

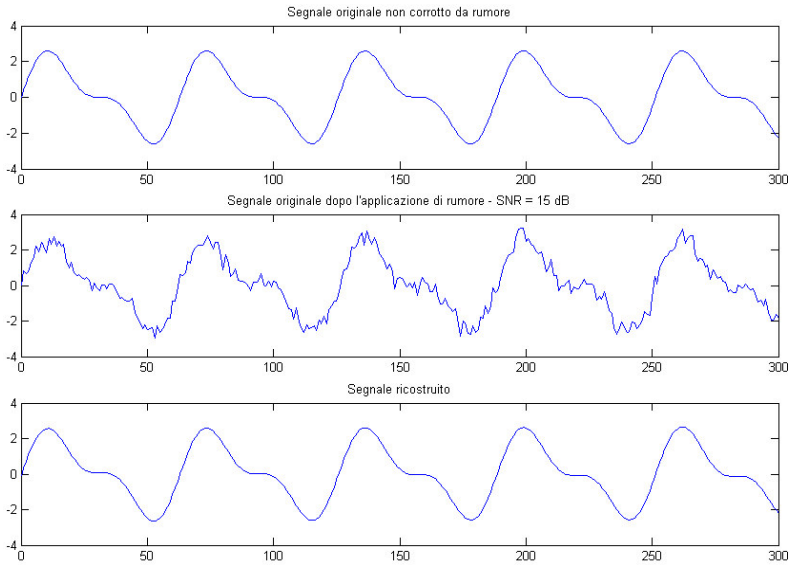

Figura 4.16. Ricostruzione di un segnale con due sinusoidi e SNR = 15 dB

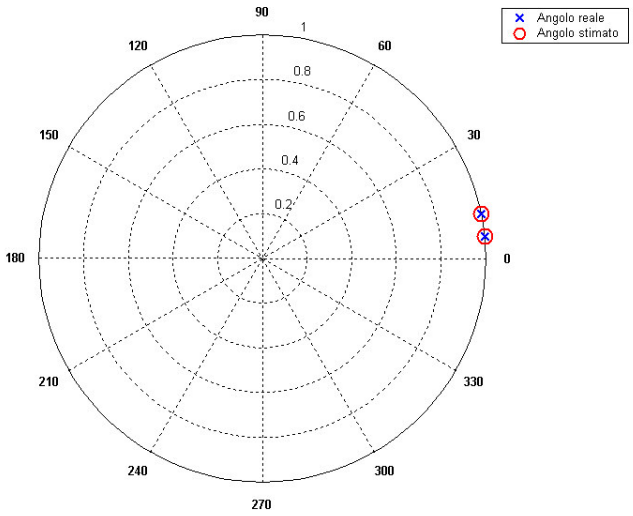

Figura 4.17. Pulsazione (reale e stimata) in gradi/sec sul cerchio unitario

Ampiezza		**Frequenza (rad/sec)**	
Reale	**Stimata**	**Reale**	**Stimata**
2	1.9879	0.1	0.1002
1	1.0242	0.2	0.1997

Si nota la notevole precisione con la quale le due sinusoidi vengono ricostruite, peraltro riscontrabile osservando il segnale ricostruito. Le due ampiezze sono ricostruite con errori percentuali rispettivamente dello 0.6% e del 2.42%; le ampiezze sono ricostruite con errori relativi dello 0.2% e dello 0.15%.

Diminuendo però il SNR fino a 10 dB, l'algoritmo di Smyth smette di fornire indicazioni utili, come evidenziato attraverso i grafici seguenti.

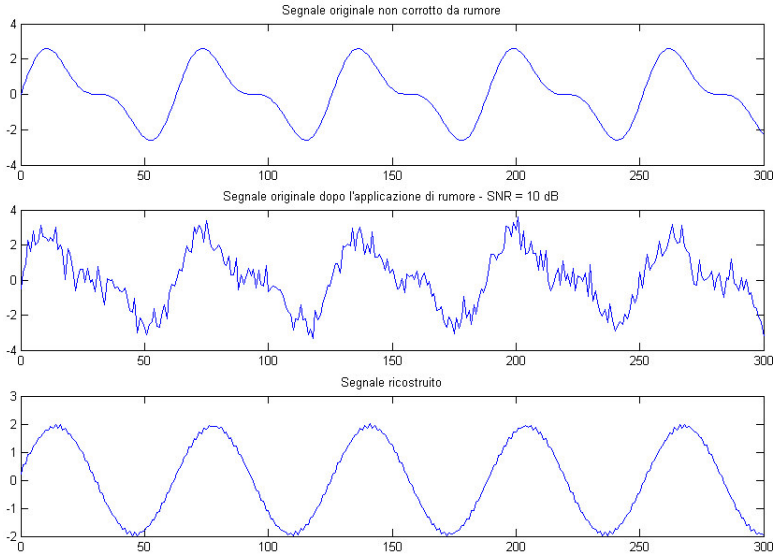

Figura 4.18. Ricostruzione di un segnale con due sinusoidi e SNR = 10 dB (in questo caso l'algoritmo di Smyth fallisce)

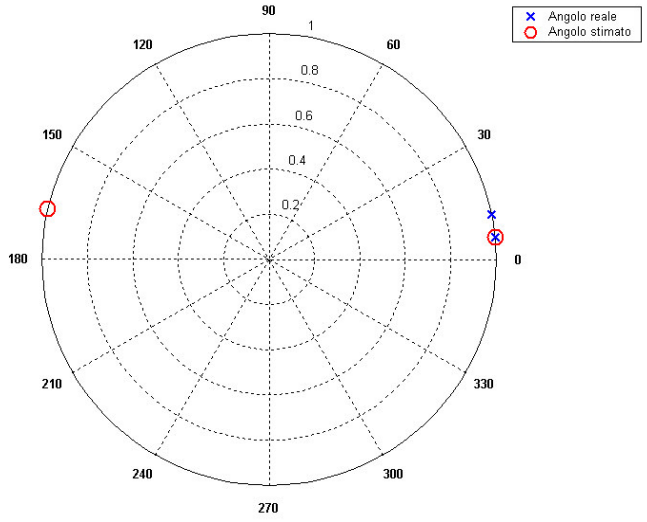

Figura 4.19. Pulsazione (reale e stimata) in gradi/sec sul cerchio unitario

In questo caso, quindi si può concludere che MUSIC ha dato prestazioni migliori dell'algoritmo di Smyth.

In generale, MUSIC fornisce stime più attendibili dell'algoritmo di Smyth, potendo gestire il parametro *M* e quindi considerando una sorta di mediazione del rumore nel tempo. Smyth ottiene prestazioni elevatissime fino ad un certo punto, quando improvvisamente smette di dare prestazioni

attendibili. MUSIC, invece, fornisce stime che vanno sempre più deteriorandosi se il parametro M si mantiene costante; tale parametro però può essere aumentato a piacere (a scapito comunque del tempo di elaborazione ma a vantaggio generalmente della qualità della stima) conferendo a MUSIC una robustezza difficile da migliorare.