UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO MATEMATICA

Università degli Studi di Padova

DIPARTIMENTO DI MATEMATICA "TULLIO LEVI-CIVITA"

Corso di Laurea in Matematica

Tesi di Laurea

Metodo di Prony e sua implementazione numerica

Relatore:
Prof. Alvise Sommariva

Laureanda:
Sara Albi
Matricola: 2068073

Anno Accademico 2025/2026

25 Settembre 2026

Indice

Introduzione

Il metodo di Prony fu sviluppato da Gaspard Riche de Prony, ingegnere, matematico e musicologo francese, nel 1795. Il proposito è di estrarre informazioni a partire da un segnale campionato uniformemente da cui è possibile stimare frequenze e ampiezze per ricostruire il segnale stesso tramite una serie di esponenziali complessi

$$f(t) = \sum_{j=1}^{n} \alpha_j \exp(\phi_j t), \quad \alpha_j, \phi_j \in \mathbb{C}.$$

L'obiettivo del Capitolo 1 è determinare i coefficienti α_j e gli esponenti ϕ_j, per $j = 1, \ldots, n$. In particolare, se è fissata la sparsità n, ovvero il numero n di componenti esponenziali con coefficienti non nulli, per ricostruire il segnale sono necessari $2n$ campionamenti equidistanti. Uno dei vantaggi è l'approssimazione del valore del segnale per qualsiasi istante di tempo t. Inoltre, in caso n sia molto inferiore al numero di campionamenti, si ha una compressione del segnale.

Nel Capitolo 2 viene introdotta l'implementazione numerica del metodo di Prony. Si descrive il principale ostacolo numerico del metodo originale e vengono proposti due metodi numerici che sono considerati delle varianti del metodo di Prony, ovvero i metodi ESPRIT e *Matrix Pencil* che sono sue realizzazioni numeriche particolarmente stabili.

Successivamente, nel Capitolo 3 vengono proposte delle implementazioni numeriche, sviluppate in ambiente MATLAB, dei metodi ESPRIT e Matrix Pencil. Vengono quindi presentati esempi numerici in cui vengono ricostruite combinazioni lineari di funzioni esponenziali attraverso i metodi precedentemente descritti. Dai test, si deduce che i metodi ESPRIT e Matrix Pencil sono dei validi sostituti numerici del metodo di Prony in quanto risolvono il problema della ricostruzione di un segnale dati alcuni campionamenti con errori relativi ed errori massimi assoluti molto piccoli.

Infine, nell'Appendice vengono riportati i codici MATLAB utilizzati nei test e viene illustrato uno storico esempio numerico di Cornelius Lanczos svolto nel 1956 che illustra la potenziale instabilità del metodo di Prony.

Capitolo 1

Il metodo di Prony

In questo Capitolo vengono presentate tre versioni equivalenti del metodo di Prony applicato a segnali non modulati, ovvero con ampiezza e frequenza indipendenti dal tempo. L'obiettivo comune alle tre versioni è quello di determinare i parametri tali per cui è possibile definire una funzione, espressa come somma di esponenziali, che ricostruisce il segnale di partenza.

Vogliamo rappresentare il segnale $f(t)$ tramite somma di funzioni esponenziali, quindi nella forma

$$f(t) = \sum_{j=1}^{n} \alpha_j \exp(\phi_j t), \quad \alpha_j, \phi_j \in \mathbb{C}, \tag{1.1}$$

dove assumiamo nota la sparsità $n \in \mathbb{N}$. Chiameremo i coefficienti α_j anche *ampiezze complesse* e gli esponenti ϕ_j *frequenze complesse*. Di fatto, le ampiezze reali sono $|\alpha_j|$ e le frequenze reali sono $\Im(\phi_j)$, dove con $\Im$ denoteremo la parte immaginaria di un numero complesso.

Già nel 1795 Prony dimostrò come, nota la sparsità n, sia possibile determinare i coefficienti $\{\alpha_j\}$ e gli esponenti $\{\phi_j\}$, $j = 1, \dots, n$, a partire da soli $2n$ campioni equidistanti. Per determinare la numerosità dei campioni si considera quanto enunciato nel Teroema di Nyquist-Shannon. Per una trattazione più esaustiva, si rimanda a [4, p. 220].

Teorema 1.0.1. *Dato un segnale continuo $x(t)$ la cui banda di frequenze sia limitata dalla frequenza f_M e dato $n \in \mathbb{Z}$, è possibile ricostruire in modo univoco il segnale a partire dai suoi campioni $s(n\Delta t)$ presi a frequenze $f_s = \frac{1}{\Delta t}$ se $f_s \geq f_M$.*

Nel caso in cui rispettiamo le ipotesi del Teorema 1.0.1 appena enunciato, evitiamo il fenomeno dell'*aliasing*, ovvero un fenomeno di distorsione che si verifica nel passaggio da un segnale continuo a un segnale discreto.

Per passare dalla descrizione del segnale nel tempo continuo al tempo discreto, introduciamo, dunque, lo schema di campionamento equispaziato. Sia $\Delta \in \mathbb{R} \setminus \{0\}$ il passo di campionamento tale che $|\Im(\phi_j)| < \pi/|\Delta|$. Questo vincolo è necessario in quanto impedisce il fenomeno di sovrapposizione delle frequenze. Infatti, la funzione esponenziale complessa $\exp : \mathbb{C} \to \mathbb{C}$ è periodica di periodo $2\pi i$ e dunque $\exp(\Delta\phi_j) = \exp(\Delta\phi_j + 2\pi i k)$, $k \in \mathbb{Z}$.

Perciò, imponendo che $\Im(\phi_j)\Delta \in (-\pi, \pi)$, si garantisce che ad ogni $\exp(\Delta\phi_j)$ sia associata in modo univoco la frequenza ϕ_j.

1.1 L'approccio originale

Ripercorriamo il procedimento originale proposto da Prony.

Denotiamo il segnale campionato con

$$f_k := f(k\Delta), \quad k = 0, \dots, 2n-1, \tag{1.2}$$

assumendo che la prima osservazione sia campionata al tempo 0, e introduciamo nuove variabili Φ_j definite da

$$\Phi_j := \exp(\phi_j \Delta), \quad j = 1, \dots, n. \tag{1.3}$$

Definiamo il polinomio $\mathcal{P}_n$, detto *polinomio di Prony*,

$$\mathcal{P}_n(z) := \prod_{j=1}^{n}(z - \Phi_j) = \sum_{i=0}^{n} b_{n-i} z^i, \quad \text{fissato } b_0 = 1, \tag{1.4}$$

le cui radici sono evidentemente Φ_j e i cui coefficienti b_{n-i}, $i = 0, \dots, n-1$, sono incogniti e dipendono da Φ_j. Conseguentemente, abbiamo che il polinomio di Prony valutato nelle radici Φ_j è pari a zero, ovvero

$$\sum_{i=0}^{n} b_{n-i} \Phi_j^i = 0. \tag{1.5}$$

Per unicità di notazione con quanto descritto successivamente nella Sezione 1.2, denotiamo in ordine decrescente i coefficienti del polinomio (1.4).

Poiché vogliamo determinare la funzione

$$f(t) = \sum_{j=1}^{n} \alpha_j \exp(\phi_j t)$$

che ricostruisce il segnale, in virtù di (1.1), (1.2) e (1.3), il vincolo $f_k = f(k\Delta)$, $k = 0, \dots, 2n-1$, diventa

$$f_k = \sum_{j=1}^{n} \alpha_j \Phi_j^k, \quad k = 0, \dots, 2n-1, \tag{1.6}$$

in quanto si richiede $f_k = \sum_{j=1}^{n} \alpha_j \exp(\phi_j k\Delta) = \sum_{j=1}^{n} \alpha_j \Phi_j^k$.

Da (1.6) e (1.5), per $s = 0, \dots, n-1$, otteniamo

$$\sum_{i=0}^{n} b_{n-i} f_{s+i} = \sum_{i=0}^{n} b_{n-i} \Big(\sum_{j=1}^{n} \alpha_j \Phi_j^{s+i}\Big) = \sum_{j=1}^{n} \alpha_j \Phi_j^s \Big(\sum_{i=0}^{n} b_{n-i} \Phi_j^i\Big) = 0.$$

Dunque, considerando il primo e l'ultimo termine della catena di uguaglianze sopra, isolando il termine di indice n della sommatoria e spostandolo a sinistra dell'uguaglianza,

otteniamo

$$\sum_{i=0}^{n-1} b_{n-i} f_{s+i} = -f_{s+n}, \quad s = 0, \dots, n-1, \tag{1.7}$$

poiché in (1.4) avevamo imposto $b_0 = 1$. L'insieme delle equazioni (1.7) costituisce il seguente sistema lineare

$$\begin{cases} f_0 b_n + f_1 b_{n-1} + ... + f_{n-1} b_1 = -f_n \\ f_1 b_n + f_2 b_{n-1} + ... + f_n b_1 = -f_{n+1} \\ \vdots \\ f_{n-1} b_n + f_n b_{n-1} + ... + f_{2n-2} b_1 = -f_{2n-1} \end{cases}, \tag{1.8}$$

che in forma matriciale si presenta come

$$\begin{pmatrix} f_0 & f_1 & \cdots & f_{n-1} \\ f_1 & f_2 & \cdots & f_n \\ \vdots & \vdots & & \vdots \\ f_{n-1} & f_n & \cdots & f_{2n-2} \end{pmatrix} \begin{pmatrix} b_n \\ b_{n-1} \\ \vdots \\ b_1 \end{pmatrix} = \begin{pmatrix} -f_n \\ -f_{n+1} \\ \vdots \\ -f_{2n-1} \end{pmatrix}, \tag{1.9}$$

dove la matrice del sistema è quadrata di ordine n, simmetrica e ha elementi costanti sulle antidiagonali. In particolare è una matrice di Hankel. Possiamo risolvere questo sistema lineare (n incognite b_{n-i} con $i = 0, \dots, n-1$ e n equazioni) e determinare i coefficienti b_{n-i}.

Di conseguenza, è possibile fattorizzare il polinomio di Prony $\mathcal{P}_n$ e ricavare le sue radici Φ_j. Inoltre, dalla definizione di Φ_j introdotta in (1.3), è sufficiente invertire la formula per ottenere

$$\phi_j = \frac{\log \Phi_j}{\Delta}, \quad j = 1, \dots, n. \tag{1.10}$$

Infine, sostituendo l'espressione appena trovata per gli esponenti ϕ_j nel vincolo di interpolazione della funzione nei campionamenti iniziali (1.6) e risolvendo le prime n equazioni del sistema sovradeterminato ottenuto, si ricavano anche i coefficienti α_j.

Abbiamo raggiunto lo scopo e possiamo costruire con i parametri determinati la funzione $f(t)$ che interpola i segnali campionati.

1.2 Il procedimento di Padé

Esaminiamo di seguito il procedimento descritto in [7] e introduciamo alcune definizioni preliminari.

Definizione 1.2.1. Si dice *approssimante di Padé* $[M/N]_f$ di $f(z)$ il rapporto tra due polinomi in z, $P_M(z)$ e $Q_N(z)$ di grado rispettivamente M e N, che ha le stesse $M + N$

derivate di $f(z)$ in $z = 0$. Si scrive quindi

$$[M, N]_f := \frac{P_M(z)}{Q_N(z)} = f(z) + O(M + N + 1)$$

e, da questo punto di vista, gli approssimanti di Padé sono costruiti con i coefficienti della serie di Taylor. Si osserva infatti che, dalla formula precedente, si ricava la scrittura

$$Q_N(z)f(z) - P_M(z) = O(M + N + 1)$$

da cui segue che i coefficienti dei polinomi $P_M(z)$ e $Q_N(z)$ sono determinati da un insieme di equazioni lineari a partire dai primi $M + N + 1$ coefficienti della serie di Taylor.

Definizione 1.2.2. Sia x_n, $n \in \mathbb{N}$, una successione di numeri complessi. Si dice *trasformata $\mathscr{Z}$ (unilatera)* di $\{x_n\}_n$ la serie formale di potenze $X(z) = \mathscr{Z}\{x_n\}(z) = \sum_{n=0}^{\infty} x_n z^{-n}$, per $z \in \mathbb{C}$.

Nota 1.2.1. *Nell'ambito della teoria dei segnali, la successione $\{x_n\}_n$ rappresenta tipicamente i campionamenti di un segnale casuale $f : \mathbb{R} \to \mathbb{C}$.*

Descriviamo il procedimento. Consideriamo $F(z)$ la trasformata $\mathscr{Z}$ della successione (infinita) di campioni $\{f_k\}_k$

$$F(z) := \mathscr{Z}\{f(k\Delta)\} = \sum_{k=0}^{\infty} f_k z^{-k} = f_0 + f_1 z^{-1} + \ldots + f_{2n-1} z^{-(2n-1)} + \ldots \quad . \tag{1.11}$$

Consideriamo inoltre la trasformata $\mathscr{Z}$ della successione $\{\exp(\phi_j k\Delta)\}_k$, ovvero

$$\mathscr{Z}\{e^{(\phi_j k\Delta)}\}(z) = \sum_{k=0}^{\infty} \exp(\phi_j k\Delta) z^{-k} = \sum_{k=0}^{\infty} \Phi_j^k z^{-k} = \sum_{k=0}^{\infty} \left(\frac{\Phi_j}{z}\right)^k = \frac{1}{1 - \frac{\Phi_j}{z}} = \frac{z}{z - \Phi_j}.$$

dove abbiamo usato il prolungamento analitico della serie geometrica.

Definiamo la funzione razionale nella variabile z

$$r_{n,n}(z) := \frac{a_n z^n + a_{n-1} z^{n-1} + \ldots + a_1 z}{z^n + b_1 z^{n-1} + \ldots + b_{n-1} z + b_n}. \tag{1.12}$$

Affinché $r_{n,n}$ sia una approssimante di Padé di $F(z)$, si verifica che basta imporre l'uguaglianza tra i primi $2n$ termini di

$$\begin{aligned} &a_n z^n + a_{n-1} z^{n-1} + \ldots + a_1 z = \\ &\quad = (z^n + b_1 z^{n-1} + \ldots + b_{n-1} z + b_n)(f_0 + f_1 z^{-1} + \ldots + f_{2n-1} z^{-(2n-1)} + \ldots), \end{aligned} \tag{1.13}$$

ovvero l'uguaglianza tra i coefficienti delle potenze da z^n fino a $z^{-(n-1)}$.

Questo comporta che

$$\begin{cases} f_0 = a_n \\ f_0 b_1 + f_1 = a_{n-1} \\ \vdots \\ f_0 b_{n-1} + f_1 b_{n-2} + \ldots + f_{n-2} b_1 + f_{n-1} = a_1 \end{cases} \tag{1.14}$$

$$\begin{cases} f_0 b_n + f_1 b_{n-1} + \ldots + f_{n-1} b_1 + f_n = 0 \\ f_1 b_n + f_2 b_{n-1} + \ldots + f_n b_1 + f_{n+1} = 0 \\ \vdots \\ f_{n-1} b_{n-1} + f_n b_{n-1} + \ldots + f_{2n-2} b_1 + f_{2n-1} = 0. \end{cases} \tag{1.15}$$

È immediato osservare che (1.15) coincide con (1.7) ottenuto nella Sezione 1.1, riscrivibile mediante un sistema lineare definito da una opportuna matrice di Hankel.

Dunque, possiamo concludere che i termini $\{b_{n-i}\}_i$, $\{\Phi_j\}_j$ e, di conseguenza, $\{\phi_j\}_j$ determinati dal procedimento di Padé sono gli stessi di quelli determinati dal metodo originale. Possiamo pertanto ricavare i coefficienti $\{\alpha_j\}_j$ di (1.6).

Tuttavia, invece di procedere come nella Sezione precedente e risolvere le prime n equazioni di (1.6), possiamo sfruttare la proprietà delle funzioni razionali di Padé. Noti $b_1, \ldots, b_n$, possiamo tramite il sistema (1.14) ricavare gli $a_1, \ldots, a_n$ e dunque costruire interamente la funzione razionale (1.12). Dividiamo per z e fattorizziamo il denominatore come in (1.4), così otteniamo

$$\frac{r_{n,n}(z)}{z} = \frac{1}{z} \cdot \frac{a_n z^n + a_{n-1} z^{n-1} + \ldots + a_1 z}{z^n + b_1 z^{n-1} + \ldots + b_{n-1} z + b_n} = \frac{a_n z^{n-1} + a_{n-1} z^{n-2} + \ldots + a_1}{\prod_{j=1}^{n} (z - \Phi_j)}. \tag{1.16}$$

Poiché si tratta di una funzione razionale con poli semplici distinti, possiamo riscivere l'ultimo membro di (1.16) in fratti semplici

$$\frac{r_{n,n}(z)}{z} = \sum_{j=i}^{n} \frac{\alpha_j}{z - \Phi_j}. \tag{1.17}$$

Infine, possiamo ricavare i termini $\Phi_1, \ldots, \Phi_n$ e quindi i parametri $\phi_1, \ldots, \phi_n$ delle funzioni esponenziali.

Pur senza dimostrarlo dettagliatamente, Weiss e Mc Donough asseriscono che i termini $\{\alpha_j\}_{j=1,\ldots,n}$ sono esattamente le ampiezze in (1.1). Per maggiori dettagli e un interessante esempio numerico, si veda [7, p. 148].

Abbiamo raggiunto quindi i medesimi risultati della Sezione 1.1 e mostrato che il metodo via approssimanti di Padé è teoricamente equivalente a quello di Prony.

1.3 Osservazioni

Alternativamente alla trasformata $\mathscr{Z}$ della successione di campioni f_k, considerando la serie formale di potenze

$$F(z) = \sum_{k=0}^{\infty} f_k z^k = f_0 + f_1 z + \ldots + f_{2n-1} z^{2n-1} + \ldots \quad , \tag{1.18}$$

possiamo comunque ottenere risultati analoghi a quelli della Sezione precedente. Sostituendo in (1.18) l'espressione dei dati campionati $f_k = \sum_{j=1}^{n} \alpha_j \Phi^k$ otteniamo

$$F(z) = \sum_{k=0}^{\infty} f_k z^k = \sum_{k=0}^{\infty} (\sum_{j=1}^{n} \alpha_j \Phi_j^k) z^k = \sum_{j=1}^{n} \alpha_j (\sum_{k=0}^{\infty} \Phi_j^k z^k) = \sum_{j=1}^{n} \frac{\alpha_j}{1 - \Phi_j z}.$$

Da questa espressione deduciamo che $F(z)$ è una funzione razionale definita dal rapporto di due polinomi. In particolare, calcolando i minimi comuni multipli, si evince che quello a numeratore, che chiamiamo P_{n-1}, ha grado $n-1$ mentre quello a denominatore, che denotiamo con Q_n, ha grado n. Quindi possiamo scrivere $F(z)$ come

$$F(z) = \frac{P_{n-1}(z)}{Q_n(z)}. \tag{1.19}$$

Sfruttando la proprietà degli approssimanti di Padé, possiamo ricostruire la funzione $F(z)$ calcolando l'approssimante di Padé $[n-1/n]_F$ fissati soli i coefficienti $f_0, \ldots, f_{2n-1}$. Così facendo possiamo calcolare i Φ_j dal polinomio al denominatore

$$Q_n(z) = \prod_{j=1}^{n} (1 - \Phi_j z) = b_n z^n + \ldots + b_1 z + 1, \tag{1.20}$$

in quanto i Φ_j corrispondono agli inversi dei poli dell'approssimante di Padé. Successivamente possiamo calcolare le ampiezze α_j dalla decomposizione in fratti semplici di $[n-1/n]_F$. Osserviamo però che la serie di potenze (1.18) e la trasformata $\mathscr{Z}$ di f_k (1.11) sono tali che

$$\mathscr{Z}\{f_k\}(z) = F(1/z) \tag{1.21}$$

Con questa osservazione, non è difficile dedurre che il polinomio di Prony (1.4), definito nella Sezione 1.1, è legato al denominatore di $[n-1/n]_F$, $Q_n(z)$, e, in particolare, vale

$$z^n F(1/z) = Q_n(z), \tag{1.22}$$

dove con F intendiamo la serie di potenze in (1.18).

1.4 Un procedimento basato sulle matrici di Hankel

Descriviamo, infine, un procedimento basato sull'algebra lineare che viene proposto in [1]. A partire dal modello precedentemente descritto in cui la funzione da ricostruire è

$$f(t) = \sum_{j=1}^{n} \alpha_j \exp(\phi_j t), \quad \alpha_j, \phi_j \in \mathbb{C},$$

avendo a disposizione i dati campionati a intervalli equispaziati, come descritto in (1.2) e (1.3), intendiamo determinare le ampiezze α_j e le frequenze ϕ_j.

Attraverso i campioni f_k, $k = 0, \dots, 2n-1$, costruiamo la matrice di Hankel $H_n^{(m)} \in \mathbb{C}^{n\times n}$

$$H_n^{(m)} := (f_{m+i+j-2})_{i,j=1}^{n} = \begin{pmatrix} f_m & f_{m+1} & \cdots & f_{m+n-1} \\ f_{m+1} & f_{m+2} & \cdots & f_{m+n} \\ \vdots & \vdots & \vdots & \vdots \\ f_{m+n-1} & f_{m+n} & \cdots & f_{m+2n-2} \end{pmatrix}, \quad m \in \mathbb{N}. \tag{1.23}$$

Osserviamo che, per costruzione, la matrice $H_n^{(m)} \in \mathbb{C}^{n\times n}$ è simmetrica e gli elementi sulle antidiagonali sono costanti. Inoltre, la componente con indici di riga i e di colonna j dipende solo dalla somma $i+j$. Sostituendo in $H_n^{(m)}$ l'espressione (1.6) per i campionamenti equispaziati f_k, si trova la seguente fattorizzazione

$$H_n^{(m)} = V_n D_\alpha D_\Phi^m V_n^T. \tag{1.24}$$

dove

$$V_n = (\Phi_j^{i-1})_{i,j=1}^{n} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \Phi_1 & \Phi_2 & \cdots & \Phi_n \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_1^{n-1} & \Phi_2^{n-1} & \cdots & \Phi_n^{n-1} \end{pmatrix},$$

$$D_\alpha = \mathrm{diag}(\alpha_1, ..., \alpha_n) = \begin{pmatrix} \alpha_1 & 0 & \cdots & 0 \\ 0 & \alpha_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \alpha_n \end{pmatrix},$$

$$D_\Phi = \mathrm{diag}(\Phi_1, ..., \Phi_n) = \begin{pmatrix} \Phi_1 & 0 & \cdots & 0 \\ 0 & \Phi_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \Phi_n \end{pmatrix}.$$

La relazione (1.24) dipende unicamente dalla definizione dei segnali campionati $f_k = \sum_{j=1}^{n} \alpha_j \Phi_j^k$, $k = 0, \dots, 2n-1$.

Infatti, da

$$\begin{aligned}(V_n D_\alpha D_\Phi^m V_n^T)_{i,j} &= \sum_{k=1}^{n} (V_n)_{i,k}\alpha_k \Phi_k^m (V_n)_{k,j}^T \\ &= \sum_{k=1}^{n} \Phi_k^{i-1}\alpha_k \Phi_k^m \Phi_k^{j-1} \\ &= \sum_{k=1}^{n} \alpha_k \Phi_k^{m+i+j-2} = f_{m+i+j-2} \\ &= (H_n^{(m)})_{i,j}. \end{aligned} \tag{1.25}$$

si deduce che

$$V_n D_\alpha D_\Phi^m V_n^T = H_n^{(m)}.$$

Notiamo, inoltre, che la matrice di Hankel $H_n^{(m)}$ è invertibile in quanto lo sono tutte le matrici coinvolte nella sua fattorizzazione. V_n e V_n^T hanno rango pieno, dato che abbiamo assunto inizialmente le frequenze $\{\phi_j\}_j$, e quindi $\{\Phi_j\}_j$, tutte distinte. Anche D_α ha rango pieno, poiché se esistesse un indice $j = 1, \ldots, n$ tale per cui l'ampiezza α_j della j-esima componente del segnale fosse uguale a 0, allora sarebbe sufficiente non considerare quella componente. Infine, D_Φ, matrice diagonale, ha rango pieno in quanto nessuno dei suoi elementi diagonali può essere nullo, essendo definiti tramite la funzione esponenziale complessa.

Dalla fattorizzazione (1.24) della matrice $H_n^{(m)}$, possiamo determinare i $\{\Phi_j\}_{j=1,\ldots,n}$ e quindi le frequenze $\{\phi_j\}_{j=1,\ldots,n}$ nel modo seguente. Introduciamo le matrici

$$H_n^{(0)} = V_n D_\alpha V_n^T, \tag{1.26}$$

$$H_n^{(1)} = V_n D_\alpha D_\Phi V_n^T. \tag{1.27}$$

e risolviamo il problema *agli autovalori generalizzati*

$$H_n^{(0)} v_j = \lambda_j H_n^{(1)} v_j, \quad j = 1, \ldots, n \tag{1.28}$$

dove $\{\lambda_j\}_{j=1,\ldots,n}$ sono gli autovalori generalizzati e $\{v_j\}_{j=1,\ldots,n}$ i corrispettivi autovettori generalizzati. Per una trattazione generale più approfondita del problema agli autovalori generalizzati rimandiamo a [2, p. 497].

Da (1.28), sostituendo le espressioni precedentemente calcolate (1.26) e (1.27), troviamo

$$V_n D_\alpha V_n^T v_j = \lambda_j V_n D_\alpha D_\Phi V_n^T v_j. \tag{1.29}$$

Osserviamo che entrambi i membri dell'uguaglianza (1.29) hanno in comune i fattori $V_n D_\alpha$ a sinistra e V_n^T a destra e che il calcolo degli autovalori generalizzati per (1.28), ovvero il calcolo dei $\lambda_j \in \mathbb{C}$ tali che soddisfano $\det(H_n^{(0)} - \lambda_j H_n^{(1)}) = 0$, coincide quindi con il calcolo

degli autovalori della matrice D_Φ. Infatti, poiché, per le proprietà del determinante, vale

$$\begin{aligned}\det(H_n^{(0)} - \lambda_j H_n^{(1)}) &= \det(V_n D_\alpha V_n^T - \lambda_j V_n D_\alpha D_\Phi V_n^T) \\ &= \det(V_n D_\alpha (Id - \lambda_j D_\Phi) V_n^T) \\ &= \det(V_n)\det(D_\alpha)\det(Id - \lambda_j D_\Phi)\det(V_n^T),\end{aligned} \tag{1.30}$$

e poiché le matrici V_n e D_α sono non singolari, allora imporre che (1.30) sia uguale a 0 è equivalente a calcolare

$$\det(Id - \lambda_j D_\Phi) = 0,$$

ovvero a calcolare gli autovalori della matrice D_Φ. Considerando che la matrice D_Φ è diagonale, i suoi autovalori corrispondono esattamente ai suoi valori diagonali, ovvero $\Phi_1, \ldots, \Phi_n$, e quindi possiamo determinare le frequenze $\phi_1, \ldots, \phi_n$.

Successivamente, calcoliamo le ampiezze attraverso il sistema lineare (1.6).

Noto il numero n di componenti esponenziali, il sistema ha n equazioni linearmente indipendenti e possiamo risolverlo direttamente e determinare le ampiezze. Ad esempio, considerando i primi n campionamenti potremmo risolvere il sistema di n equazioni linearmente indipendenti

$$\begin{pmatrix} 1 & 1 & \cdots & 1 \\ \Phi_1 & \Phi_2 & \cdots & \Phi_n \\ \Phi_1^2 & \Phi_2^2 & \cdots & \Phi_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_1^{n-1} & \Phi_2^{n-1} & \cdots & \Phi_n^{n-1} \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_{n-1} \end{pmatrix}. \tag{1.31}$$

Così, abbiamo concluso il calcolo dei parametri necessari per ricostruire il segnale di partenza.

Capitolo 2

Implementazione del metodo di Prony e sue varianti

2.1 Realizzazione numerica del metodo di Prony

Nella Sezione 1.1 abbiamo descritto a livello teorico il metodo di Prony standard che chiameremo *originale* e che possiamo riassumere attraverso il seguente Algoritmo.

Assunto che sia $f(t) = \sum_{j=1}^{n} \alpha_j \exp(\phi_j t)$, con $\alpha_j, \phi_j \in \mathbb{C}$, $|\Im(\phi_j)| < \pi/|\Delta|$, mediante lo pseudocodice in Algoritmo 1 mostriamo come calcolare i parametri rilevanti $\{\alpha_j\}_j$, $\{\phi_j\}_j$ con $j = 1, \dots, n$.

Algoritmo 1 Algoritmo del metodo di Prony originale.

Input: n, $f_k = f(k\Delta)$, $k = 0, \dots, 2n-1$, Δ

1. Costruire la matrice $H_n^{(0)}$, risolvere il sistema dato da (1.9) e determinare $\{b_{n-i}\}_{i=0,\dots,n-1}$.
2. Fissato $b_0 = 1$, costruire il polinomio di Prony $\mathcal{P}_n(z)$ definito in (1.4) e calcolarne le radici $\{\Phi_j\}_{j=1,\dots,n}$.
3. Ricavare le frequenze $\{\phi_j\}_{j=1,\dots,n}$ con la formula (1.10).
4. Determinare le ampiezze $\{\alpha_j\}_{j=1,\dots,n}$ risolvendo il sistema retto dalla matrice di Vandermonde

$$\begin{pmatrix} 1 & 1 & \cdots & 1 \\ \Phi_1 & \Phi_2 & \cdots & \Phi_n \\ \Phi_1^2 & \Phi_2^2 & \cdots & \Phi_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_1^{n-1} & \Phi_2^{n-1} & \cdots & \Phi_n^{n-1} \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_{n-1} \end{pmatrix}$$

dato dalle prime n equazioni di (1.6).

Output: $\{\alpha_j\}_j$, $\{\phi_j\}_j$ con $j = 1, \dots, n$.

La principale difficoltà di questo Algoritmo è la ricerca di tutte le radici del polinomio di Prony $\mathcal{P}_n(z)$.

Per ovviare a ciò, affrontiamo il problema della ricerca delle radici di $\mathcal{P}_n(z)$ mediante l'uso della matrice compagna relativa al vettore dei coefficienti del polinomio $\mathcal{P}_n(z)$.

Definizione 2.1.1. Dato il vettore $\mathbf{p} = (p_0, \ldots, p_{n-1})^T$ dei coefficienti del polinomio monico $P(X) = p_0 + p_1 X + \ldots + p_{n-1}X^{n-1} + X^n$ si definisce la sua *matrice compagna* $C(P)$ come

$$C(P) = \begin{pmatrix} 0 & 0 & \cdots & 0 & -p_0 \\ 1 & 0 & \cdots & 0 & -p_1 \\ 0 & 1 & \cdots & 0 & -p_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -p_{n-1} \end{pmatrix}$$

Osservazione 2.1.1. *Gli autovalori della matrice compagna $C(P)$ corrispondono alle radici del relativo polinomio P, in quanto il polinomio caratteristico della matrice $C(P)$, il suo polinomio minimo e il polinomio P coincidono.*

Sia $\mathbf{b} = (b_{n-i})_{i=0,\ldots,n-1}$ il vettore dei coefficienti di $\mathcal{P}_n(z)$, ricavato dal primo passo dell'Algoritmo 1. Per l'osservazione precedente, trasformiamo il problema di ricerca degli zeri di $\mathcal{P}_n(z)$ in un problema di ricerca degli autovalori di $C(\mathcal{P})$, dove

$$C(\mathcal{P}) = \begin{pmatrix} 0 & 0 & \cdots & 0 & -b_n \\ 1 & 0 & \cdots & 0 & -b_{n-1} \\ 0 & 1 & \cdots & 0 & -b_{n-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -b_1 \end{pmatrix}.$$

Consideriamo, come nella Sezione (1.4), le matrici di Hankel formate dai dati in input

$$H_n^{(0)} = \begin{pmatrix} f_0 & f_1 & \cdots & f_{n-1} \\ f_1 & f_2 & \cdots & f_n \\ \vdots & \vdots & \vdots & \vdots \\ f_{n-1} & f_n & \cdots & f_{2n-2} \end{pmatrix}, \quad H_n^{(1)} = \begin{pmatrix} f_1 & f_2 & \cdots & f_n \\ f_2 & f_3 & \cdots & f_{n+1} \\ \vdots & \vdots & \vdots & \vdots \\ f_n & f_{n+1} & \cdots & f_{2n-1} \end{pmatrix}. \tag{2.1}$$

Abbiamo già osservato in precedenza, con il sistema (1.9), che $\mathbf{b}$ è l'unico vettore che combina linearmente le colonne di $H_n^{(0)}$ formando l'ultima colonna di $H_n^{(1)}$. Conseguentemente possiamo formulare la relazione

$$H_n^{(0)} C(\mathcal{P}) = H_n^{(1)}. \tag{2.2}$$

In letteratura, la formula (2.2) identifica la matrice compagna $C(\mathcal{P})$ come un *operatore di shift* tra $H_n^{(0)}$ e $H_n^{(1)}$.

Introduciamo alcune definizioni utili in seguito.

Definizione 2.1.2. Siano $A, B \in \mathbb{C}^{n\times n}$. L'insieme delle matrici della forma $A - \lambda B$, con $\lambda \in \mathbb{C}$, è detto *fascio di matrici A e B*, o *matrix pencil*. Gli *autovalori di un fascio di*

matrici A e B sono elementi dell'insieme $\lambda(A, B)$ definito da

$$\lambda(A, B) := \{\lambda \in \mathbb{C} : \quad \det(A - \lambda B) = 0\}.$$

Definizione 2.1.3. Sia $H \in \mathbb{C}^{n\times n}$. L'*insieme degli autovalori* di H è $\lambda(H)$ definito da

$$\lambda(H) := \{\lambda_j \in \mathbb{C} : \quad Hx_j = \lambda_j x_j, j = 1, \dots, n\}.$$

Tramite il seguente Lemma, partendo dalle proprietà dei fasci di matrici, mostriamo come determinare gli autovalori della matrice compagna $C(\mathcal{P})$.

Lemma 2.1.1. *Gli autovalori del fascio di matrici $H_n^{(1)}, H_n^{(0)}$ sono esattamente gli autovalori della matrice $C(\mathcal{P})$.*

Dimostrazione. Sia $\mathbf{x} \in \mathbb{C}^n$ un autovettore della matrice $C(\mathcal{P}) \in \mathbb{C}^{n\times n}$ relativo all'autovalore $\lambda \in \mathbb{C}$. Consideriamo la relazione precedentemente formulata

$$H_n^{(0)} C(\mathcal{P}) = H_n^{(1)}.$$

Valutiamo queste matrici nell'autovettore $\mathbf{x}$ e otteniamo

$$\begin{aligned} H_n^{(0)} C(\mathcal{P})\mathbf{x} &= H_n^{(1)}\mathbf{x}, \\ H_n^{(0)}(C(\mathcal{P})\mathbf{x}) - H_n^{(1)}\mathbf{x} &= 0, \\ H_n^{(0)}(\lambda\mathbf{x}) - H_n^{(1)}\mathbf{x} &= 0, \\ [\lambda H_n^{(0)} - H_n^{(1)}]\mathbf{x} &= 0. \end{aligned}$$

Abbiamo quindi mostrato che $\lambda(C(\mathcal{P})) = \lambda(H_n^{(1)}, H_n^{(0)})$. □

Dunque, come si evince da [6, Cap. 3], con quanto appena descritto possiamo, attraverso un metodo numericamente stabile, ricavare gli zeri del polinomio di Prony originale cercati al secondo passo dell'Algoritmo 1.

Nel concreto, raramente è nota a priori la sparsità n della combinazione lineare delle componenti esponenziali con cui vogliamo ricostruire la funzione f. Tuttavia è spesso noto un limite superiore L, tale che $n \leq L \leq N$, dove $2N$ è il numero di segnali campionati da cui abbiamo selezionato i *primi* $2n$ per l'esecuzione del metodo di Prony.

In [6, Cap. 2] è stato dimostrato che esistono essenzialmente tre modi di utilizzare i segnali campionati nell'Algoritmo, nel caso in cui n non sia nota.

Il caso più semplice è quello in cui non si considera l'informazione del limite superiore L e si costruisce una matrice di Hankel quadrata $H_N^{(0)} = (f_{i+j-2})_{i,j=1}^{N}$, così da usare tutti i dati disponibili. Tuttavia, questa scelta comporta in generale un eccessivo costo computazionale.

Un'altra opzione sarebbe considerare una matrice di Hankel quadrata, di dimensioni più piccole, $H_{L+1}^{(0)} = (f_{i+j-2})_{i,j=1}^{L+1}$, ma in questo modo non si stanno usando tutti i dati

e dunque si ottengono dei risultati poco accurati. Il compromesso ideale è costruire una matrice rettangolare $H = H^{(0)}_{2N-L,L} \in \mathbb{C}^{(2N-L)\times L}$ con $H_{i,j} = f_{i+j-2}$, che utilizza tutti i dati e al contempo preserva la stabilità, tuttavia necessitando di un'analisi più complicata.

Entriamo nel dettaglio di quest'ultimo approccio, basato su una matrice rettangolare, che estende il metodo di Prony ai casi in cui il numero esatto di componenti non è noto a priori.

Se $L > n$, denotiamo con $\mathbf{q} \in \mathbb{C}^L$ il vettore che risolve (in qualche senso da definire) $H^{(0)}_{2N-L,L}\mathbf{q} = -(f_{L+i-1})_{i=1}^{2N-L}$, ovvero

$$\begin{pmatrix} f_0 & f_1 & \cdots & f_{L-1} \\ f_1 & f_2 & \cdots & f_L \\ \vdots & \vdots & & \vdots \\ f_{2N-L-1} & f_{2N-L-2} & \cdots & f_{2N-2} \end{pmatrix} \begin{pmatrix} q_0 \\ q_1 \\ \vdots \\ q_{L-1} \end{pmatrix} = - \begin{pmatrix} f_L \\ f_{L+1} \\ \vdots \\ f_{2N-1} \end{pmatrix}. \tag{2.3}$$

Attraverso il vettore $\mathbf{q}$, soluzione del sistema sopra (2.3), possiamo costruire $\mathcal{Q}(z)$ il *polinomio di Prony correlato* e, fissando $q_L = 1$, definiamo

$$\mathcal{Q}(z) = \sum_{i=0}^{L} q_i z^i.$$

Per il Teorema fondamentale dell'algebra, vale la fattorizzazione di $\mathcal{Q}(z)$ nel prodotto di L fattori lineari $\mathcal{Q}(z) = \prod_{j=1}^{L}(z - \lambda_j)$, dove ogni λ_j è una radice di $\mathcal{Q}(z)$.

In particolare, otteniamo che tutti gli zeri del polinomio di Prony $\mathcal{P}_n(z)$, ovvero le informazioni di cui abbiamo bisogno per ricostruire il segnale, sono conservati in $\mathcal{Q}(z)$. Ciò deriva dal fatto che la matrice rettangolare $H^{(0)}_{2N-L,L}$ non è a rango massimo in quanto viene costruita partendo dai campioni del segnale, il quale dipende da n componenti e, per l'assunzione iniziale, il rango di $H^{(0)}_{2N-L,L}$, ovvero n, è inferiore a L. Dunque il sistema lineare che determina i coefficienti di $\mathcal{Q}(z)$ ammette infinite soluzioni. In altre parole, la soluzione $\mathbf{q}$ del sistema (2.3) non è unica bensì dipende da $L - n$ parametri. Essendo $\mathbf{q}$ il vettore dei coefficienti del polinomio $\mathcal{Q}(z)$, allora quest'ultimo non è unico. Fissato un vettore dei coefficienti, possiamo fattorizzare $\mathcal{Q}(z)$ come

$$\mathcal{Q}(z) = \prod_{j=1}^{n}(z - \Phi_j)\mathcal{R}(z) = \mathcal{P}_n(z)\mathcal{R}(z),$$

dove $\mathcal{R}(z)$ è un polinomio, che possiamo chiamare *residuo*, di grado $L - n$ i cui coefficienti corrispondono agli $L - n$ parametri del vettore scelto. Al variare di $\mathbf{q}$, le radici di $\mathcal{P}_n(z)$ rimangono invariate mentre le radici di $\mathcal{R}(z)$ variano a seconda della combinazione degli $L - n$ parametri scelti.

Nonostante ciò, è possibile adattare l'equazione (2.2), usata in precedenza, al caso rettangolare nel seguente modo.

Definiamo le due matrici rettangolari analoghe a (2.1) come

$$H^{(m)}_{2N-L,L} := (f_{m+i+j-2})^{2N-L,L}_{i,j=1}, \quad m = 0, 1.$$

La relazione, analoga alla (2.2), tra queste due matrici rettangolari è

$$H^{(0)}_{2N-L,L} C(\mathcal{Q}) = H^{(1)}_{2N-L,L}.$$

Possiamo estendere il Lemma (2.1.1) al caso di matrici rettangolari, essendo la dimostrazione simile alla precedente. A tal proposito, è sufficiente sostituire l'autovettore $\mathbf{x}$ con il vettore singolare destro per mostrare che possiamo determinare gli autovalori di $C(\mathcal{Q})$ risolvendo il *problema del fascio di matrici rettangolare*, o *rectangular matrix pencil problem*, dato da

$$H^{(1)}_{2N-L,L}\mathbf{x} = \lambda H^{(0)}_{2N-L,L}\mathbf{x}. \tag{2.4}$$

Riassumendo, nel caso in cui è noto il limite superiore $L \leq N$, costruiamo una matrice di Hankel rettangolare. Possiamo, poi, scegliere se fattorizzarla mediante la decomposizione QR oppure mediante la decomposizione a valori singolari. Il primo approccio conduce al Metodo del Matrix Pencil che verrà descritto nella Sezione (2.3), mentre il secondo conduce al Metodo ESPRIT che verrà descritto nella Sezione (2.2).

2.2 Metodo ESPRIT

Il metodo ESPRIT (acronimo di Estimation of Signal Parameters via Rotational Techniques) è una variante del metodo di Prony. Il suo vantaggio consiste nell'evitare di calcolare i coefficienti del polinomio di Prony e le sue radici, attraverso la risoluzione di un problema agli autovalori generalizzati.

In questa Sezione, adottiamo una notazione di tipo MATLAB, in cui con `m:n` indichiamo il vettore `[m,m+1,...,n-1,n]`.

Supponiamo che sia noto il limite superiore della sparsità L, e che $H_{2N-L,L+1}$ sia la matrice di Hankel generata dai dati campionati $f_0, \ldots, f_{2N-1}$, fattorizzata tramite *SVD*,

$$H_{2N-L,L+1} = U_{2N-L} D_{2N-L,L+1} W_{L+1}, \tag{2.5}$$

dove

- U_{2N-L} e W_{L+1} sono matrici unitarie, di dimensioni rispettivamente $2N-L$ e $L+1$,
- $D_{2N-L,L+1}$ è la matrice (rettangolare) le cui entrate diagonali sono i valori singolari di $H_{2N-L,L+1}$, che denotiamo con $\sigma_1, \ldots, \sigma_{L+1}$.

Possiamo riordinare le colonne di U_{2N-L} e le righe di W_{L+1} in modo tale che i valori

singolari siano disposti in ordine non crescente. Inoltre, possiamo selezionare le sottomatrici

$$D_{2N-L,n} := D_{2N-L,L+1}(1:2N-L, 1:n) = \begin{pmatrix} \mathrm{diag}(\sigma_j)_{j=1}^{n} \\ \mathbf{0}_{2N-L-n,n} \end{pmatrix},$$
$$W_{n,L+1} := W_{L+1}(1:n, 1:L+1),$$

dato che, come dimostrato in [6, pp. 20-22], il rango della matrice (2.5) coincide con la sparsità cercata. Per questo motivo, in aritmetica esatta, si hanno i valori singolari $\sigma_{n+1} = \sigma_{n+2} = \ldots = \sigma_{L+1} = 0$.

Dunque, possiamo scrivere una versione semplificata della fattorizzazione (2.5)

$$H_{2N-L,L+1} = U_{2N-L} D_{2N-L,n} W_{n,L+1}.$$

Definiamo anche

$$W_{n,L}^{(m)} := W_{n,L+1}(1:n, 1+m:L+m), \quad m = 0, 1, \tag{2.6}$$

tramite cui possiamo scrivere

$$H_{2N-L,L}^{(m)} = U_{2N-L} D_{2N-L,n} W_{n,L}^{(m)}, \quad m = 0, 1.$$

Da quanto osservato finora, possiamo dedurre che il problema agli autovalori generalizzati del fascio di matrici $zH_{2N-L}^{(0)} - H_{2N-L}^{(1)}$ è equivalente al problema agli autovalori generalizzati del fascio di matrici

$$zD_{2N-L,n} W_{n,L}^{(0)} - D_{2N-L,n} W_{n,L}^{(1)}, \tag{2.7}$$

poiché U_{2N-L} è unitaria. Moltiplicando a destra il problema (2.7) trasposto per la matrice

$$\begin{pmatrix} \mathrm{diag}(\sigma_j^{-1})_{j=1}^{n} \\ \mathbf{0}_{2N-L-n,n} \end{pmatrix},$$

otteniamo il problema agli autovalori generalizzati del fascio di matrici

$$z{W_{n,L}^{(0)}}^T - {W_{n,L}^{(1)}}^T, \tag{2.8}$$

che ha gli stessi autovalori (eccetto quelli nulli) di (2.7).

Infine, determiniamo gli autovalori precedentemente indicati con z come autovalori della matrice definita da

$$Z_n^{SVD} := ({W_{n,L}^{(0)}}^T)^+ {W_{n,L}^{(1)}}^T, \tag{2.9}$$

dove con V^+ denotiamo la matrice pseudoinversa di Moore-Penrose di V.

In conclusione, possiamo riassumere il metodo ESPRIT tramite l'Algoritmo 2.

Algoritmo 2 Algoritmo ESPRIT.

Input: $f_0, \ldots, f_{2N-1}$, $L \geq n$, Δ, ϵ.

1. Eseguire la decomposizione a valori singolari come in (2.5). Determinare il rango della matrice $H_{2N-L,L+1}$ e costruire le matrici $W_{n,L}^{(0)}$ e $W_{n,L}^{(1)}$ come in (2.6).
2. Calcolare gli n autovalori di Z_n^{SVD} definita come in (2.9) e ricavare le frequenze ϕ_j con la formula (1.10).
3. Calcolare le ampiezze α_j risolvendo il sistema (sovradeterminato) $\sum_{j=1}^{n} \alpha_j \lambda_j^k = f_k$, $k = 0, \ldots, 2N-1$, dove $\{\lambda_j\}_j$ sono gli autovalori calcolati al passo 2.

Output: $\{\alpha_j\}_j$, $\{\phi_j\}_j$ con $j = 1, \ldots, n$.

Osserviamo che il valore di soglia ϵ, richiesto in input dall'Algoritmo 2, serve per determinare il rango della matrice $H_{2N-L,L+1}$. Infatti, numericamente, calcoliamo i rapporti tra un valore singolare e il primo σ_1 (che è il massimo degli $\{\sigma_i\}_i$) e quando otteniamo un rapporto sotto il valore soglia allora abbiamo determinato il rango (che corrisponde all'indice del numeratore del primo rapporto sotto il valore soglia).

2.3 Metodo del Matrix Pencil

Il metodo del Matrix Pencil è un'ulteriore variante del metodo di Prony. La differenza sostanziale che sussite tra il metodo ESPRIT e il metodo del Matrix Pencil sta nella fattorizzazione della matrice di Hankel $H_{2N-L,L+1}$ costruita a partire dai dati campionati $f_0, \ldots, f_{2N-1}$, in quanto il metodo ESPRIT si basa sulla decomposizione SVD mentre il Matrix Pencil su quella di tipo QR.

In questo caso, assumiamo inizialmente noto il limite superiore della sparsità L, e consideriamo la matrice di Hankel $H_{2N-L,L+1}$. Riportiamo solo l'Algoritmo 3, dato che i passaggi formali ricalcano quelli della Sezione precedente (2.2). Per i dettagli rimandiamo a [5].

Algoritmo 3 Algoritmo Matrix Pencil.

Input: $f_0, \ldots, f_{2N-1}$, $L \geq n$, Δ, ϵ.

1. (a) Eseguire la decomposizione QR della matrice $H_{2N-L,L+1}$ con matrice di permutazione Π_{L+1} tale che

$$H_{2N-L,L+1}\Pi_{L+1} = Q_{2N-L}R_{2N-L,L+1},$$

dove gli elementi diagonali della matrice R sono ordinati in ordine non crescente.

   (b) Definire la matrice $D = \text{diag}(R_{2N-L,L+1})$ e determinare il rango n di $H_{2N-L,L+1}$.

   (c) Definire la sottomatrice $D_n = D(1:n, 1:n)$ e costruire le matrici

$$S_{n,L}^{(m)} = D_n^{-1} R_{2N-L,L+1} \Pi_{L+1}^T (1:n, 1+m:L+m), \quad m = 0, 1.$$

2. Calcolare gli n autovalori della matrice definita da

$$Z_n^{QR} = ({S_{n,L}^{(0)}}^T)^+ {S_{n,L}^{(1)}}^T$$

e ricavare le frequenze ϕ_j con la formula (1.10).

3. Calcolare le ampiezze α_j risolvendo il sistema (sovradeterminato) $\sum_{j=1}^{n} \alpha_j \lambda_j^k = f_k$, $k = 0, \ldots, 2N-1$, dove $\{\lambda_j\}_j$ sono gli autovalori calcolati al passo 2.

Output: $\{\alpha_j\}_j$, $\{\phi_j\}_j$ con $j = 1, \ldots, n$.

Osserviamo che, similmente alla Sezione (2.2), per determinare il rango della matrice $H_{2N-L,L+1}$ dobbiamo calcolare i rapporti tra i valori diagonali di $R_{2N-L,L+1}$ e il primo di essi e verificare quale sia, tra questi rapporti, il primo sotto il valore soglia ϵ.

Capitolo 3

Esempi numerici

In questo Capitolo, testiamo e discutiamo le implementazioni del metodo di Prony precedentemente introdotte. In particolare, intendiamo validare numericamente i metodi ESPRIT e Matrix Pencil descritti al Capitolo 2, mostrando che ricostruiscono *sufficientemente bene* una data funzione attraverso suoi campionamenti equispaziati.

3.1 Le funzioni MATLAB utilizzate nei test numerici

Proponiamo un'implementazione in ambiente MATLAB dei metodi ESPRIT e Matrix Pencil, partendo da quelle presenti in [6]. Tali procedure sono state personalizzate e integrate al fine di adattarle agli obiettivi della nostra analisi.

In particolare, sia in `esprit` sia in `matrix_pencil`, sono state modificate le righe di codice riguardanti la ricerca del rango della matrice di Hankel, in modo tale da forzarlo ad essere pari ad una certa sparsità fissata nel caso in cui questo dovesse risultare strettamente minore di quest'ultima.

Inoltre, nella funzione `esprit`, diversamente da quella di riferimento, per determinare correttamente le frequenze sono stati estratti i valori coniugati degli autovalori della matrice Z.

Riportiamo di seguito le *function* che implementano tali metodi.

```
%% METODO ESPRIT

function [lambda,alpha] = esprit(dati,L,sparsita,epsilon)

% Costruiamo la matrice di Hankel attraverso i dati
H = transpose(hankel(dati(1:L+1),dati(L+1:end)));
% Decomposizione a valori singolari
[~,S,W] = svd(H);
% Cerchiamo la sparsità attraverso il rango della matrice di Hankel
n1 = find(diag(S)/S(1,1) < epsilon);
if isempty(n1)
    n = sparsita;
else
```

```
    n = n1(1)-1;
end
n = max(n, sparsita);

W0 = W(1:L,1:n);
W1 = W(2:L+1,1:n);
Z = W0\W1;
% Determiniamo gli autovalori della matrice quadrata Z
lambda = conj(eig(Z));

% Determiniamo le ampiezze
V = construct_V(lambda,length(dati));
alpha = V\(dati.');

end
```

```
%% METODO MATRIX PENCIL

function [lambda,alpha] = matrix_pencil(dati,L,sparsita,epsilon)

% Costruiamo la matrice di Hankel attraverso i dati
H = transpose(hankel(dati(1:L+1),dati(L+1:end)));
% Decomposizione QR
[~,R,P] = qr(H);
D = diag(R);
% Cerchiamo la sparsità attraverso il rango della matrice di Hankel
n1 = find(D/D(1) < epsilon);
if isempty(n1)
    n = sparsita;
else
    n = n1(1)-1;
end
n = max(n,sparsita);
Dn = inv(diag(D(1:n)));

S = R*P';
S0 = Dn*S(1:n,1:L);
S1 = Dn*S(1:n,2:L+1);
Z = pinv(transpose(S0))*transpose(S1);
% Determiniamo gli autovalori della matrice quadrata Z
lambda = eig(Z);

% Determiniamo le ampiezze
V = construct_V(lambda,length(dati));
alpha = V\(dati.');

end
```

Nelle precedenti funzioni è stata utilizzata la subroutine `construct_V`, che, al fine di ottenere una corretta costruzione di una matrice di Vandermonde a partire da un vettore dato e dalla sparsità fissata, è stata riadattata mediante il seguente listato.

```
function V = construct_V(v,n) % Input: v vettore riga, n sparsità.

t = (0:n-1)';
V = (v(:)).'.^t;

end
```

Per concludere la presentazione dei codici, riportiamo anche la funzione `accoppia`, con cui riordiniamo le ampiezze e le frequenze ricavate dalle applicazioni dei metodi, in modo tale da poterle *accoppiare* e definire correttamente le componenti della funzione ricostruita.

```
function w_ord = accoppia(v,w) % Input: vettori riga o colonna

w_ord = zeros(size(v));
% Definiamo un vettore booleano per tracciare gli elementi già accoppiati
usati = false(size(w));

for i = 1:length(v)
    % Calcoliamo le differenze con gli elementi non ancora usati
    diff = abs(v(i) - w);
    % Impostiamo a Inf gli elementi già usati
    diff(usati) = Inf;
    % Cerchiamo la differenza minima e l'indice corrispondente
    [~,index_min] = min(diff);
    % Aggiorniamo il vettore ordinato
    w_ord(i) = w(index_min);
    % Aggiorniamo il vettore booleano
    usati(index_min) = true;
end

end
```

3.2 I test numerici

Terminata la trattazione preliminare delle funzioni, possiamo procedere con l'esecuzione dei test numerici.

3.2.1 Un primo test numerico

Per il primo test numerico, ci ispiriamo a quanto descritto in [6, pp. 41-42] per le scelte della funzione di partenza, del numero di dati da campionare, del limite superiore da considerare per applicare i metodi e, successivamente, anche per la tipologia di perturbazione da applicare alle osservazioni iniziali.

Consideriamo la funzione

$$f(t) = \exp(0.3it) - \exp(0.7it) + 2\cos(t) - 2\exp(2.3it) + 5\exp(2.9it). \tag{3.1}$$

Osserviamo che f è costituita da una somma di funzione esponenziali, infatti la componente $2\cos(t)$, per la Formula di Eulero $\exp(it) = \cos(t) + i\sin(t)$, può essere riscritta come la parte reale della funzione complessa $\exp(it)$. Dunque, poiché

$$\cos(t) = \frac{\exp(it) + \exp(-it)}{2},$$

allora la funzione (3.1) può essere riscritta come

$$f(t) = \exp(0.3it) - \exp(0.7it) + \exp(it) + \exp(-it) - 2\exp(2.3it) + 5\exp(2.9it).$$

Da tale scrittura deduciamo i valori delle ampiezze e delle frequenze che dobbiamo determinare attraverso il metodo di Prony. I valori delle ampiezze sono

$$1, \quad -1, \quad 1, \quad 1, \quad -2, \quad 5,$$

e i valori delle rispettive frequenze

$$0.3i, \quad -7i, \quad i, \quad -i, \quad 2.3i, \quad 2.9i.$$

Per estrarre i campioni, fissiamo $N = 40$, $\Delta = 1$ e valutiamo f nei punti $t = 0, 1, 2, \ldots, 79$, ottenendo così 80 osservazioni. Inoltre fissiamo $L = 20$ come limite superiore all'effettiva sparsità $n = 6$.

Come in [6], assumiamo di non essere a conoscenza dei dati esatti, ma di possedere una serie di osservazioni affette da errore, che possiamo esprimere nella forma

$$\tilde{f}_k = f_k + e_k, \quad k = 0, \ldots, 79,$$

dove e_k è un errore gaussiano di media 0 e varianza $\sigma^2 = 10^{-14}, 10^{-13}, \ldots, 10^{-2}$. Più precisamente, per ogni scelta di σ^2 perturbiamo il segnale f con 500 diverse istanze di rumore aggiuntivo, applichiamo i metodi e calcoliamo la media dei risultati. Studiamo poi, al variare di σ^2, la media degli errori relativi per ogni parametro determinato nelle 500 prove e la media degli errori assoluti massimi sui due vettori delle ampiezze e delle frequenze, le cui componenti corrispondono ai parametri determinati nelle 500 prove. Tali errori sono intesi nel senso delle seguenti definizioni.

Definizione 3.2.1. Sia $x \in \mathbb{C}$ un'approssimazione del valore esatto $x^* \in \mathbb{C}$. L'*errore relativo* e_r tra x e x^* è definito da

$$e_r = \frac{|x - x^*|}{|x^*|},$$

dove con $|\cdot|$ si intende il modulo di un numero complesso.

Definizione 3.2.2. Sia $x \in \mathbb{C}^n$ un'approssimazione del vettore esatto $x^* \in \mathbb{C}^n$. L'*errore assoluto massimo* e_a tra x e x^* è definito come il massimo degli errori assoluti tra le

componenti dei vettori, pertanto

$$e_a = \max_{i=1}^{n} |x_i - x^*| = \|x - x^*\|_\infty$$

Il codice sorgente di questo test è riportato nell'Appendice A, mentre di seguito descriviamo i risultati ottenuti sugli errori nelle seguenti Tabelle 3.1, 3.2, 3.3, 3.4, 3.5 e 3.6. Precisiamo che, ad ogni esecuzione del test, gli output sono generalmente diversi, ma ciò dipende esclusivamente dalla perturbazione casuale dei campionamenti con cui costruiamo la matrice di Hankel alla quale applicare i metodi. Tuttavia, ad ulteriore prova dell'affidabilità dei metodi ESPRIT e Matrix Pencil, osserviamo che la ripetizione sistematica della simulazione mostra che la differenza sui risultati è irrilevante.

Durante l'esecuzione del test emergono alcuni *warning* relativi alle ultime righe della funzione `esprit`. Questi, probabilmente, sono causati dal fatto che la matrice V potrebbe non avere rango massimo.

Nonostante ciò, possiamo dedurre che entrambi i metodi ricostruiscono il segnale con sufficiente precisione. Dai risultati contenuti nelle Tabelle 3.1, 3.2, 3.3, 3.4, 3.5 e 3.6, possiamo osservare che, in generale, più aumenta la varianza σ^2 più aumentano anche gli errori, ma, come si evince dai grafici nelle Figure 3.1, 3.2, 3.3 e 3.4, anche nei casi peggiori, in cui gli errori hanno lo stesso ordine di grandezza delle perturbazioni, la ricostruzione del segnale è comunque accurata.

σ^2	α_1	α_2	α_3	α_4	α_5	α_6
10^{-14}	1.20×10^{-8}	2.20×10^{-8}	2.04×10^{-8}	3.04×10^{-8}	1.07×10^{-8}	4.30×10^{-9}
10^{-13}	3.91×10^{-8}	6.77×10^{-8}	6.68×10^{-8}	9.90×10^{-8}	3.31×10^{-8}	1.32×10^{-8}
10^{-12}	1.20×10^{-7}	2.17×10^{-7}	2.01×10^{-7}	3.09×10^{-7}	1.01×10^{-7}	4.36×10^{-8}
10^{-11}	4.00×10^{-3}	4.00×10^{-3}	4.00×10^{-3}	4.00×10^{-3}	4.00×10^{-3}	4.00×10^{-3}
10^{-10}	1.28×10^{-6}	2.16×10^{-6}	2.11×10^{-6}	3.10×10^{-6}	1.05×10^{-6}	4.35×10^{-7}
10^{-9}	3.86×10^{-6}	6.84×10^{-6}	6.60×10^{-6}	9.78×10^{-6}	3.13×10^{-6}	1.33×10^{-6}
10^{-8}	1.23×10^{-5}	2.16×10^{-5}	2.01×10^{-5}	3.06×10^{-5}	1.03×10^{-5}	4.13×10^{-6}
10^{-7}	3.96×10^{-5}	7.06×10^{-5}	6.63×10^{-5}	9.89×10^{-5}	3.42×10^{-5}	1.39×10^{-5}
10^{-6}	1.19×10^{-4}	2.15×10^{-4}	2.03×10^{-4}	3.07×10^{-4}	1.06×10^{-4}	4.12×10^{-5}
10^{-5}	3.88×10^{-4}	6.67×10^{-4}	6.44×10^{-4}	9.75×10^{-4}	3.25×10^{-4}	1.41×10^{-4}
10^{-4}	1.16×10^{-3}	2.17×10^{-3}	2.02×10^{-3}	3.07×10^{-3}	1.02×10^{-3}	4.21×10^{-4}
10^{-3}	3.87×10^{-3}	6.85×10^{-3}	6.51×10^{-3}	9.82×10^{-3}	3.37×10^{-3}	1.35×10^{-3}
10^{-2}	1.20×10^{-2}	2.12×10^{-2}	2.08×10^{-2}	3.12×10^{-2}	1.08×10^{-2}	4.29×10^{-3}

Tabella 3.1: Errori relativi sulle ampiezze $\{\alpha_j\}$, $j = 1, \ldots, 6$ determinate tramite metodo ESPRIT.

σ^2	ϕ_1	ϕ_2	ϕ_3	ϕ_4	ϕ_5	ϕ_6
10^{-14}	1.60×10^{-9}	7.14×10^{-10}	4.84×10^{-10}	4.59×10^{-10}	1.01×10^{-10}	3.33×10^{-11}
10^{-13}	5.29×10^{-9}	2.10×10^{-9}	1.49×10^{-9}	1.47×10^{-9}	3.23×10^{-10}	1.01×10^{-10}
10^{-12}	1.57×10^{-8}	6.97×10^{-9}	4.70×10^{-9}	4.77×10^{-9}	9.80×10^{-10}	3.36×10^{-10}
10^{-11}	9.45×10^{-8}	3.83×10^{-8}	2.83×10^{-8}	2.32×10^{-8}	5.42×10^{-9}	1.62×10^{-9}
10^{-10}	1.59×10^{-7}	6.95×10^{-8}	4.82×10^{-8}	4.80×10^{-8}	1.03×10^{-8}	3.31×10^{-9}
10^{-9}	5.07×10^{-7}	2.16×10^{-7}	1.54×10^{-7}	1.52×10^{-7}	3.08×10^{-8}	1.02×10^{-8}
10^{-8}	1.54×10^{-6}	6.98×10^{-7}	4.66×10^{-7}	4.78×10^{-7}	1.00×10^{-7}	3.20×10^{-8}
10^{-7}	4.91×10^{-6}	2.22×10^{-6}	1.50×10^{-6}	1.53×10^{-6}	3.35×10^{-7}	1.08×10^{-7}
10^{-6}	1.58×10^{-5}	6.75×10^{-6}	4.82×10^{-6}	4.58×10^{-6}	1.05×10^{-6}	3.15×10^{-7}
10^{-5}	5.13×10^{-5}	2.18×10^{-5}	1.43×10^{-5}	1.55×10^{-5}	3.17×10^{-6}	1.08×10^{-6}
10^{-4}	1.59×10^{-4}	7.00×10^{-5}	4.63×10^{-5}	4.70×10^{-5}	9.99×10^{-6}	3.30×10^{-6}
10^{-3}	5.06×10^{-4}	2.17×10^{-4}	1.55×10^{-4}	1.52×10^{-4}	3.33×10^{-5}	1.02×10^{-5}
10^{-2}	1.63×10^{-3}	6.72×10^{-4}	4.62×10^{-4}	4.74×10^{-4}	1.07×10^{-4}	3.29×10^{-5}

Tabella 3.2: Errori relativi sulle frequenze $\{\phi_j\}$, $j = 1, \ldots, 6$, determinate tramite metodo ESPRIT.

σ^2	α_1	α_2	α_3	α_4	α_5	α_6
10^{-14}	1.26×10^{-8}	2.20×10^{-8}	2.05×10^{-8}	3.05×10^{-8}	1.05×10^{-8}	4.29×10^{-9}
10^{-13}	3.87×10^{-8}	6.74×10^{-8}	6.47×10^{-8}	9.88×10^{-8}	3.28×10^{-8}	1.35×10^{-8}
10^{-12}	1.25×10^{-7}	2.09×10^{-7}	2.10×10^{-7}	3.11×10^{-7}	1.09×10^{-7}	4.10×10^{-8}
10^{-11}	3.92×10^{-7}	6.92×10^{-7}	6.49×10^{-7}	9.75×10^{-7}	3.30×10^{-7}	1.32×10^{-7}
10^{-10}	1.25×10^{-6}	2.18×10^{-6}	2.06×10^{-6}	3.12×10^{-6}	1.09×10^{-6}	4.28×10^{-7}
10^{-9}	3.93×10^{-6}	6.97×10^{-6}	6.54×10^{-6}	9.80×10^{-6}	3.22×10^{-6}	1.32×10^{-6}
10^{-8}	1.25×10^{-5}	2.18×10^{-5}	2.06×10^{-5}	3.09×10^{-5}	1.00×10^{-5}	4.14×10^{-6}
10^{-7}	3.90×10^{-5}	6.99×10^{-5}	6.41×10^{-5}	9.56×10^{-5}	3.43×10^{-5}	1.29×10^{-5}
10^{-6}	1.24×10^{-4}	2.10×10^{-4}	2.09×10^{-4}	3.06×10^{-4}	1.05×10^{-4}	4.38×10^{-5}
10^{-5}	3.90×10^{-4}	6.80×10^{-4}	6.46×10^{-4}	9.62×10^{-4}	3.36×10^{-4}	1.35×10^{-4}
10^{-4}	1.25×10^{-3}	2.18×10^{-3}	2.01×10^{-3}	3.06×10^{-3}	1.06×10^{-3}	4.10×10^{-4}
10^{-3}	3.75×10^{-3}	6.54×10^{-3}	6.35×10^{-3}	9.73×10^{-3}	3.33×10^{-3}	1.33×10^{-3}
10^{-2}	1.21×10^{-2}	2.18×10^{-2}	2.11×10^{-2}	3.12×10^{-2}	1.09×10^{-2}	4.20×10^{-3}

Tabella 3.3: Errori relativi sulle ampiezze $\{\alpha_j\}$, $j = 1, \ldots, 6$, determinate tramite metodo Matrix Pencil.

σ^2	ϕ_1	ϕ_2	ϕ_3	ϕ_4	ϕ_5	ϕ_6
10^{-14}	1.59×10^{-9}	7.00×10^{-10}	5.00×10^{-10}	4.57×10^{-10}	1.02×10^{-10}	3.30×10^{-11}
10^{-13}	4.95×10^{-9}	2.10×10^{-9}	1.52×10^{-9}	1.55×10^{-9}	3.14×10^{-10}	1.07×10^{-10}
10^{-12}	1.63×10^{-8}	6.77×10^{-9}	4.92×10^{-9}	4.75×10^{-9}	1.04×10^{-9}	3.15×10^{-10}
10^{-11}	5.04×10^{-8}	2.21×10^{-8}	1.45×10^{-8}	1.51×10^{-8}	3.25×10^{-9}	1.03×10^{-9}
10^{-10}	1.57×10^{-7}	6.91×10^{-8}	4.63×10^{-8}	4.65×10^{-8}	1.05×10^{-8}	3.35×10^{-9}
10^{-9}	5.10×10^{-7}	2.20×10^{-7}	1.51×10^{-7}	1.53×10^{-7}	3.13×10^{-8}	1.03×10^{-8}
10^{-8}	1.60×10^{-6}	7.07×10^{-7}	4.56×10^{-7}	4.64×10^{-7}	9.99×10^{-8}	3.20×10^{-8}
10^{-7}	4.99×10^{-6}	2.21×10^{-6}	1.46×10^{-6}	1.48×10^{-6}	3.22×10^{-7}	9.89×10^{-8}
10^{-6}	1.61×10^{-5}	6.71×10^{-6}	4.86×10^{-6}	4.67×10^{-6}	1.03×10^{-6}	3.34×10^{-7}
10^{-5}	5.23×10^{-5}	2.21×10^{-5}	1.47×10^{-5}	1.44×10^{-5}	3.38×10^{-6}	1.05×10^{-6}
10^{-4}	1.62×10^{-4}	7.09×10^{-5}	4.59×10^{-5}	4.70×10^{-5}	1.03×10^{-5}	3.17×10^{-6}
10^{-3}	5.00×10^{-4}	2.12×10^{-4}	1.47×10^{-4}	1.47×10^{-4}	3.27×10^{-5}	1.04×10^{-5}
10^{-2}	1.62×10^{-3}	6.79×10^{-4}	4.82×10^{-4}	4.62×10^{-4}	1.06×10^{-4}	3.26×10^{-5}

Tabella 3.4: Errori relativi sulle frequenze $\{\phi_j\}$, $j = 1, \ldots, 6$, determinate tramite metodo Matrix Pencil.

σ^2	Ampiezze	Frequenze
10^{-14}	3.64×10^{-8}	7.45×10^{-10}
10^{-13}	1.16×10^{-7}	2.34×10^{-9}
10^{-12}	3.65×10^{-7}	7.46×10^{-9}
10^{-11}	2.00×10^{-2}	5.24×10^{-8}
10^{-10}	3.68×10^{-6}	7.51×10^{-8}
10^{-9}	1.15×10^{-5}	2.38×10^{-7}
10^{-8}	3.60×10^{-5}	7.42×10^{-7}
10^{-7}	1.18×10^{-4}	2.37×10^{-6}
10^{-6}	3.64×10^{-4}	7.44×10^{-6}
10^{-5}	1.16×10^{-3}	2.35×10^{-5}
10^{-4}	3.56×10^{-3}	7.40×10^{-5}
10^{-3}	1.16×10^{-2}	2.38×10^{-4}
10^{-2}	3.65×10^{-2}	7.54×10^{-4}

Tabella 3.5: Errori assoluti massimi sulle ampiezze e sulle frequenze, $\{\alpha_j\}$ e $\{\phi_j\}$, $j = 1, \ldots, 6$, determinate tramite metodo ESPRIT.

σ^2	Ampiezze	Frequenze
10^{-14}	3.64×10^{-8}	7.59×10^{-10}
10^{-13}	1.18×10^{-7}	2.37×10^{-9}
10^{-12}	3.64×10^{-7}	7.45×10^{-9}
10^{-11}	1.14×10^{-6}	2.35×10^{-8}
10^{-10}	3.70×10^{-6}	7.33×10^{-8}
10^{-9}	1.15×10^{-5}	2.39×10^{-7}
10^{-8}	3.62×10^{-5}	7.38×10^{-7}
10^{-7}	1.14×10^{-4}	2.34×10^{-6}
10^{-6}	3.63×10^{-4}	7.40×10^{-6}
10^{-5}	1.14×10^{-3}	2.36×10^{-5}
10^{-4}	3.59×10^{-3}	7.47×10^{-5}
10^{-3}	1.14×10^{-2}	2.34×10^{-4}
10^{-2}	3.64×10^{-2}	7.52×10^{-4}

Tabella 3.6: Errori assoluti massimi sulle ampiezze e sulle frequenze, $\{\alpha_j\}$ e $\{\phi_j\}$, $j = 1, \ldots, 6$, determinate tramite metodo Matrix Pencil.

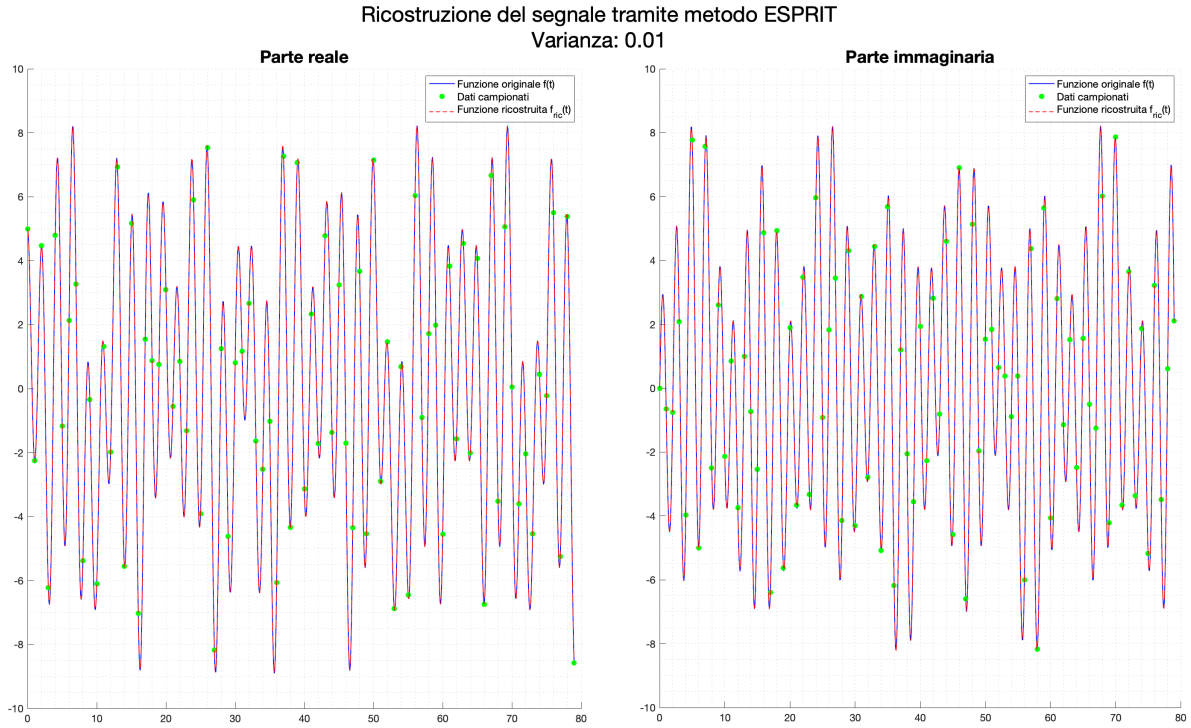


Figura 3.1: Grafico della funzione esatta $f(t)$, dei dati campionati e della funzione ricostruita $f_{ric}(t)$ tramite metodo ESPRIT applicato ai dati perturbati casualmente con funzione di errore gaussiana con media 0 e varianza $\sigma^2 = 10^{-2}$.

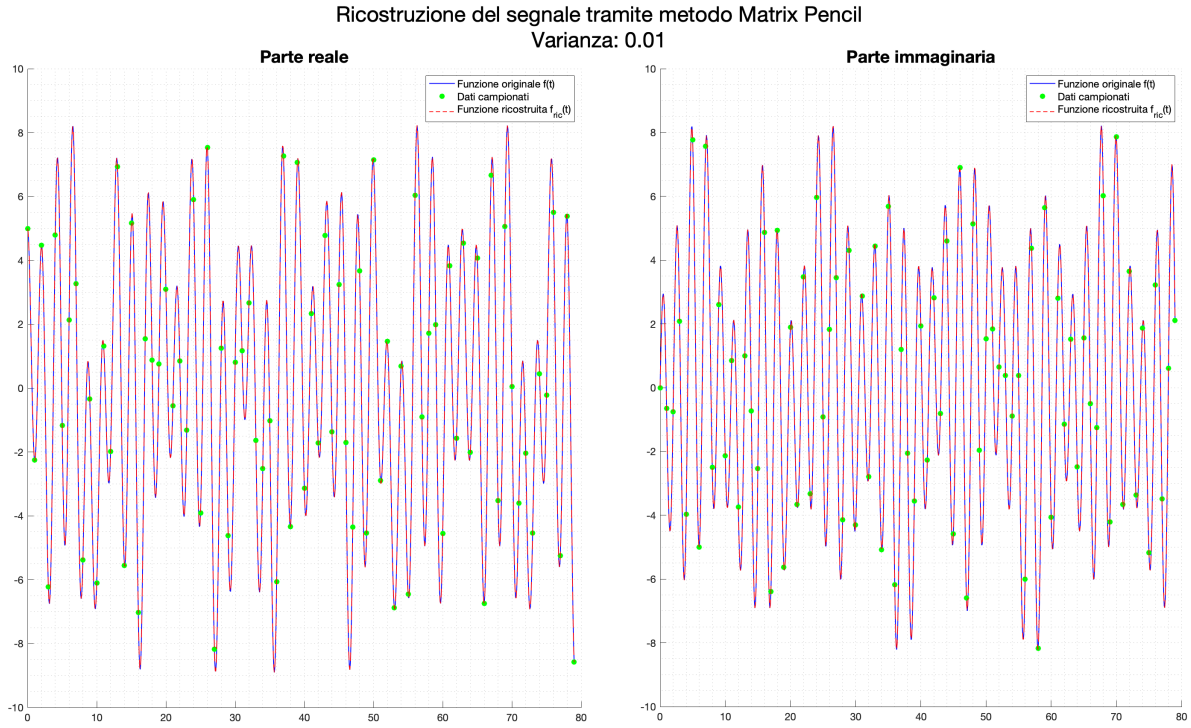


Figura 3.2: Grafico della funzione esatta $f(t)$, dei dati campionati e della funzione ricostruita $f_{ric}(t)$ tramite metodo Matrix Pencil applicato ai dati perturbati casualmente con funzione di errore gaussiana con media 0 e varianza $\sigma^2 = 10^{-2}$.

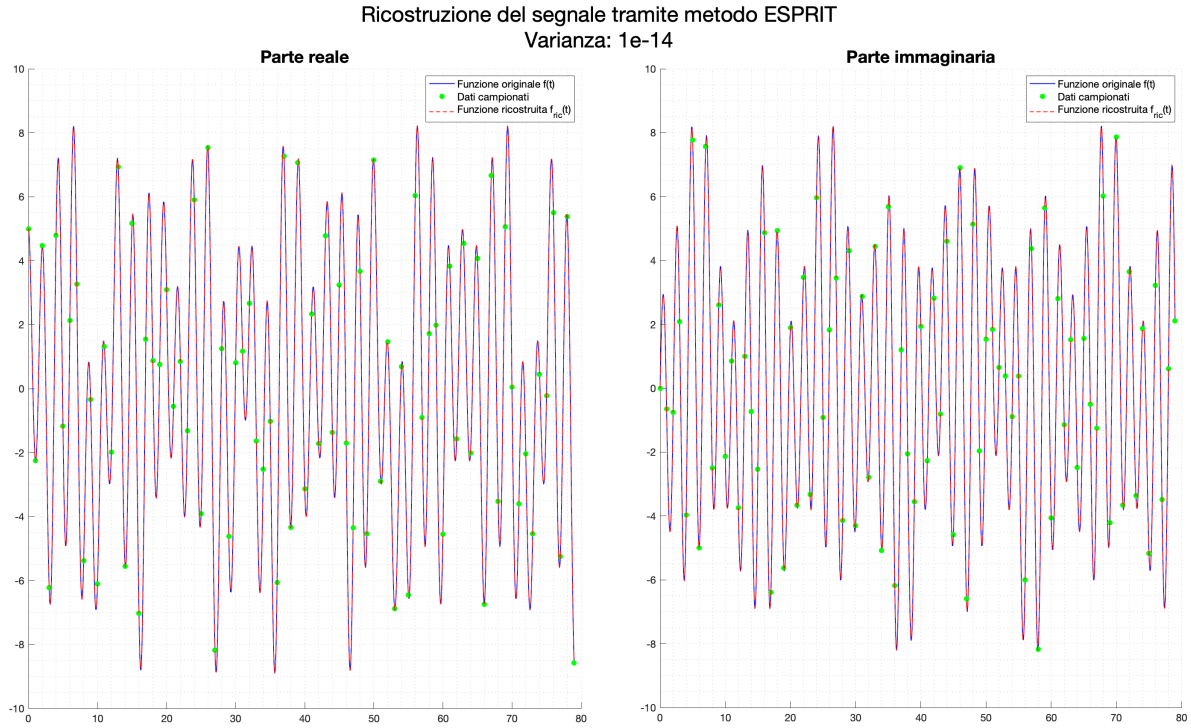


Figura 3.3: Grafico della funzione esatta $f(t)$, dei dati campionati e della funzione ricostruita $f_{ric}(t)$ tramite metodo ESPRIT applicato ai dati perturbati casualmente con funzione di errore gaussiana con media 0 e varianza $\sigma^2 = 10^{-14}$.

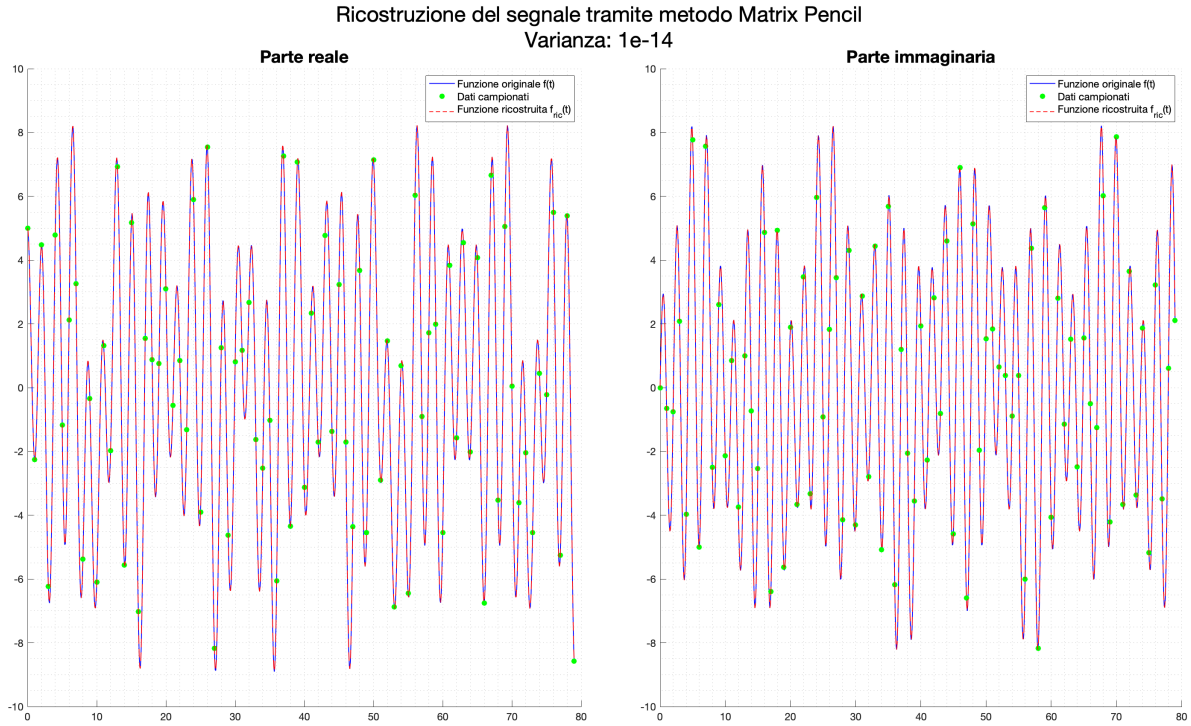


Figura 3.4: Grafico della funzione esatta $f(t)$, dei dati campionati e della funzione ricostruita $f_{ric}(t)$ tramite metodo Matrix Pencil applicato ai dati perturbati casualmente con funzione di errore gaussiana con media 0 e varianza $\sigma^2 = 10^{-14}$.

3.2.2 Un secondo test numerico

Ad ulteriore verifica del corretto funzionamento dei metodi ESPRIT e Matrix Pencil, presentiamo un secondo test numerico, basato sull'esperimento, condotto dal matematico Cornelius Lanczos, ampiamente descritto nell'Appendice B.

Consideriamo la funzione

$$f(t) = 0.0951e^{-t} + 0.8607e^{-3t} + 1.5576e^{-5t}, \quad t \geq 0 \tag{3.2}$$

e osserviamo che in (3.2) sia le ampiezze sia le frequenze sono puramente reali. Le ampiezze sono

$$0.0951, \quad 0.8607, \quad 1.5576, \tag{3.3}$$

e le rispettive frequenze sono

$$-1, \quad -3, \quad -5. \tag{3.4}$$

Come in [3, p. 276], per costruire l'insieme dei dati campionati, consideriamo un intervallo temporale di $\Delta = 0.05$ e valutiamo f nei punti $0, \Delta, 2 \cdot \Delta, \ldots, 23 \cdot \Delta$. Fissiamo la sparsità $n = 3$ e applichiamo i metodi descritti precedentemente ai dati campionati. Eseguiamo il codice tre volte: dapprima senza arrotondare i valori delle osservazioni iniziali, poi arrotondandoli a 12 e a 6 cifre decimali, creando così del rumore.

Il codice sorgente di questo test è riportato nell'Appendice A. Precisiamo che, modificando il valore di `digit_round` nello script, possiamo arrotondare i valori dei dati campionati a seconda del livello di precisione che vogliamo impostare. Nel caso in cui vogliamo prendere in considerazione l'intera precisione numerica dei dati, possiamo escludere le righe 18 e 19 del codice tramite un commento.

Dopo aver eseguito i test, possiamo dedurre, tramite le seguenti Tabelle 3.7, 3.8, 3.9, 3.10, 3.11 e 3.12 e le successive Figure 3.5, 3.6 e 3.7, che le funzioni f_{ric} costruite a partire dai parametri calcolati ricostruiscono *molto bene* i dati campionati inizialmente.

Dati esatti	ESPRIT	Matrix Pencil
2.51340000000000	2.51340000000000	2.51340000000000
2.04433337329109	2.04433337329109	2.04433337329109
1.66840443656437	1.66840443656437	1.66840443656437
1.36641802120829	1.36641802120829	1.36641802120829
1.12323248737248	1.12323248737248	1.12323248737249
0.926889718003714	0.926889718003714	0.926889718003715
0.767933856372759	0.767933856372759	0.767933856372759
0.638877552310634	0.638877552310634	0.638877552310634
0.533783531740170	0.533783531740169	0.533783531740168
0.447936361734709	0.447936361734709	0.447936361734708
0.377584788435010	0.377584788435009	0.377584788435008
0.319739319932635	0.319739319932634	0.319739319932633
0.272013077374646	0.272013077374646	0.272013077374645
0.232496552903175	0.232496552903175	0.232496552903174
0.199658954606508	0.199658954606507	0.199658954606507
0.172270412691387	0.172270412691387	0.172270412691386
0.149340566016837	0.149340566016837	0.149340566016837
0.130070020692174	0.130070020692174	0.130070020692174
0.113811932464400	0.113811932464400	0.113811932464401
0.100041558755877	0.100041558755878	0.100041558755878
0.0883320908454002	0.0883320908454009	0.0883320908454019
0.0783354401934973	0.0783354401934981	0.0783354401934994
0.0697669374344863	0.0697669374344872	0.0697669374344887
0.0623931253671945	0.0623931253671955	0.0623931253671972

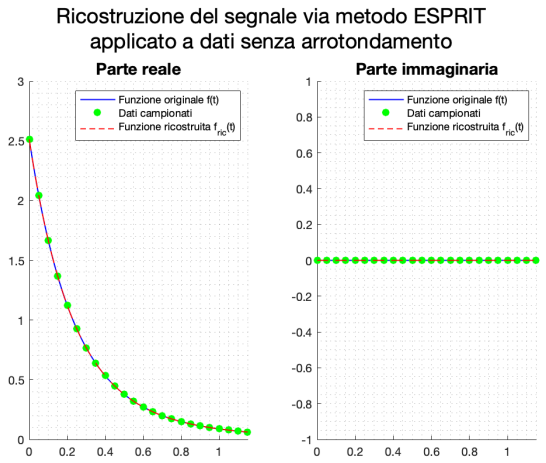
Tabella 3.7: Confronto tra dati esatti senza arrotondamento, dati ricostruiti tramite metodo ESPRIT e dati ricostruiti tramite metodo Matrix Pencil

Dati esatti	ESPRIT	Matrix Pencil
2.513400000000	2.51340000000007	2.51340000000006
2.044333373291	2.04433337329081	2.04433337329081
1.668404436564	1.66840443656404	1.66840443656404
1.366418021208	1.36641802120803	1.36641802120804
1.123232487372	1.12323248737236	1.12323248737236
0.926889718004	0.926889718003719	0.926889718003721
0.767933856373	0.767933856372872	0.767933856372872
0.638877552311	0.638877552310821	0.638877552310820
0.533783531740	0.533783531740397	0.533783531740394
0.447936361735	0.447936361734946	0.447936361734943
0.377584788435	0.377584788435231	0.377584788435227
0.319739319933	0.319739319932822	0.319739319932818
0.272013077375	0.272013077374786	0.272013077374782
0.232496552903	0.232496552903260	0.232496552903257
0.199658954607	0.199658954606535	0.199658954606532
0.172270412691	0.172270412691355	0.172270412691353
0.149340566017	0.149340566016749	0.149340566016748
0.130070020692	0.130070020692032	0.130070020692032
0.113811932464	0.113811932464210	0.113811932464212
0.100041558756	0.100041558755645	0.100041558755647
0.088332090845	0.0883320908451309	0.0883320908451343
0.078335440193	0.0783354401931962	0.0783354401932005
0.069766937434	0.0697669374341584	0.0697669374341636
0.062393125367	0.0623931253668446	0.0623931253668506

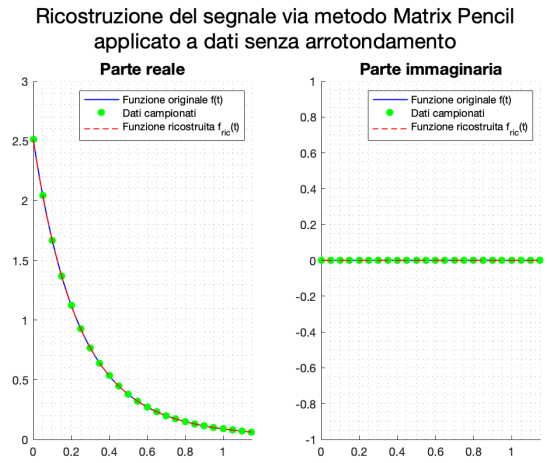
Tabella 3.8: Confronto tra dati esatti con arrotondamento a 12 difre decimali, dati ricostruiti tramite metodo ESPRIT e dati ricostruiti tramite metodo Matrix Pencil.

Dati esatti	ESPRIT	Matrix Pencil
2.513400	2.51340007626766	2.51340007679277
2.044333	2.04433299580079	2.04433299585574
1.668404	1.66840401358255	1.66840401317413
1.366418	1.36641772218877	1.36641772156103
1.123232	1.12323235381748	1.12323235323688
0.926890	0.926889730453762	0.926889730112970
0.767934	0.767933972921657	0.767933972911608
0.638878	0.638877729103779	0.638877729421766
0.533784	0.533783732457290	0.533783733031217
0.447936	0.447936560659006	0.447936561376070
0.377585	0.377584970064130	0.377584970797529
0.319739	0.319739476971413	0.319739477602040
0.272013	0.272013208205870	0.272013208637971
0.232497	0.232496659146829	0.232496659317803
0.199659	0.199659039068968	0.199659038954066
0.172270	0.172270477816614	0.172270477429543
0.149341	0.149340612841677	0.149340612232169
0.130070	0.130070048249603	0.130070047498903
0.113812	0.113811937552882	0.113811936767994
0.100042	0.100041535987058	0.100041535294409
0.088332	0.0883320328876019	0.0883320324266440
0.078335	0.0783353381393592	0.0783353380564775
0.069767	0.0697667812341209	0.0697667816771008
0.062393	0.0623929042813595	0.0623929053945642

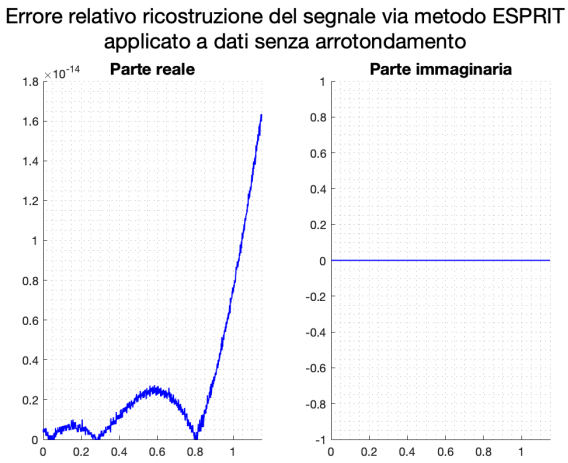
Tabella 3.9: Confronto tra dati esatti con arrotondamento a 6 difre decimali, dati ricostruiti tramite metodo ESPRIT e dati ricostruiti tramite metodo Matrix Pencil.



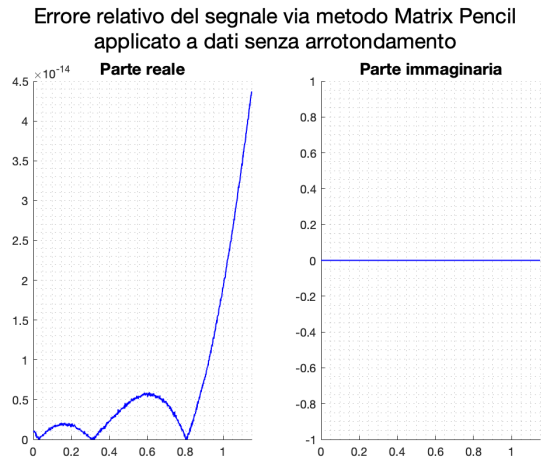
(a) Grafico della funzione esatta $f(t)$, dei dati esatti senza arrotondamento e della funzione ricostruita $f_{ric}(t)$ tramite metodo ESPRIT.



(b) Grafico della funzione esatta $f(t)$, dei dati esatti senza arrotondamento e della funzione ricostruita $f_{ric}(t)$ tramite metodo Matrix Pencil.

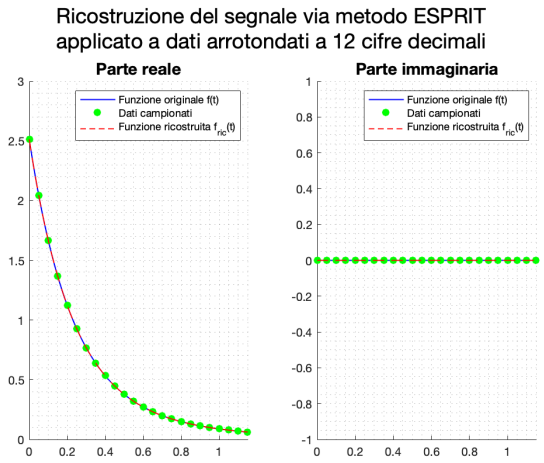


(c) Grafico dell'errore relativo di ricostruzione di $f(t)$ tramite metodo ESPRIT applicati ai dati senza arrotondamento.

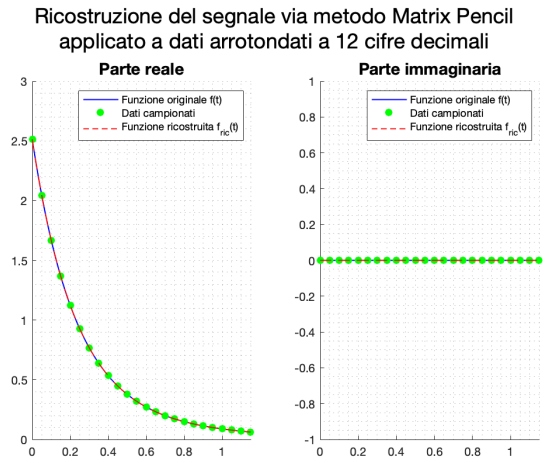


(d) Grafico dell'errore relativo di ricostruzione di $f(t)$ tramite metodo Matrix Pencil applicati ai dati senza arrotondamento.

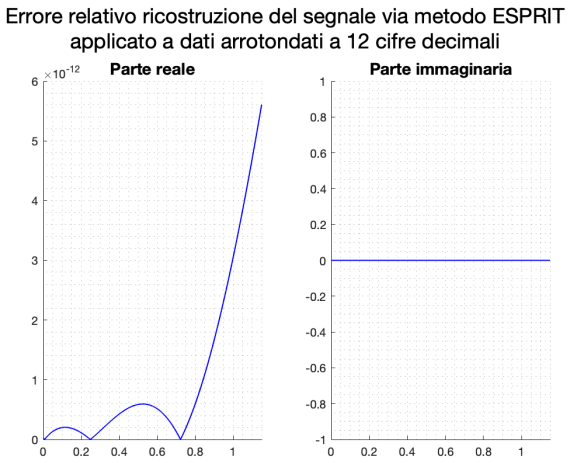
Figura 3.5: Output grafici del test eseguito a partire dai dati esatti senza arrotondamento.



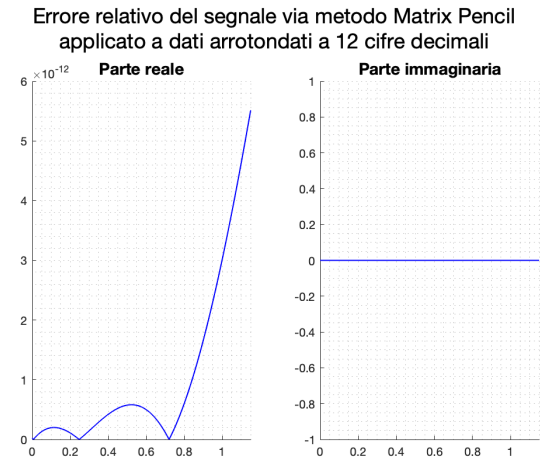
(a) Grafico della funzione esatta $f(t)$, dei dati esatti con arrotondamento a 12 cifre decimali e della funzione ricostruita $f_{ric}(t)$ tramite metodo ESPRIT.



(b) Grafico della funzione esatta $f(t)$, dei dati esatti con arrotondamento a 12 cifre decimali e della funzione ricostruita $f_{ric}(t)$ tramite metodo Matrix Pencil.

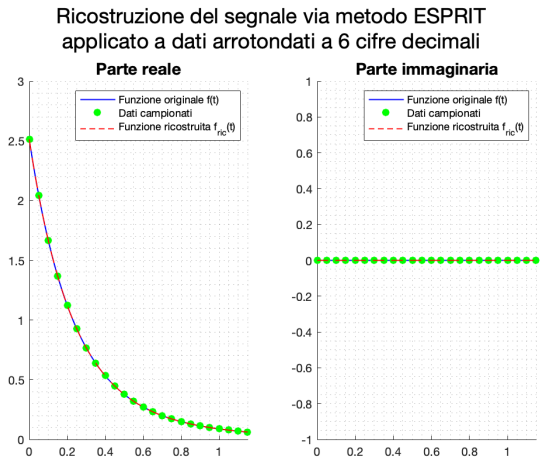


(c) Grafico dell'errore relativo di ricostruzione di $f(t)$ tramite metodo ESPRIT applicati ai dati esatti con arrotondamento a 12 cifre decimali.

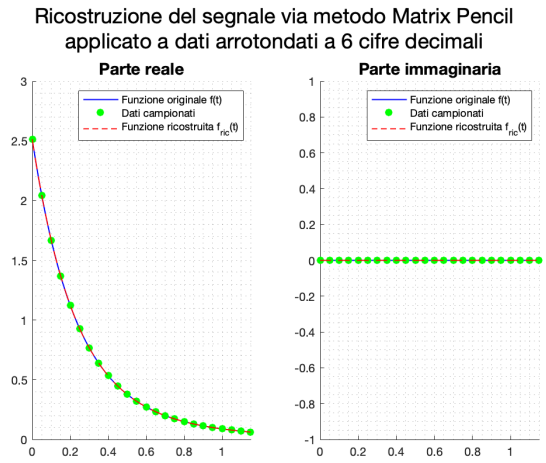


(d) Grafico dell'errore relativo di ricostruzione di $f(t)$ tramite metodo Matrix Pencil applicati ai dati esatti con arrotondamento a 12 cifre decimali.

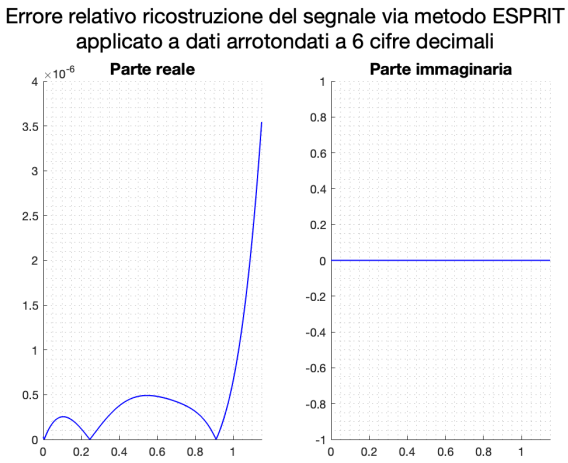
Figura 3.6: Output grafici del test eseguito a partire dai dati esatti arrotondati a 12 cifre decimali.



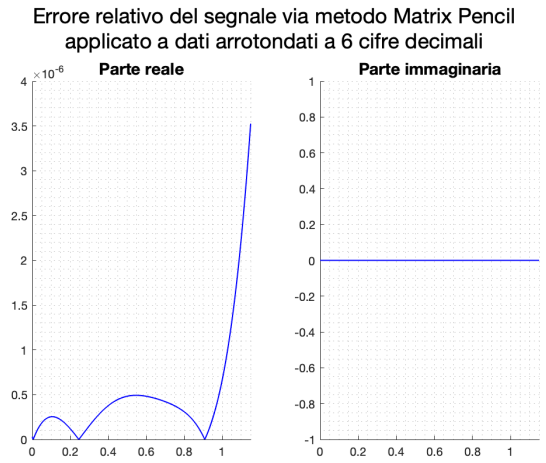
(a) Grafico della funzione esatta $f(t)$, dei dati esatti con arrotondamento a 6 cifre decimali e della funzione ricostruita $f_{ric}(t)$ tramite metodo ESPRIT.



(b) Grafico della funzione esatta $f(t)$, dei dati esatti con arrotondamento a 6 cifre decimali e della funzione ricostruita $f_{ric}(t)$ tramite metodo Matrix Pencil.



(c) Grafico dell'errore relativo di ricostruzione di $f(t)$ tramite metodo ESPRIT applicati ai dati esatti con arrotondamento a 6 cifre decimali.



(d) Grafico dell'errore relativo di ricostruzione di $f(t)$ tramite metodo Matrix Pencil applicati ai dati esatti con arrotondamento a 6 cifre decimali.

Figura 3.7: Output grafici del test eseguito a partire dai dati esatti arrotondati a 6 cifre decimali.

Tuttavia, in questo test, i metodi non riescono a riprodurre delle *buone* approssimazioni per le ampiezze (3.3) e per le frequenze (3.4) esatte. Possiamo dedurre ciò, osservando i valori degli errori relativi calcolati su ciascun parametro, organizzati nelle seguenti Tabelle 3.10, 3.11 e 3.12.

Ampiezza	ESPRIT	Matrix Pencil
0.0951	15.3785488959000	15.3785488959016
0.8607	1.20735302782520e-13	2.79135892330526e-13
1.5576	0.938944530046216	0.938944530046231

(a) Errori relativi sulle ampiezze approssimate tramite i metodi ESPRIT e Matrix Pencil a partire da valori non arrotondati dei campionamenti.

Frequenza	ESPRIT	Matrix Pencil
-1	3.99999999999987	3.99999999999968
-3	3.40468394218381e-15	3.71554638907886e-14
-5	0.799999999999996	0.800000000000042

(b) Errori relativi sulle frequenze approssimate tramite i metodi ESPRIT e Matrix Pencil a partire da valori non arrotondati dei campionamenti.

Tabella 3.10: Errori relativi sulle ampiezze e sulle frequenze con campionamenti non arrotondati.

Ampiezza	ESPRIT	Matrix Pencil
0.0951	15.3785488948435	15.3785488948357
0.8607	1.07355870575453e-10	1.07972704141592e-10
1.5576	0.938944530041052	0.938944530040924

(a) Errori relativi sulle ampiezze approssimate tramite i metodi ESPRIT e Matrix Pencil a partire da valori dei campionamenti arrotondati a 12 cifre decimali.

Frequenza	ESPRIT	Matrix Pencil
-1	4.00000000008425	4.00000000008437
-3	3.56165467489215e-11	3.61728424991270e-11
-5	0.799999999990313	0.799999999990205

(b) Errori relativi sulle frequenze approssimate tramite i metodi ESPRIT e Matrix Pencil a partire da valori dei campionamenti arrotondati a 12 cifre decimali.

Tabella 3.11: Errori relativi sulle ampiezze e sulle frequenze con campionamenti arrotondati a 12 cifre decimali.

Ampiezza	ESPRIT	Matrix Pencil
0.0951	15.3730115018579	15.3730456421905
0.8607	0.000463755412205140	0.000461253544301581
1.5576	0.938862655265989	0.938863356898517

(a) Errori relativi sulle ampiezze approssimate tramite i metodi ESPRIT e Matrix Pencil a partire da valori dei campionamenti arrotondati a 6 cifre decimali.

Frequenza	ESPRIT	Matrix Pencil
-1	4.00033259293782	4.00033081609112
-3	0.000287899899376942	0.000285825490664138
-5	0.799869812373742	0.799870962711592

(b) Errori relativi sulle frequenze approssimate tramite i metodi ESPRIT e Matrix Pencil a partire da valori dei campionamenti arrotondati a 6 cifre decimali.

Tabella 3.12: Errori relativi sulle ampiezze e sulle frequenze con campionamenti arrotondati a 6 cifre decimali.

Conclusioni

In questa tesi abbiamo approfondito il metodo di Prony per la ricostruzione di segnali tramite somme finite di esponenziali complessi.

Attraverso l'analisi teorica condotta inizialmente, abbiamo evidenziato le equivalenze tra l'approccio originale di Prony, il procedimento basato sugli approssimanti di Padé e la formulazione basata sulle matrici di Hankel.

Nonostante la solidità del modello teorico, il metodo di Prony originale risente dal punto di vista numerico di problemi di condizionamento legati al calcolo delle radici del polinomio di Prony. Per superare tale criticità numerica e trattare casi in cui la sparsità del segnale non è nota a priori, abbiamo considerato il metodo basato sui fasci di matrici. In particolare, abbiamo introdotto i metodi numerici ESPRIT e Matrix Pencil, basati rispettivamente sulle decomposizioni SVD e QR, attraverso i quali abbiamo costruito algoritmi alternativi all'originale.

Successivamente, mediante i test numerici implementati in ambiente MATLAB, abbiamo validato l'efficacia di tali varianti. Sia nel caso di segnali complessi affetti da errore gaussiano sia nell'esperimento di Lanczos con dati soggetti ad arrotondamento, i metodi ESPRIT e Matrix Pencil hanno garantito la ricostruzione del segnale e dei relativi parametri con errori ridotti.

Bibliografia

[1] A. Cuyt, W. Lee, Y. Hou, *Validated analysis of modulated signals: From de Prony to Padé and beyond*, Journal of Computational and Applied Mathematics 413 (2022), articolo 114346.

[2] G.H. Golub, C.F. Van Loan, Matrix Computations, 4th edition, John Hopkins University Press, Baltimore, 2013.

[3] C. Lanczos, Applied Analysis, Sir Isaac Pitman & Sons, 1957.

[4] R.J. Marks II, Advanced Topics in Shannon Sampling and Interpolation Theory, Springer-Verlag, 1993.

[5] D. Potts, M. Tasche, *Parameter estimation for nonincreasing exponential sums by Prony-like methods*, Linear Algebra and its Applications 439 (2012), pp.1024-1039.

[6] T.P. aus Rostock, Generalized Prony Method, Mathematisch-naturwissenschaftliche Fakultät, Georg-August-Universität Göttingen, 2013.

[7] L. Weiss, R.N. Mc Donough, *Prony's method, Z-transform, and Padé approximation*, SIAM Review 5(2) (1963), pp.145-149.

Appendice A

Codici MATLAB dei test numerici

Riportiamo gli script MATLAB completi che abbiamo utilizzato per l'esecuzione dei test numerici descritti nel Capitolo 3. Ricordiamo che le *function* utilizzate nel seguente codice sono reperibili nella Sezione 3.1.

A.1 Codice MATLAB del primo test numerico

Riportiamo il codice MATLAB relativo al test descritto nella Sezione 3.2.1.

```
%%  TEST METODO DI PRONY

clear all
close all
clc

% Definiamo la funzione da ricostruire
f = @(t) exp(0.3i*t) - exp(0.7i*t) + 2*cos(t) - 2*exp(2.3i*t) + ...
    5*exp(2.9i*t);
% Definiamo i campioni equispaziati con distanza pari a 1
N = 40;
t_campioni = 0:2*N-1;
dati = f(t_campioni);
% Definiamo l'upper bound e la sparsità
L = 20;
n = 6;
% Definiamo il vettore delle ampiezze effettive
ampiezze = [1,-1,1,1,-2,5];
% Definiamo il vettore delle frequenze effettive
frequenze = [0.3i,0.7i,1i,-1i,2.3i,2.9i];

% Tolleranza
epsilon = 1e-7;

% Definiamo le diverse varianze
esponenti = 14:-1:2;
num_steps = length(esponenti);
```

```
varianze = zeros(1,num_steps);
deviazioni = zeros(1,num_steps);
for i = 1:num_steps
    varianze(i) = 10^(-esponenti(i));
    deviazioni(i) = sqrt(varianze(i));
end

% Inizializziamo i vettori degli errori
err_rel_amp_es = zeros(num_steps,6);
err_rel_freq_es = zeros(num_steps,6);
err_inf_amp_es = zeros(num_steps,1);
err_inf_freq_es = zeros(num_steps,1);
err_rel_amp_mp = zeros(num_steps,6);
err_rel_freq_mp = zeros(num_steps,6);
err_inf_amp_mp = zeros(num_steps,1);
err_inf_freq_mp = zeros(num_steps,1);

%%
% ------------------------------------------------------------------------
%   ESPRIT
% ------------------------------------------------------------------------
for i = 1:num_steps
    dev_attuale = deviazioni(i);
    alpha_500 = zeros(500,6);
    phi_500 = zeros(500,6);
    err_rel_amp_500 = zeros(500,6);
    err_rel_freq_500 = zeros(500,6);
    err_inf_amp_500 = zeros(500,1);
    err_inf_freq_500 = zeros(500,1);

    % 500 prove
    for k = 1:500
        % Creiamo il vettore dei rumori
        e = (dev_attuale / sqrt(2)) * (randn(1, 2*N) + 1i * randn(1, 2*N));
        % Generiamo i dati perturbati
        dati_perturbati = dati + e;
        % Applichiamo il metodo ESPRIT
        [lambda, alpha] = esprit(dati_perturbati,L,n,epsilon);
        % Calcoliamo le frequenze e le riordiniamo
        phi = log(lambda);
        phi_ord = accoppia(frequenze,phi);
        % Riordiniamo le ampiezze ottenute tramite il metodo
        alpha_ord = accoppia(ampiezze,alpha);
        % Salviamo i valori delle ampiezze e delle frequenze ottenute
        alpha_500(k,:) = alpha_ord;
        phi_500(k,:) = phi_ord;
        % Calcoliamo gli errori relativi e in norma infinito sulle ampiezze
        % e sulle frequenze determinate alla k-esima prova
        err_rel_amp_500(k,:) = [abs((ampiezze-alpha_ord)./ampiezze)];
        err_rel_freq_500(k,:) = [abs((frequenze-phi_ord)./frequenze)];
```

```
        err_inf_amp_500(k,1) = max(abs(ampiezze - alpha_ord));
        err_inf_freq_500(k,1) = max(abs(frequenze - phi_ord));
    end

    % Calcoliamo i valori medi delle ampiezze e delle frequenze
    alpha_mean = mean(alpha_500,1);
    phi_mean = mean(phi_500,1);

    % Calcoliamo la media degli errori
    err_rel_amp_es(i,:) = mean(err_rel_amp_500, 1);
    err_rel_freq_es(i,:) = mean(err_rel_freq_500, 1);
    err_inf_amp_es(i,1) = mean(err_inf_amp_500);
    err_inf_freq_es(i,1) = mean(err_inf_freq_500);

    % Definiamo la funzione ricostruita
    f_ric = @(t) alpha_mean(1)*exp(phi_mean(1)*t) + ...
        alpha_mean(2)*exp(phi_mean(2)*t) + alpha_mean(3)*exp(phi_mean(3)*t) + ...
        alpha_mean(4)*exp(phi_mean(4)*t) + alpha_mean(5)*exp(phi_mean(5)*t) + ...
        alpha_mean(6)*exp(phi_mean(6)*t);
    % Grafico
    t_continuo = linspace(t_campioni(1),t_campioni(end), 1000);
    titolo = ['Ricostruzione del segnale tramite metodo ESPRIT'; ...
        "Varianza: " + varianze(i)];
    figure
    % Parte reale
    subplot(1,2,1)
    hold on
    grid minor
    plot(t_continuo,real(f(t_continuo)),"b-",LineWidth=1)
    plot(t_campioni,real(dati),"go",MarkerFaceColor="g")
    plot(t_continuo,real(f_ric(t_continuo)),"r--",LineWidth=1)
    title('Parte reale','Fontsize',14)
    legend('Funzione originale f(t)','Dati campionati', ...
        'Funzione ricostruita f_{ric}(t)')
    hold off
    % Parte immaginaria
    subplot(1,2,2)
    hold on
    grid minor
    plot(t_continuo,imag(f(t_continuo)),"b-",LineWidth=1)
    plot(t_campioni,imag(dati),"go",MarkerFaceColor="g")
    plot(t_continuo,imag(f_ric(t_continuo)),"r--",LineWidth=1)
    title('Parte immaginaria','Fontsize',14)
    legend('Funzione originale f(t)','Dati campionati', ...
        'Funzione ricostruita f_{ric}(t)')
    hold off
    sgtitle(titolo,'Fontsize',18)

end

```

```
% Inseriamo in due tabelle gli errori relativi sulle ampiezze e sulle
% frequenze che abbiamo determinato
TabErrRelAmpES = table(varianze(:),err_rel_amp_es, ...
    'VariableNames',{'Varianze','Err. relativo amp.'});
TabErrRelFreqES = table(varianze(:),err_rel_freq_es, ...
    'VariableNames',{'Varianze','Err. relativo freq.'});
openvar("TabErrRelAmpES")
openvar("TabErrRelFreqES")
% Inseriamo in un'unica tabella gli errori assoluti massimo sulle ampiezze
% e sulle frequenze che abbiamo determinato
TabErrInfES = table(varianze(:),err_inf_amp_es,err_inf_freq_es, ...
    'VariableNames',{'Varianze','Err. assoluto massimo amp.', ...
    'Err. assoluto massimo freq.'});
openvar("TabErrInfES")

%%
% ------------------------------------------------------------------------
%   MATRIX PENCIL
% ------------------------------------------------------------------------
for i = 1:num_steps
    dev_attuale = deviazioni(i);
    alpha_500 = zeros(500,6);
    phi_500 = zeros(500,6);
    err_rel_amp_500 = zeros(500,6);
    err_rel_freq_500 = zeros(500,6);
    err_inf_amp_500 = zeros(500,1);
    err_inf_freq_500 = zeros(500,1);

    % 500 prove
    for k = 1:500
        % Creiamo il vettore dei rumori
        e = (dev_attuale / sqrt(2)) * (randn(1, 2*N) + 1i * randn(1, 2*N));
        % Generiamo i dati perturbati
        dati_perturbati = dati + e;
        % Applichiamo il metodo Matrix Pencil
        [lambda, alpha] = matrix_pencil(dati_perturbati,L,n,epsilon);
        % Calcoliamo le frequenze e le riordiniamo
        phi = log(lambda);
        phi_ord = accoppia(frequenze,phi);
        % Riordiniamo le ampiezze ottenute tramite il metodo
        alpha_ord = accoppia(ampiezze,alpha);
        % Salviamo i valori delle ampiezze e delle frequenze ottenute
        alpha_500(k,:) = alpha_ord;
        phi_500(k,:) = phi_ord;
        % Calcoliamo gli errori relativi e in norma infinito sulle ampiezze
        % e sulle frequenze determinate alla k-esima prova
        err_rel_amp_500(k,:) = [abs((ampiezze-alpha_ord)./ampiezze)];
        err_rel_freq_500(k,:) = [abs((frequenze-phi_ord)./frequenze)];
        err_inf_amp_500(k,1) = max(abs(ampiezze - alpha_ord));
        err_inf_freq_500(k,1) = max(abs(frequenze - phi_ord));
```

```
    end

    % Calcoliamo i valori medi delle ampiezze e delle frequenze
    alpha_mean = mean(alpha_500,1);
    phi_mean = mean(phi_500,1);

    % Calcoliamo la media degli errori
    err_rel_amp_mp(i,:) = mean(err_rel_amp_500, 1);
    err_rel_freq_mp(i,:) = mean(err_rel_freq_500, 1);
    err_inf_amp_mp(i,1) = mean(err_inf_amp_500);
    err_inf_freq_mp(i,1) = mean(err_inf_freq_500);

    % Definiamo la funzione ricostruita
    f_ric = @(t) alpha_mean(1)*exp(phi_mean(1)*t) + ...
        alpha_mean(2)*exp(phi_mean(2)*t) + alpha_mean(3)*exp(phi_mean(3)*t) + ...
        alpha_mean(4)*exp(phi_mean(4)*t) + alpha_mean(5)*exp(phi_mean(5)*t) + ...
        alpha_mean(6)*exp(phi_mean(6)*t);
    % Grafico
    t_continuo = linspace(t_campioni(1),t_campioni(end), 1000);
    titolo = ['Ricostruzione del segnale tramite metodo Matrix Pencil'; ...
        "Varianza: " + varianze(i)];
    figure
    % Parte reale
    subplot(1,2,1)
    hold on
    grid minor
    plot(t_continuo,real(f(t_continuo)),"b-",LineWidth=1)
    plot(t_campioni,real(dati),"go",MarkerFaceColor="g")
    plot(t_continuo,real(f_ric(t_continuo)),"r--",LineWidth=1)
    title('Parte reale','Fontsize',14)
    legend('Funzione originale f(t)','Dati campionati', ...
        'Funzione ricostruita f_{ric}(t)')
    hold off
    % Parte immaginaria
    subplot(1,2,2)
    hold on
    grid minor
    plot(t_continuo,imag(f(t_continuo)),"b-",LineWidth=1)
    plot(t_campioni,imag(dati),"go",MarkerFaceColor="g")
    plot(t_continuo,imag(f_ric(t_continuo)),"r--",LineWidth=1)
    title('Parte immaginaria','Fontsize',14)
    legend('Funzione originale f(t)','Dati campionati', ...
        'Funzione ricostruita f_{ric}(t)')
    hold off
    sgtitle(titolo,'Fontsize',18)

end

% Inseriamo in due tabelle gli errori relativi sulle ampiezze e sulle
% frequenze che abbiamo determinato
```

```
TabErrRelAmpMP = table(varianze(:),err_rel_amp_mp, ...
    'VariableNames',{'Varianze','Err. relativo amp.'});
TabErrRelFreqMP = table(varianze(:),err_rel_freq_mp, ...
    'VariableNames',{'Varianze','Err. relativo freq.'});
openvar("TabErrRelAmpMP")
openvar("TabErrRelFreqMP")
% Inseriamo in un'unica tabella gli errori assoluti massimo sulle ampiezze
% e sulle frequenze che abbiamo determinato
TabErrInfMP = table(varianze(:),err_inf_amp_mp,err_inf_freq_mp, ...
    'VariableNames',{'Varianze','Err. assoluto massimo amp.', ...
    'Err. assoluto massimo freq.'});
openvar("TabErrInfMP")
```

A.2 Codice MATLAB del secondo test numerico

Riportiamo il codice MATLAB relativo al test descritto nella Sezione 3.2.2.

```
%% Esperimento Lanczos

clear all
close all
clc

% Definiamo la funzione esatta
f = @(t) 0.0951*exp(-t) + 0.8607*exp(-3*t) + 1.5576*exp(-5*t);

% Parametri di campionamento
t0 = 0;
d = 0.05;
N = 12;
t_campioni = t0 + d*(t0:2*N-1);
% Generazione dei dati
dati = f(t_campioni);
% Arrotondiamo alla "digit_round" cifra decimale
digit_round = 6;
dati = round(dati,digit_round);
% Definiamo l'upper bound e la sparsità
L = 7;
n = 3;

% Definiamo il vettore delle ampiezze effettive
ampiezze = [0.0951,0.8607,1.5576];
% Definiamo il vettore delle frequenze effettive
frequenze = [-1,-3,-5];

% Tolleranza
epsilon = 1e-7;

%%
% ------------------------------------------------------------------------
```

```
%   ESPRIT
% ------------------------------------------------------------------------
% Ricaviamo i parametri attraverso il metodo ESPRIT
[lambda,alpha] = esprit(dati,L,n,epsilon);
phi = log(lambda)./d;

% Definiamo la funzione ricostruita
t_continuo = linspace(t_campioni(1),t_campioni(end),1000);
f_ric = @(t) alpha(1)*exp(phi(1)*t) + alpha(2)*exp(phi(2)*t) + ...
    alpha(3)*exp(phi(3)*t);

% Grafico della ricostruzione del segnale
% titolo = ["Ricostruzione del segnale via metodo ESPRIT";
%    "applicato a dati senza arrotondamento"];
% titolo = ["Ricostruzione del segnale via metodo ESPRIT";
%    "applicato a dati arrotondati a 12 cifre decimali"];
titolo = ["Ricostruzione del segnale via metodo ESPRIT";
    "applicato a dati arrotondati a 6 cifre decimali"];
figure(1)
% Parte reale
subplot(1,2,1)
hold on
grid minor
plot(t_continuo,real(f(t_continuo)),"b-",LineWidth=1)
plot(t_campioni,real(dati),"go",MarkerFaceColor="g")
plot(t_continuo,real(f_ric(t_continuo)),"r--",LineWidth=0.9)
title('Parte reale','Fontsize',14)
legend('Funzione originale f(t)','Dati campionati', ...
    'Funzione ricostruita f_{ric}(t)')
hold off
% Parte immaginaria
subplot(1,2,2)
hold on
grid minor
plot(t_continuo,imag(f(t_continuo)),"b-",LineWidth=1)
plot(t_campioni,imag(dati),"go",MarkerFaceColor="g")
plot(t_continuo,imag(f_ric(t_continuo)),"r--",LineWidth=0.9)
title('Parte immaginaria','Fontsize',14)
legend('Funzione originale f(t)','Dati campionati', ...
    'Funzione ricostruita f_{ric}(t)')
hold off
sgtitle(titolo,'Fontsize',18)

% Grafico dell'errore relativo di ricostruzione in scala semilogaritmica
% titolo = ["Errore relativo ricostruzione del segnale via metodo ESPRIT";
%   "applicato a dati senza arrotondamento"];
% titolo = ["Errore relativo ricostruzione del segnale via metodo ESPRIT";
%    "applicato a dati arrotondati a 12 cifre decimali"];
titolo = ["Errore relativo ricostruzione del segnale via metodo ESPRIT";
    "applicato a dati arrotondati a 6 cifre decimali"];
```

```
figure(2)
subplot(1,2,1)
hold on
grid minor
den = abs(real(f(t_continuo)))+(real(f(t_continuo)) == 0) * eps;
semilogy(t_continuo,abs(real(f(t_continuo))-real(f_ric(t_continuo)))./den, ...
    "b-",LineWidth=1)
title('Parte reale','Fontsize',14)
subplot(1,2,2)
hold on
grid minor
den = abs(imag(f(t_continuo)))+(imag(f(t_continuo)) == 0) * eps;
semilogy(t_continuo,abs(imag(f(t_continuo))-imag(f_ric(t_continuo)))./den, ...
    "b-",LineWidth=1)
title('Parte immaginaria','Fontsize',14)
hold off
sgtitle(titolo,'Fontsize',18)

% Definiamo i vettori degli errori realitivi su ogni parametro
err_rel_amp_es = abs((ampiezze - alpha.')./ampiezze);
err_rel_freq_es = abs((frequenze - phi.')./frequenze);
% Definiamo gli errori massimi assoluti su ogni parametro
err_inf_amp_es = max(abs(ampiezze - alpha.'));
err_inf_freq_es = max(abs(frequenze - phi.'));
% Calcoliamo la ricostruzione dei dati campionati in partenza
dati_ric_es = f_ric(t_campioni);

%%
% ------------------------------------------------------------------------
%   MATRIX PENCIL
% ------------------------------------------------------------------------
% Ricaviamo i parametri attraverso il metodo Matrix Pencil
[lambda,alpha] = matrix_pencil(dati,L,n,epsilon);
phi = log(lambda)./d;

% Definiamo la funzione ricostruita
t_continuo = linspace(t_campioni(1),t_campioni(end),1000);
f_ric = @(t) alpha(1)*exp(phi(1)*t) + ...
        alpha(2)*exp(phi(2)*t) + alpha(3)*exp(phi(3)*t);

% Grafico della ricostruzione del segnale
% titolo = ["Ricostruzione del segnale via metodo Matrix Pencil";
%   "applicato a dati senza arrotondamento"];
% titolo = ["Ricostruzione del segnale via metodo Matrix Pencil";
%    "applicato a dati arrotondati a 12 cifre decimali"];
titolo = ["Ricostruzione del segnale via metodo Matrix Pencil";
    "applicato a dati arrotondati a 6 cifre decimali"];
figure(3)
% Parte reale
subplot(1,2,1)
```

```
hold on
grid minor
plot(t_continuo,real(f(t_continuo)),"b-",LineWidth=1)
plot(t_campioni,real(dati),"go",MarkerFaceColor="g")
plot(t_continuo,real(f_ric(t_continuo)),"r--",LineWidth=0.9)
title('Parte reale','Fontsize',14)
legend('Funzione originale f(t)','Dati campionati', ...
    'Funzione ricostruita f_{ric}(t)')
hold off
% Parte immaginaria
subplot(1,2,2)
hold on
grid minor
plot(t_continuo,imag(f(t_continuo)),"b-",LineWidth=1)
plot(t_campioni,imag(dati),"go",MarkerFaceColor="g")
plot(t_continuo,imag(f_ric(t_continuo)),"r--",LineWidth=0.9)
title('Parte immaginaria','Fontsize',14)
legend('Funzione originale f(t)','Dati campionati', ...
    'Funzione ricostruita f_{ric}(t)')
hold off
sgtitle(titolo,'Fontsize',18)

% Grafico dell'errore relativo di ricostruzione in scala semilogaritmica
% titolo = ["Errore relativo del segnale via metodo Matrix Pencil";
%   "applicato a dati senza arrotondamento"];
% titolo = ["Errore relativo del segnale via metodo Matrix Pencil";
%    "applicato a dati arrotondati a 12 cifre decimali"];
titolo = ["Errore relativo del segnale via metodo Matrix Pencil";
    "applicato a dati arrotondati a 6 cifre decimali"];
figure(4)
subplot(1,2,1)
hold on
grid minor
den = abs(real(f(t_continuo))) + (real(f(t_continuo)) == 0) * eps;
semilogy(t_continuo,abs(real(f(t_continuo)) - real(f_ric(t_continuo)))./den, ...
    "b-",LineWidth=1)
title('Parte reale','Fontsize',14)
subplot(1,2,2)
hold on
grid minor
den = abs(imag(f(t_continuo)))+(imag(f(t_continuo)) == 0) * eps;
semilogy(t_continuo,abs(imag(f(t_continuo))-imag(f_ric(t_continuo)))./den, ...
    "b-",LineWidth=1)
title('Parte immaginaria','Fontsize',14)
hold off
sgtitle(titolo,'Fontsize',18)

% Definiamo i vettori degli errori realitivi su ogni parametro
err_rel_amp_mp = abs((ampiezze - alpha.')./ampiezze);
err_rel_freq_mp = abs((frequenze - phi.')./frequenze);
```

```
% Definiamo gli errori massimi assoluti su ogni parametro
err_inf_amp_mp = max(abs(ampiezze - alpha.'));
err_inf_freq_mp = max(abs(frequenze - phi.'));
% Calcoliamo la ricostruzione dei dati campionati in partenza
dati_ric_mp = f_ric(t_campioni);

%%
% Inseriamo in due tabelle gli errori relativi sulle ampiezze e sulle
% frequenze che abbiamo determinato
TabErrRelAmp = table(ampiezze(:),err_rel_amp_es(:),err_rel_amp_mp(:), ...
    'VariableNames',{'Ampiezza','Err. rel. ESPRIT','Err. rel. Matrix Pencil'});
TabErrRelFreq = table(frequenze(:),err_rel_freq_es(:),err_rel_freq_mp(:), ...
    'VariableNames',{'Frequenza','Err. rel. ESPRIT','Err. rel. Matrix Pencil'});
% Inseriamo in due tabelle gli errori massimi assoluti sulle ampiezze e sulle
% frequenze che abbiamo determinato
TabErrInfAmp = table(err_inf_amp_es,err_inf_amp_mp,'VariableNames', ...
    {'ESPRIT','Matrix Pencil'});
TabErrInfFreq = table(err_inf_freq_es,err_inf_freq_mp,'VariableNames', ...
    {'ESPRIT','Matrix Pencil'});
openvar("TabErrRelAmp")
openvar("TabErrRelFreq")
openvar("TabErrInfAmp")
openvar("TabErrInfFreq")

%%
% Inseriamo in una tabella i valori delle osservazioni iniziali e le loro
% approssimazioni ottenute tramite i metodi applicati
TabDatiRic = table(dati(:),dati_ric_es(:),dati_ric_mp(:),'VariableNames', ...
    {'Dati','Dati ric. ESPRIT','Dati ric. Matrix Pencil'});
openvar("TabDatiRic")
```

Appendice B

L'esperimento di Lanczos

In [3, pp. 272-280] Cornelius Lanczos analizza il problema fisico delle misurazioni del decadimento radioattivo.

In questo contesto, consideriamo le ampiezze $\{\alpha_j\}_{j=1,\dots,n}$ e gli esponenti $\{\phi_j\}_{j=1,\dots,n}$ come numeri reali. In particolare, gli esponenti sono negativi poiché stiamo considerando il caso del decadimento radioattivo. Possiamo dunque rappresentare tale funzione f come

$$f(t) = \sum_{j=1}^{n} \alpha_j \exp(-\phi_j t), \quad \alpha_j \in \mathbb{R}, \phi_j \in \mathbb{R}_{\geq 0}, \tag{B.1}$$

dove abbiamo messo in evidenza il segno dell'esponente. Ripercorriamo il metodo originale descritto precedentemente, seppur modificando le espressioni descritte nella Sezione 1.1 affinché siano coerenti con le nuove variabili appena introdotte. Resta inteso che i passaggi logici del procedimento rimangono invariati. L'esperimento numerico di Lanczos segue i passaggi del metodo di Prony per determinare gli esponenti e conclude il calcolo delle ampiezze risolvendo il sistema (1.6), interpretandolo come un *problema dei momenti pesati*. Per una descrizione più estesa di questo problema si veda quanto descritto da Lanczos in [3, pp. 274-275].

Esaminiamo nel dettaglio l'esperimento numerico il cui codice MATLAB è riportato nell'appendice. Dalla legge del decadimento radioattivo considerato da Lanczos

$$f(t) = 0.0951e^{-t} + 0.8607e^{-3t} + 1.5576e^{-5t}, \quad t \geq 0, \tag{B.2}$$

estraiamo un insieme di 24 osservazioni di decadimento, a intervalli temporali di 3 minuti, ovvero $\Delta = 0.05$ ore a partire dall'istante 0, organizzate nella seguente Tabella (B.1).

k	f_k	k	f_k	k	f_k	k	f_k	k	f_k	k	f_k
1	2.51	5	1.12	9	0.53	13	0.27	17	0.15	21	0.09
2	2.04	6	0.93	10	0.45	14	0.23	18	0.13	22	0.08
3	1.67	7	0.77	11	0.38	15	0.20	19	0.11	23	0.07
4	1.37	8	0.64	12	0.32	16	0.17	20	0.10	24	0.06

Tabella B.1: Osservazioni di $f(t)$ a intervalli $\Delta = 0.05$ a partire da $t_0 = 0$.

Tali osservazioni sono da considerare accurate fino a mezza unità della seconda cifra decimale. A priori non sappiamo con quante componenti esponenziali ricostruire la funzione, perciò, conoscendo (B.2), proviamo con $n = 3$ componenti.

Osserviamo che, dal punto di vista teorico, la somma di funzioni esponenziali traslate nel tempo conserva la stessa forma esponenziale e, in particolare, gli stessi esponenti. Formalmente, se consideriamo una generica funzione composta di una sola componente esponenziale

$$f(t) := \alpha \exp(\phi t)$$

e sommiamo un gruppo di N campioni di f consecutivi distanziati di un intervallo fissato Δ, allora otteniamo la funzione $S(t)$ definita da

$$S(t) := f(t) + f(t + \Delta) + f(t + 2\Delta) + \ldots + f(t + (N - 1)\Delta)$$

che possiamo chiamare *somma traslata*. Sostituendo la definizione di $f(t)$ e raccogliendo a fattor comune, possiamo scrivere la funzione $S(t)$ come

$$S(t) = \alpha \exp(\phi t)(1 + \exp(\phi\Delta) + \exp(2\phi\Delta) + \ldots + \exp((N - 1)\phi\Delta)) = \text{cost} \cdot \alpha \exp(\phi t).$$

Poiché nella prima parte di questo esempio numerico l'obiettivo è determinare gli esponenti, il fatto che nella funzione somma traslata l'ampiezza sia scalata di un fattore costante è trascurabile. Possiamo ipotizzare che la scelta di Lanczos di considerare, per questo scopo, la funzione somma traslata invece che la media $S(t)/N$, che sarebbe matematicamente più intuitiva, sia dovuta ad una questione di *risparmio* di operazioni, nel senso che un'ulteriore divisione per N sarebbe comunque stata assorbita dalla costante ma avrebbe gravato sui tempi di calcolo dei calcolatori.

Proseguiamo dunque l'esperimento con questa strategia adottata da Lanczos, finalizzata ad aumentare il passo tra i nuovi dati campionati e, di conseguenza, a ridurre l'effetto dell'errore di misurazione.

Concretamente, raggruppiamo i dati in $2n = 6$ gruppi di 4 osservazioni consecutive ciascuno. Utilizziamo come nuovi dati le somme delle osservazioni di ogni gruppo e come nuovo intervallo di tempo $\tilde{\Delta} = 4\Delta = 0.2$.

Le nuove osservazioni sono

$$7.59, \quad 3.46, \quad 1.68, \quad 0.87, \quad 0.49, \quad 0.30.$$

Risolviamo il sistema, analogo al (1.15),

$$\begin{cases} 7.59b_3 + 3.46b_2 + 1.68b_1 + 0.87 = 0 \\ 3.46b_3 + 1.68b_2 + 0.87b_1 + 0.49 = 0 \\ 1.68b_3 + 0.87b_2 + 0.49b_1 + 0.30 = 0 \end{cases} . \tag{B.3}$$

Dividiamo ciascuna equazione per il coefficiente di b_3 e otteniamo il sistema

$$\begin{cases} b_3 + 0.4559b_2 + 0.2213b_1 + 0.1146 = 0 \\ b_3 + 0.4855b_2 + 0.2514b_1 + 0.1416 = 0 \\ b_3 + 0.5179b_2 + 0.2917b_1 + 0.1786 = 0 \end{cases} . \tag{B.4}$$

Per determinare b_1 e b_2, sottraiamo alla seconda equazione la prima e alla terza la seconda. Consideriamo allora

$$\begin{cases} 0.0296b_2 + 0.0301b_1 + 0.0270 = 0 \\ 0.0324b_2 + 0.0402b_1 + 0.0370 = 0 \end{cases} . \tag{B.5}$$

Osserviamo che, a causa delle regole di propagazione dell'imprecisione, l'accuratezza dei coefficienti del sistema (B.5) è di ± 0.01. Dunque possiamo vedere facilmente che (B.5) è composto da due equazioni ridondanti e che quindi non è possibile determinare in modo univoco b_1 e b_2. Questo mostra che le osservazioni della Tabella B.1 sono descrivibili tramite sole $n = 2$ componenti.

Quindi ricominciamo l'esperimento cercando di ricostruire la funzione di decadimento attraverso $n = 2$ componenti.

Suddividiamo le 24 osservazioni iniziali in $2n = 4$ gruppi ciascuno di 6 campionamenti di f. Sommando per ogni gruppo le osservazioni otteniamo 4 nuovi dati

$$9.64, \quad 3.09, \quad 1.15, \quad 0.51. \tag{B.6}$$

Il nuovo sistema diventa

$$\begin{cases} 9.64b_2 + 3.09b_1 + 1.15 = 0 \\ 3.09b_2 + 1.15b_1 + 0.51 = 0 \end{cases} \tag{B.7}$$

da cui ricaviamo le soluzioni

$$b_2 = 0.1640, \quad b_1 = -0.8839.$$

Ora, sostituiamo quanto appena trovato in (1.5) e otteniamo l'equazione di secondo grado

$$\begin{aligned} \sum_{i=0}^{n} b_{n-i}\Phi^i &= 0, \\ b_2 + b_1\Phi + b_0\Phi^2 &= 0, \\ 0.1640 - 0.8839\Phi + \Phi^2 &= 0, \end{aligned} \tag{B.8}$$

le cui radici sono

$$\Phi_1 = 0.619, \quad \Phi_2 = 0.265.$$

L'intervallo tra i nuovi dati (B.6) $\tilde{\Delta}$ risulta essere pari a $6 \cdot \Delta = 6 \cdot 0.05 = 0.3$, perciò gli esponenti, secondo la formula

$$\Phi_j := \exp(\phi_j \tilde{\Delta}), \quad j = 1, ..., n,$$

sono

$$\phi_1 = \frac{\log \Phi_1}{\tilde{\Delta}} = 4.45, \quad \phi_2 = \frac{\log \Phi_2}{\tilde{\Delta}} = 1.58. \tag{B.9}$$

Per determinare le ampiezze α_1 e α_2, teoricamente dovremmo risolvere il sistema (1.6), interpretandolo come problema dei momenti pesati. Dunque, in questo esempio, basterebbero solo 2 dati osservati per risolvere il sistema. Tuttavia, la scelta di soli due dati su 24 totali amplificherebbe l'errore sperimentale di misurazione e sarebbe da considerare uno *spreco* di informazioni. Infatti, Lanczos preferisce usare un metodo di *isolamento iterativo* per calcolare le ampiezze. In generale, questo prevede, inizialmente, di scegliere un combinazione lineare di n dati equidistanti, che isola il termine contenente una delle ampiezze α_j riducendo a zero i coefficienti dei termini contenenti le restanti ampiezze. Successivamente, si applica la stessa combinazione lineare ad altre n-uple di dati campionati; in questo modo si ottiene una successione di termini del tipo $\text{cost} \cdot \alpha_j \exp(-\phi_j t)$, da cui ricavare la singola ampiezza α_j. In questo modo possono essere ricavate tutte le ampiezze.

Nell'esempio in questione, calcoliamo le ampiezze con il metodo proposto la Lanczos, cominciando dal determinare α_1. Scegliamo una combinazione lineare dei dati f_1 e f_5 in modo tale che il termine contenente α_2 sia nullo. Scegliamo dunque i pesi w_1 e w_2 come segue

$$w_1 = -\exp(-\phi_2 \cdot 4 \cdot 0.05), \quad w_2 = 1 \tag{B.10}$$

e calcoliamo il valore della combinazione lineare di f_1 e f_5 con i pesi w_1 e w_2

$$\begin{aligned} V(1) &= w_1 f_1 + w_2 f_5 \\ &= -\exp(-\phi_2 \cdot 4 \cdot 0.05)[\alpha_1 \exp(-\phi_1 t_0) + \alpha_2 \exp(-\phi_2 t_0)] + \\ &\quad + [\alpha_1 \exp(-\phi_1 t_4) + \alpha_2 \exp(-\phi_2 t_4)]. \end{aligned} \tag{B.11}$$

Sfruttando il fatto che $t_4 = t_0 + 4 \cdot 0.05$, possiamo sviluppare l'espressione

$$\begin{aligned} V(1) &= w_1 f_1 + w_2 f_5 \\ &= -\exp(-\phi_2 \cdot 4 \cdot 0.05)[\alpha_1 \exp(-\phi_1 t_0) + \alpha_2 \exp(-\phi_2 t_0)] + \\ &\quad + [\alpha_1 \exp(-\phi_1 t_4) + \alpha_2 \exp(-\phi_2 t_4)] \\ &= -\exp(-\phi_2 \cdot 4 \cdot 0.05)[\alpha_1 \exp(-\phi_1 t_0) + \alpha_2 \exp(-\phi_2 t_0)] + \\ &\quad + [\alpha_1 \exp(-\phi_1 \cdot (t_0 + 4 \cdot 0.05) + \alpha_2 \exp(-\phi_2 \cdot (t_0 + 4 \cdot 0.05)] \\ &= \alpha_1[-\exp(-\phi_2 \cdot 4 \cdot 0.05) \exp(-\phi_1 t_0) + \exp(-\phi_1 \cdot (t_0 + 4 \cdot 0.05)] + \\ &\quad + \alpha_2[-\exp(-\phi_2 \cdot 4 \cdot 0.05) \exp(-\phi_2 t_0) + \exp(-\phi_2 \cdot (t_0 + 4 \cdot 0.05)] \\ &= \alpha_1[-\exp(-\phi_2 \cdot 4 \cdot 0.05) \exp(-\phi_1 t_0) + \exp(-\phi_1 \cdot (t_0 + 4 \cdot 0.05)] \\ &= \alpha_1 \exp(-\phi_1 t_0)[-\exp(-\phi_2 \cdot 4 \cdot 0.05) + \exp(-\phi_1 \cdot 4 \cdot 0.05)] \end{aligned} \tag{B.12}$$

da cui ricavare α_1, nel seguente modo

$$\alpha_1 = \frac{V(1) \exp(\phi_1 t_0)}{-\exp(-\phi_2 \cdot 4 \cdot 0.05) + \exp(-\phi_1 \cdot 4 \cdot 0.05)} = 2.245. \tag{B.13}$$

Iterando questo procedimento per le prime 12 coppie di dati equispaziati (fermandoci a questo punto poiché, come fa notare Lanczos, proseguire amplificherebbe la dispersione dei dati) e mantenendo gli stessi pesi (B.10) per le combinazioni lineari, otteniamo una successione di approssimazioni di α_1. Ricaviamo la media di questa successione e utilizziamo questa come prima ampiezza

$$\alpha_1 = 2.202.$$

In modo analogo determiniamo anche α_2, trovando il valore numerico

$$\alpha_2 = 0.305.$$

Dunque, l'espressione della funzione che abbiamo ricostruito è

$$f_{ric}(t) = 2.202e^{-4.45t} + 0.305e^{-1.58t}. \tag{B.14}$$

Osserviamo che questa funzione ricostruisce sufficientemente bene i punti. Infatti, valutando f_{ric} nei tempi $\{t_i\}_{i=0,\dots,23}$, in cui abbiamo campionato i dati originali, otteniamo quanto riassunto nella Tabella B.2.

k	$f_{ric,k}$	k	$f_{ric,k}$	k	$f_{ric,k}$	k	$f_{ric,k}$	k	$f_{ric,k}$	k	$f_{ric,k}$
1	2.507	5	1.126	9	0.533	13	0.270	17	0.148	21	0.088
2	2.044	6	0.929	10	0.447	14	0.230	18	0.130	22	0.079
3	1.672	7	0.769	11	0.376	15	0.197	19	0.114	23	0.070
4	1.370	8	0.639	12	0.318	16	0.173	20	0.100	24	0.063

Tabella B.2: Valutazione di $f_{ric}(t)$ nei tempi $\{t_i\}_{i=0,\dots 23}$.

Confrontando le Tabelle (B.2) con (B.1) possiamo osservare che la deviazione dei dati è sempre minore di 0.005 (tranne che per il quinto dato che raggiunge 0.006). Di conseguenza, osservando che la deviazione media 0.0026 è minore dell'incertezza sui dati iniziali, 0.005, possiamo concludere che la funzione ricostruita (B.14) è numericamente equivalente all'originale (B.2).

Nonostante ciò, la funzione (B.14) si presenta molto diversa dall'originale (B.2). Notiamo, infatti, che il numero di componenti esponenziali è stato ridotto da tre a due e che ampiezze ed esponenti sono stati notevolmente distorti. Lanczos giustifica questo *fallimento* spiegando che l'ostacolo non risiede nel metodo di valutazione bensì nella straordinaria sensibilità degli esponenti e delle ampiezze a variazioni molto piccole dei dati. Aggiunge, inoltre, che l'unico rimedio sarebbe un aumento dell'accuratezza.

Concludiamo illustrando nella Figura B.1 il grafico della funzione di decadimento $f(t)$, dei dati $\{f_k\}_{k=1,\dots,24}$ campionati a intervalli di $\Delta = 0.05$ e della funzione ricostruita mediante combinazione lineare di esponenziali $f_{ric}(t)$.

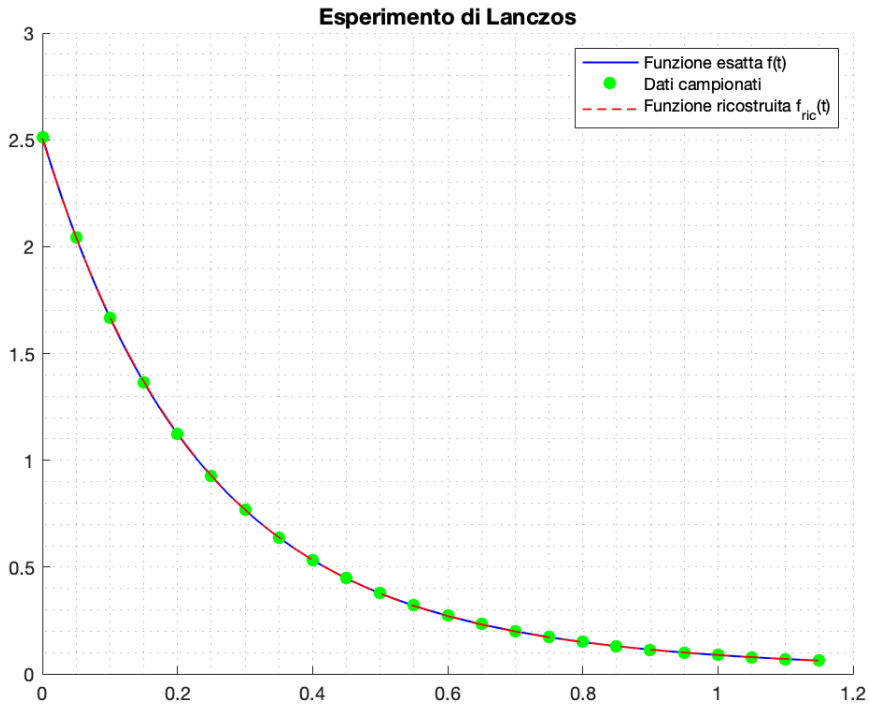


Figura B.1: Grafico della funzione di decadimento $f(t)$, dei dati f_k campionati e della funzione ricostruita f_{ric}.