

**UNIVERSITÀ DEGLI STUDI DI BERGAMO**

Dipartimento di Ingegneria Gestionale, dell'Informazione e della Produzione

Corso di Laurea Magistrale in Ingegneria Informatica

Classe N. LM32 - Lauree magistrali in Ingegneria informatica

**Diagnosi di guasti per cuscinetti a sfera  
con metodi di Compressed Sensing**

Relatore:

Chiar.mo Prof. Mirko Mazzoleni

Tesi di Laurea Magistrale

Nicola Belotti

Matricola n. 1055832

ANNO ACCADEMICO 2019/2020

# Contents

<b>1</b>	<b>Introduzione</b>	<b>5</b>
1.1	Motivazioni . . . . .	5
1.2	Compressed Sensing . . . . .	5
1.3	Schema dell'elaborato . . . . .	5
<b>2</b>	<b>Stato dell'Arte</b>	<b>8</b>
2.1	Il problema . . . . .	8
2.1.1	Health Monitoring . . . . .	8
2.1.2	I cuscinetti . . . . .	9
2.1.3	Tipi di guasto per cuscinetti a rotolamento . . . . .	9
2.2	Diagnosi classica per guasti sulla pista interna . . . . .	11
2.2.1	Calcolo frequenze di guasto . . . . .	11
2.2.2	Analisi vibrazioni . . . . .	13
2.2.3	Analisi Inviluppo . . . . .	14
2.2.4	L'algoritmo di Diagnosi per cuscinetti . . . . .	20
2.3	Il Compressed Sensing . . . . .	20
2.3.1	Introduzione . . . . .	20
2.3.2	Rappresentazioni diverse . . . . .	21
2.3.3	Sfruttare la sparsità . . . . .	22
2.4	Gli Algoritmi di ottimizzazione . . . . .	26
2.4.1	Ulteriori Formulazioni . . . . .	28
2.4.2	Lista di solver sperimentati . . . . .	30
<b>3</b>	<b>Setup Sperimentale</b>	<b>32</b>
3.1	Descrizione della macchina . . . . .	32
3.2	Descrizione dell'asse Y . . . . .	32
3.3	Sistemi di acquisizione dati . . . . .	34
3.3.1	cDAQ . . . . .	35
3.3.2	Encoder . . . . .	36
3.3.3	Termocoppia . . . . .	36
3.3.4	Accelerometro . . . . .	37
<b>4</b>	<b>Risultati</b>	<b>42</b>
4.1	Acquisizione dati . . . . .	42
4.2	Filtraggio e calcolo Inviluppo . . . . .	44
4.3	Applicazione dei metodi di Compressed Sensing . . . . .	45
4.4	Risolvere il sistema sottodeterminato . . . . .	47
4.5	Basis Pursuit Denoising . . . . .	50
4.6	LASSO . . . . .	51

4.7 Un'alternativa: il Downampling . . . . .	51
<b>5 Conclusioni</b>	<b>59</b>
5.1 Analisi risultati . . . . .	59
5.2 Confronto con l'algoritmo classico . . . . .	59
5.3 Ruolo Parametri . . . . .	59
5.4 Sviluppi futuri . . . . .	60
<b>A Codice</b>	<b>63</b>



# 1 Introduzione

L’industria 4.0 ha fatto sì che l’informatica funga da supporto ad ogni processo produttivo, diventando parte integrante di esso. Ogni macchinario ”dialoga” con gli altri, in quanto ognuno di essi è interconnesso tramite sensori. Ad esempio grazie alla manutenzione predittiva si può minimizzare lo stato di fermo dei componenti coinvolti, aumentando il rendimento di tutto il processo. L’obbiettivo di questo elaborato è offrire un altro approccio al problema dell’health monitoring dei cuscinetti di un macchinario: lo Spark X 2100 della Mandelli S.p.A. Su questo macchinario sono già stati fatti degli elaborati [1] dai colleghi del mio ateneo, quello che ci si riserva di fare è sviluppare e testare un approccio alternativo e valutarne efficacia, pregi e difetti confrontandosi con l’algoritmo classico di diagnostica per questo tipo di guasti che verrà introdotto nella prossima sezione.

## 1.1 Motivazioni

I cuscinetti sono un componente fondamentale nell’industria 4.0. Servono principalmente per ridurre l’attrito l’attrito tra due oggetti in movimento tra loro. Molto spesso succede che si possano verificare guasti a livello di questo componente e ve ne possono essere di diverso tipo. In questo elaborato si tratterà del guasto sulla pista più interna e ne verrà presentata un’alternativa all’algoritmo classico. I risultati saranno gli stessi e verranno confrontati con altre metriche in quanto come si vedrà è solo l’approccio al problema e conseguentemente la sua risoluzione a cambiare.

## 1.2 Compressed Sensing

Indica una famiglia di tecniche per trovare la soluzioni sparsa di un sistema di equazioni lineari sottodeterminato tra le infinite disponibili, a patto che questa sia sparsa. Da quando è stato scoperto ha trovato largo impiego in vari campi molto diversi tra loro. Si spazia dalla fotografia, alle risonanze magnetiche, alla tomografia. Il Compressed Sensing(CS) è in poche parole la risoluzione di sistemi lineare  $Ax = b$  sottodeterminati che ammettono soluzioni infinite, andando a cercare la soluzione più sparsa tra quelle possibili mediante opportuni algoritmi di ottimizzazione (si vedrà infatti nelle prossime sezioni che questi appartengono alla branca del *linear programming* benchè esistano anche altre risoluzioni). L’implementazione di questi metodi in questo elaborato viene completamente affidata al solver SPGL1 [2] di cui in bibliografia è presente il sito per scaricare il pacchetto per MATLAB. Online ve ne sono altri che ho avuto modo di sperimentare ma i risultati di questo elaborato in ogni caso sono stati tutti generati usando SPGL1.

## 1.3 Schema dell’elaborato

In questa sezione verrà presentato lo schema dell’elaborato, per venire incontro al lettore. Oltre a questa sezione introduttiva sono presenti altre tre sezioni:

- Stato dell'Arte: in questa sezione viene presentato l'algoritmo classico di diagnostica per i guasti ad un cuscinetto a rotolamento e viene introdotto formalmente il Compressed Sensing(formulazione del problema, ipotesi sottostanti e risoluzione del problema introdotto).
- Setup Sperimentale: descrizione del macchinario su cui viene eseguita la diagnostica del guasto, comprendendo anche i sensori disponibili.
- Risultati: Applicazione dei metodi di Compressed Sensing per la diagnostica del guasto e risultato finale.
- Conclusioni: conclusioni e ruolo dei parametri in gioco. Ulteriori sviluppi per il metodo proposto.
- A Codice: Appendice finale con note brevi sul codice MATLAB per eseguire il metodo proposto.



## 2 Stato dell'Arte

In questa sezione verrà presentato come in letteratura viene affrontato la formalizzazione di un problema di diagnosi per un guasto ad un cuscinetto, la sua risoluzione algoritmica, presentando alcuni metodi e il loro contesto di applicazione. Verrà poi introdotto il Compressed Sensing, la formalizzazione matematica, le ipotesi e le tecniche risolutive.

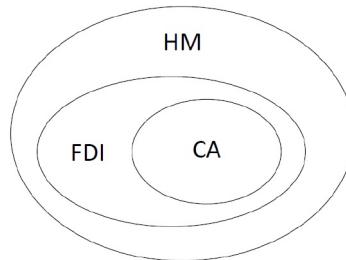
### 2.1 Il problema

#### 2.1.1 Health Monitoring

La definizione di guasto, detto anche avaria o difetto, viene definita in [3] ed è la seguente: una deviazione anormale dal valore ammissibile(o usuale) di almeno una variabile o di una proprietà caratteristica del sistema. Nel contesto industriale l'accezione di guasto può essere molteplice: si spazia dal difetto di progetto, di progettazione, al difetto dovuto a un errore nel processo produttivo o dovuto a una manutenzione assente o da un'usura di qualche suo componente durante la vita del macchinario (o relativamente anche di qualche suo componente). Il guasto può portare ad una *failure*, ovvero un fallimento nel funzionamento del sistema che può sfociare sia un'interruzione del processo produttivo che un'irregolarità non necessariamente debilitante di esso. Il monitoraggio per individuare i possibili difetti che un sistema può presentare prende il nome di *fault diagnosis* ed è un processo composto da tre differenti fasi:

- *Fault detection*: Determinazione di difetti nel sistema e quando si sono verificati.
- *Fault isolation*: Determinazione del tipo di fault e dove si è verificato all'interno di un sistema.
- *Fault identification*: Determinazione della natura di un guasto.

La supervisione dello stato di salute di sistema viene chiamata *Health Monitoring*(HM). Con le opportune tecniche è possibile monitorare i parametri significativi del sistema sottovisione e stabilirne la condizione. È possibile anche in certi contesti dare una valutazione dello stato di salute del sistema e questa è conosciuta come *Condition Assessment*. Schematicamente si può rappresentare:



Dove si vede come l'HM sia una disciplina di vario genere tra cui comprende la Fault Identification che al suo interno comprende le tecniche di Condition assessment.

Il processo di Health Monitoring può essere trattato in vario modo [4] a seconda dei contesti :

- *Model-Based Approach*: Questo tipo di approccio formula modelli matematici per la descrizione delle relazioni fra le variabili del sistema sotto controllo. Vengono formulati a partire da leggi fisiche oppure tramite metodi di indentificazione.
- *Signal-Based Approach*: Per questo tipo di tecniche è richiesta a priori la conoscenza del pattern del segnale misurabile. Gli eventuali guasti emergono nel segnale misurato, dal quale è possibile estrarre delle *features* che riguardino l'eventuale guasto.
- *Knowledge-Base Approach*: Molti contesti sono caratterizzati da un'ingente quantità di dati nei quali è possibile individuare pattern tipici di qualche tipo di guasto. Ciò è possibile grazie alle tecniche di intelligenza artificiale. Affidandosi ai soli dati, vengono anche detti *Data-Driven*

### 2.1.2 I cuscinetti

I cuscinetti a rotolamento sono componenti che permettono la rotazione reciproca di altri componenti, interponendosi tra loro. Limitano l'attrito trasformandolo da radente in volvente, riducendo l'energia dissipata(appunto nell'attrito). Gli elementi volventi sono geometricamente di vario tipo: i più comuni sono sfere, aghi, rulli (cilindrici o conici o a botte). La scelta di quale elemento utilizzare ovviamente dipende dall'applicazione, dalla tipologia del carico e dalla sua direzione. Ad esempio: per carichi assiali, i più adatti sono le sfere o i rulli conici, se invece i carichi sono radiali allora più adatti generalmente sono quelli con rulli cilindrici. Gli elementi che compongono i cuscinetti sono:

- Le superfici su cui rotolano i cuscinetti sono chiamate piste. Il carico posto sul cuscinetto è supportato da questa superficie di contatto. In genere l'anello interno poggia sull'asse o albero, mentre l'anello esterno poggia sull'alloggiamento.
- I corpi volventi (sfere o rulli). Le sfere hanno un contatto geometrico puntiforme sulle superfici delle piste dell'anello interno e di quello esterno, mentre per i rulli tale contatto è costituito da una linea. Teoricamente i cuscinetti volventi sono costruiti in modo tale da permettere un movimento di rivoluzione ai corpi volventi (rotazione simultanea attorno al proprio asse ed attorno all'asse del cuscinetto).
- La gabbia di trattenuta, separa i corpi volventi ad un intervallo regolare, li tiene in posizione tra la pista interna e quella esterna e ne permette la libera rotazione.

### 2.1.3 Tipi di guasto per cuscinetti a rotolamento

I guasti a cui può incorrere un cuscinetto sono di vario tipo: usura (dovuto a lubrificazione errata), fatica (carico troppo elevato), urti. Molto spesso le cause sono dovute alla fatica

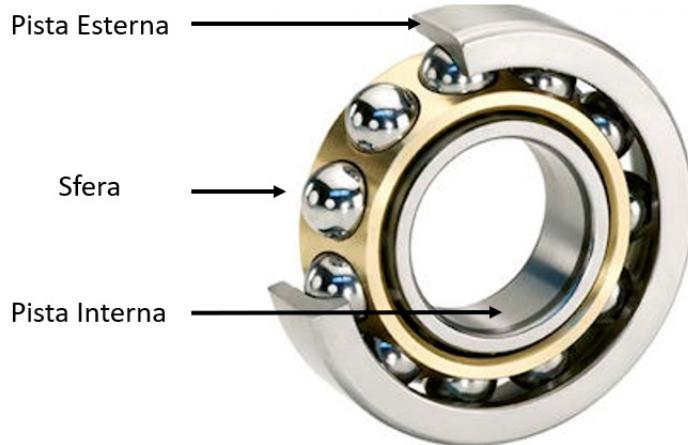


Figure 1: Sezione interna cuscinetto a sfere

o all'inadeguata lubrificazione. Nel primo caso, pur essendo sottoposto a carichi inferiori al carico di rottura, si danneggia, mentre nel secondo caso invece la mancata lubrificazione porta una sovratesteriorità locale che altera le proprietà del materiale e produce rotture, rendendo necessaria l'asportazione del materiale.

I difetti dei cuscinetti possono essere di vario tipo:

- Galling: è un tipo di usura che avviene quando due materiali sono compressi uno sull'altro e vi è frizione tra di essi.
- Spalling: oltre al rotolamento, le sfere o i rulli strisciano sulla superficie di contatto. Col passare del tempo si verifica il distacco di frammenti di metallo dalle piste e dai corpi volventi.
- Peeling: Formazione di irregolarità sul rivestimento applicato ad una superficie metallica.
- Pitting: Corrosione che porta buchi sulla superficie del metallo.
- Scoring: I detriti accumulati nel cuscinetto sotto condizioni improprie di lubrificazione o carichi eccessivi portano al guasto.
- Smearing: danno superficiale dovuto alla presenza di detriti di piccole dimensioni tra i componenti del cuscinetto a causa della rottura del film di lubrificante o a causa di slittamenti degli elementi.
- Fracture and cracks: frattura degli elementi causata ad esempio dal carico eccessivo o impulsivo, agente localmente sul componente considerato.
- Denting: Avviene quando i detriti, formati da piccole particelle metalliche, entrano nella zona di contatto tra l'elemento volvente e la pista.
- Fretting: Lo slittamento di due superfici porta all'usura. Il fretting si verifica sia sulle superfici di montaggio che sulle superfici di contatto tra le piste e gli elementi rotanti.

- Creep: creazione di un gioco dovuto allo slittamento delle superfici di montaggio.
- Seizure: Quando il cuscinetto si surriscalda velocemente durante la rotazione, il cuscinetto cambia colore. Dopo il surriscaldamento le piste, gli elementi rotanti e la gabbia lentamente iniziano a fondere e a deformarsi, accumulando sempre più danno.
- True brinelling: Avviene quando il carico sul cuscinetto è maggiore del limite elastico che caratterizza il materiale del cuscinetto.
- False brinelling: Sembra simile al true brinelling ma è dovuto a vibrazioni. Ad esempio, durante il trasporto, le vibrazioni possono far muovere gli elementi volventi che quindi lasciano dentellature sulle piste.

Con l'esecuzione di prove sperimentali si può studiare il comportamento di un cuscinetto danneggiato e il suo effetto sul sistema in cui è sottoposto. Per realizzare queste prove sperimentali è possibile operare in 2 modi: creare artificialmente il guasto(meccanicamente o chimicamente) oppure portare il cuscinetto a rompersi con l'applicazione di un carico eccessivo o di una rotazione a velocità elevata. In generale comunque si possono distinguere in localizzati, come può essere un taglio, una cricca, un'impronta o un'incisione oppure distribuiti, come un disallineamento, eccentricità degli anelli, o degli elementi volventi.

## 2.2 Diagnosi classica per guasti sulla pista interna

### 2.2.1 Calcolo frequenze di guasto

I difetti localizzati i livello dei cuscinetti possono essere di quattro tipi, ognuno dei quali genera una vibrazione ad una determinata frequenza caratteristica nello spettro. Questi difetti nello spettro sono indicatori del guasto e ne suggeriscono l'esistenza nel macchinario che si trova quindi in stato di avaria e che sarà necessaria una manutenzione. Il tipo di guasto che andremo a trattare in questa sezione e in tutto l'elaborato è il guasto sulla pista interna. L'impatto dell'elemento rotante con il difetto produce un urto, che eccita le frequenze di risonanza della struttura. I bursts generati dall'urto hanno un ampiezza che dipende dal carico agente sul cuscinetto ed è modulato dal passaggio del difetto nella zona carica del cuscinetto. Esistono formule teoriche per il calcolo delle frequenze, sia per i difetti sulla pista interna ma anche per gli altri tipi di guasti, tutte derivate dall'analisi della cinematica del cuscinetto e calcolate a partire dalla sua geometria e condizioni di lavoro. L'ipotesi sotto cui sono state dedotte è di rotolamento puro (considerando lo slittamento trascurabile).

Definendo:

- $w_i$ : velocità angolare anello interno (misurata in rad/s).
- $D$ : diametro anello esterno.
- $d$ : diametro anello interno.

- $\phi$ : angolo di contatto.
- $n$ : numero di elementi volventi.

Il diametro primitivo può essere espresso:

$$D_p = \frac{D_e + D_i}{2} \quad (1)$$

il diametro interno è:

$$D_i = D_p - d \cos \phi \quad (2)$$

mentre il diametro esterno è:

$$D_e = D_p + d \cos \phi \quad (3)$$

Utilizzando queste due equazioni è possibile calcolare le velocità periferiche delle piste come segue:

$$V_i = \omega_i \frac{D_i}{2} = \omega_i \frac{D_p - d \cos \phi}{2} \quad (4)$$

$$V_e = \omega_e \frac{D_e}{2} = \omega_e \frac{D_p + d \cos \phi}{2} \quad (5)$$

La velocità della gabbia viene calcolata come media delle velocità appena calcolate:

$$\begin{aligned} V_g &= \frac{V_i + V_e}{2} = \frac{2\omega_i \frac{D_p - d \cos \phi}{2} + 2\omega_e \frac{D_p + d \cos \phi}{2}}{2} \\ &= \frac{w_i(D_p - d \cos \phi) + w_e(D_p + d \cos \phi)}{2} \end{aligned} \quad (6)$$

Essa può essere scritta anche in funzione della frequenza  $f_g$  ottenendo:

$$V_g = \omega_g D_p = \pi f_g D_p \quad (7)$$

le frequenze angolari  $w_i$  e  $w_e$  in Hz sono:

$$w_i = 2\pi f_i \quad (8)$$

$$w_e = 2\pi f_e \quad (9)$$

La frequenza di rotazione degli elementi volventi rispetto alla pista interna, che chiamiamo  $f_{ri}$ , è quindi:

$$f_{ri} = f_g - f_i \quad (10)$$

che utilizzando 7 diventa:

$$f_{ri} = \frac{f_e + f_i}{2} \left( 1 + \frac{d}{D_p} \cos \phi \right) \quad (11)$$

Considerando che gli elementi rotanti siano  $n$  e che l'anello esterno è fermo ( $w_e = 0$ )

$$BFPI = \frac{n f_r}{2} \left( 1 - \frac{d}{D} \cos \phi \right)$$

Molto spesso i segnali derivati da macchine con componenti in rotazione si rivelano essere segnali ciclostazionari, ovvero segnali che variano nel tempo ciclicamente. Vengono usati per descrivere processi generati da fenomeni periodici i cui parametri però possono cambiare nel tempo. Si possono definire diversi ordini di ciclostazionarietà. Quella del 1° ordine ad esempio è quella che

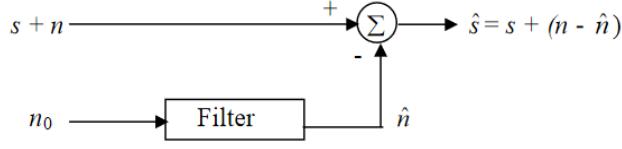
esibisce un segnale sinusoidale caratterizzato da un rumore di fondo stazionario. Diversamente, quella del 2° ordine è quella di un rumore stazionario modulato in ampiezza da una sinusoide. Benchè le condizioni di lavoro siano costanti (si pensi a velocità di rotazione, temperatura, momenti) avvengono una successione di fenomeni che rilasciano l'energia ritmicamente, che generano segnali transitori carichi di informazione. Questo treno di forze impulsive si presenta sottoforma di un treno di picchi seguiti da oscillazioni alle frequenze della struttura e che subiscono uno smorzamento molto rapido. I calcoli per la BFPI precedentemente calcolate sotto ipotesi di slittamento nullo, ma questo non è completamente trascurabile e si rivela come un differenza dell'ordine del 1-2% dalla frequenza teorica del guasto calcolata. La vibrazione di un elemento rotante si osserverà in bassa frequenza con un'armonica fondamentale e alcune prime armoniche (che dipendono dal macchinario, potrebbero essere la frequenza di rotazione dell'albero o la frequenza di ingranamento, ecc...). Per quanto riguarda le vibrazioni da danno sul cuscinetto si distribuiscono su tutto lo spettro e a seconda della grandezza della cricca sulla pista interna, la modulazione in ampiezza sarà maggiore generando uno spettro molto frastagliata (chiamato "a pettine").

### 2.2.2 Analisi vibrazioni

Purtroppo il guasto non è riconoscibile direttamente nel segnale grezzo, è necessario prima fare delle operazioni di pre-processing prima di diagnosticarlo correttamente. Una delle cause che rendono necessarie delle operazioni preliminari sui dati sono ad esempio le componenti in frequenza dovute agli ingranaggi che contaminano il segnale e mascherano il guasto. Di estrema importanza è quindi tagliare questa parte, o limitarla per consentire il riconoscimento del difetto. Vari modi sono possibili per eseguire il prefiltraggio, dipendentemente dal sistema, dai cuscinetti e ad esempio in [5] vengono citati:

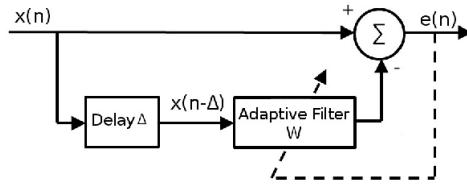
- Predizione lineare: ottiene un modello lineare per il segnale, che permetterà di rimuovere questa componente. Un esempio è l'utilizzo di un modello AR (auto regressivo) da togliere alla serie di dati nel tempo.
- Analisi degli ordini: Utilizzata quando rumore o vibrazioni nel tempo sono soggette ad un cambiamento. Il concetto di ordine viene utilizzato a frequenze rapportare rispetto alla frequenza di rotazione di riferimento. Le macchine rotanti possono incorrere in risonanze dannose che complicano LA diagnostica di guasti. Ulteriori dettagli possono essere trovati in bibliografia. [6, 7]
- Cancellazione del rumore adattiva(ANC) [8, 9]: un segnale  $s$  detto primario viene separato da un rumore  $n$  tramite l'utilizzo di un altro rumore "ausiliario"  $n_0$  correlato col primo, mediante l'utilizzo di un filtro adattativo.

Figure 2: Schema a blocchi ANC



- Cancellazione del rumore autoadattativa (SANC) [10], esegue un denoising utilizzando una versione ritardata del segnale.

Figure 3: Schema a blocchi SANC



- Separazione discreta/casuale (DRS) [11]: il SANC si adatta per piccole variazioni di velocità ma quando questa aumenta la convergenza è lunga per filtri di ordine alto. Il DRS sopprime a questo problema qualora la velocità non sia di valore variabile (in tal caso sarebbe da utilizzare un'analisi degli ordini come passo di preprocessing.)
- Media Sincrona Temporale(TSA)[12]: Separazione delle parti deterministiche del segnale dal rumore senza intaccare il segnale residuo. Metodo particolarmente lungo e che richiede parecchie operazioni per ogni armonica che compone il segnale.

L'algoritmo classico di diagnostica per un cuscinetto prevede degli stadi di preprocessing dei dati grezzi prima di poter stabilire il difetto di un macchinario a livello dei cuscinetti come quelli appena elencati. È anche possibile utilizzare questa fase di preprocessing per migliorare i risultati che si ottengono. Ad esempio il *Minimum Entropy deconvolution*(MED) può essere molto efficace nella deconvoluzione delle eccitazioni impulsive da una miscela di segnali di risposta, quando vi sono sfaldature e crepe negli ingranaggi. L'idea base del MED è trovare un filtro inverso che contrasta l'effetto del percorso di trasmissione, supponendo che l'eccitazione originale fosse impulsiva con alta curtosi.

### 2.2.3 Analisi Inviluppo

L'analisi dell'inviluppo è una tecnica ricorrente per la diagnostica dei guasti nei cuscinetti che sopprime all'impossibilità di vedere i picchi nelle frequenze di guasto direttamente nello spettro. Tramite questa tecnica [13, 5] è possibile infatti estrarre le componenti periodiche impulsive tipiche del guasto. È infatti tramite possibile estrarre la modulante a partire dal segnale modulato. Il contesto in cui viene applicato è il seguente:

- Segnale modulante:  $x_m(t) = A \cos w_m t$ , che è il segnale di interesse.

- Portante del segnale:  $x_p(t) = B \cos w_p t$ , che modulerà in ampiezza  $x_m$
- Segnale modulato:  $y(t) = C(t) \cos wt$ , che è il risultato della modulazione.

La modulazione in ampiezza si ottiene imponendo che  $C$  vari secondo la legge:

$$C(t) = A + k \cdot x_m(t) \quad (12)$$

$C(t)$  è infatti il valore funzione del tempo che modula in ampiezza il segnale,  $A$  l'ampiezza della portante in assenza di modulazione,  $k$  la costante di proporzionalità. Sostituendo  $x_m$  si ottiene:

$$C_m(t) = A + k \cdot \cos w_p t \quad (13)$$

È ora possibile riscrivere  $y(t)$  come segue:

$$y(t) = [A + k \cdot B \cos(w_p t)] \cos(w_p t) \quad (14)$$

Introduciamo  $m = k \frac{B}{A}$ , che rappresenta la massima variazione di ampiezza del segnale  $y(t)$  rispetto al valore  $A$  in assenza di modulazione, che chiameremo indice di modulazione. È quindi possibile scrivere:

$$y(t) = A[1 + m \cos(w_p t)] \cos(w_p t) \quad (15)$$

Utilizzando la formula di Werner:  $\cos(\alpha) \cdot \cos(\beta) = [\cos(\alpha - \beta) + \cos(\alpha + \beta)]$  l'equazione diventa:

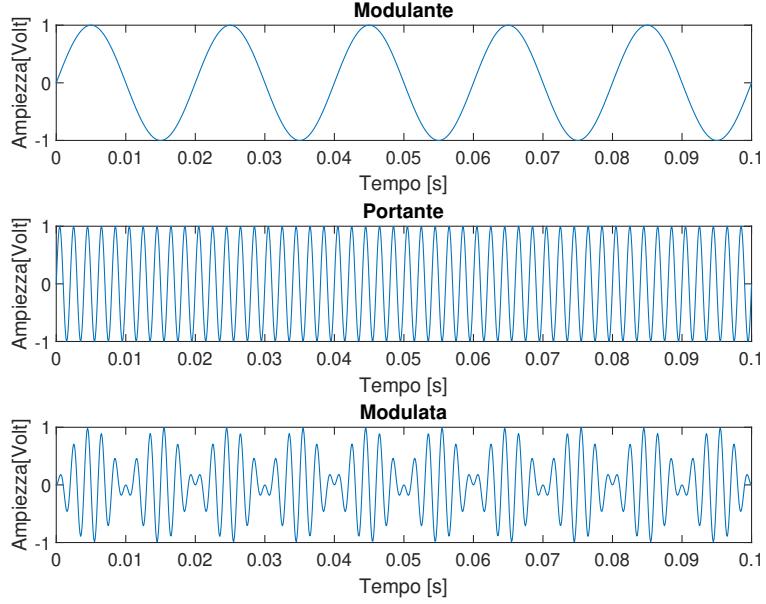
$$y(t) = A \cos(w_p t) + \frac{A \cdot m}{2} \cos(w_p - w_m)t \frac{A \cdot m}{2} \cos(w_p + w_m)t \quad (16)$$

che è un segnale composto da:

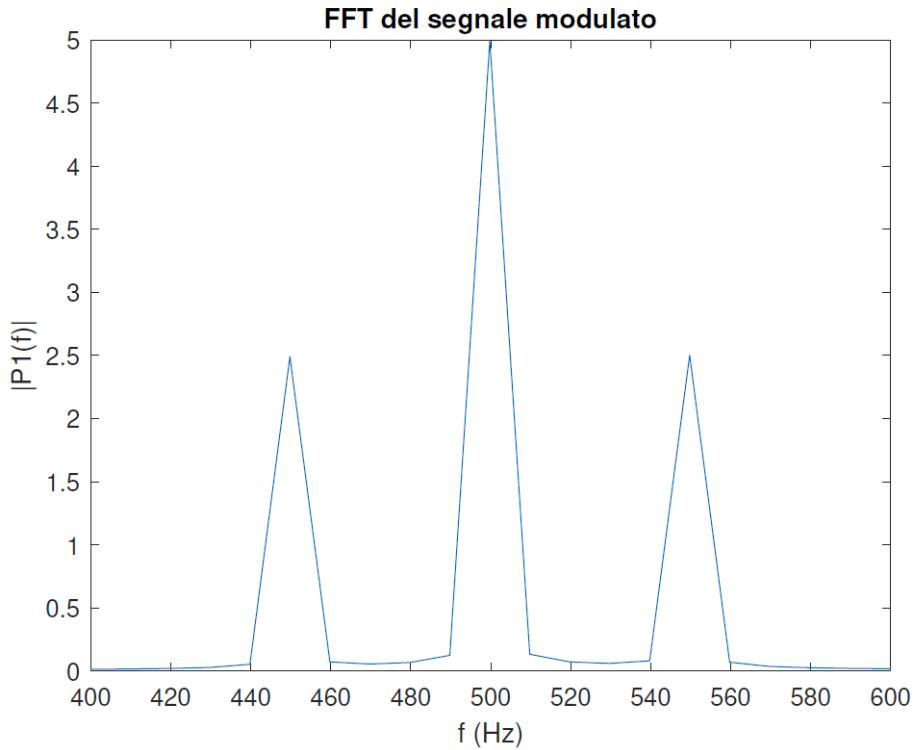
- Segnale portante ( $A \cos(w_p t)$ )
- Segnale laterale superiore ( $\frac{A \cdot m}{2} \cos(w_p + w_m)t$ )
- Segnale laterale inferiore ( $\frac{A \cdot m}{2} \cos(w_p - w_m)t$ )

Vediamo ora un esempio. Prendendo un segnale modulante con frequenza 50Hz e ampiezza 5, un segnale per la portante con frequenza 500Hz e ampiezza 5. Il segnale risultante presenterà sempre la stessa frequenza della portante ma in ampiezza varierà a seconda della modulante.

Figure 4:  $y = \sin(2 \cdot \pi \cdot 50 \cdot t)$



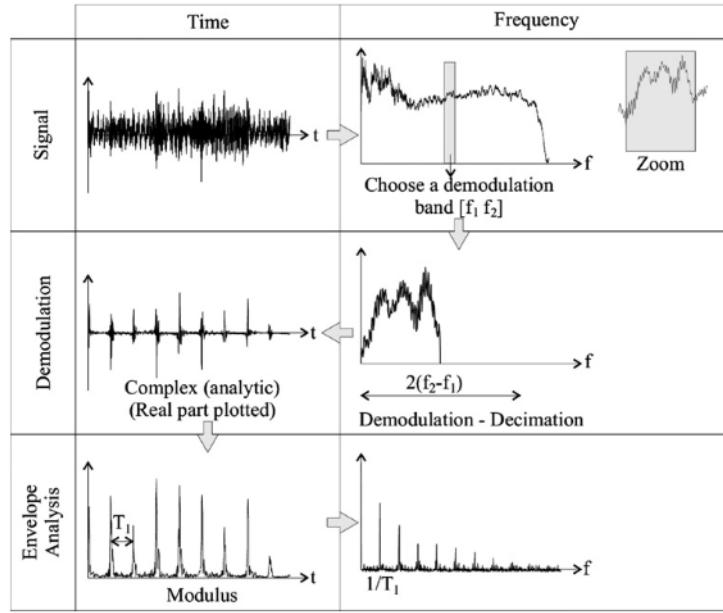
È utile andare a esaminare lo spettro del segnale modulato, dove è possibile notare come la frequenza portante sia affiancata da due componenti simmetriche (distanti la frequenza della modulante) e di ampiezza dimezzata rispetto alla modulante.



In questo semplice esempio, illustrato per chiarire il problema che vogliamo affrontare, è sufficiente vedere lo spettro e poter riconoscerne questa particolare forma.

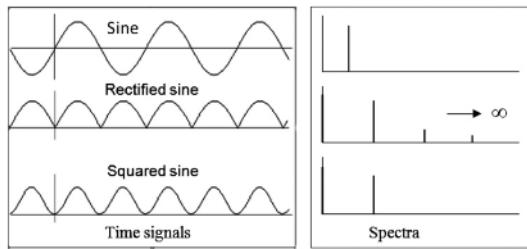
Il procedimento che viene utilizzato per l'analisi dell'inviluppo, in casi meno banali, è il seguente:

Figure 5: Schema dell'analisi dell'inviluppo secondo [5]



La trasformazione Tempo-Frequenza utilizzata è la trasformata di Hilbert. Dopo aver scelto la banda d'interesse  $[f_1, f_2]$ , in modo da isolare le frequenze tipiche del guasto da quelle che invece le mascherano sovrapponendosi, ai fini diagnostici risulta innanzitutto conveniente calcolare il quadrato dell'inviluppo [14] piuttosto che l'inviluppo in sè. Utilizzando infatti il quadrato dell'inviluppo è possibile evidenziare meglio lo spettro in frequenza e le sue caratteristiche, con ampiezze maggiori nelle frequenze di guasto teoriche che conosciamo a priori (qualora vi sia il guasto). Una dimostrazione intuitiva dell'utilità dell'utilizzo del quadrato dell'inviluppo è data dalla seguente immagine, tratta da [14]:

Figure 6: Aliasing che si potrebbe verificare[5].



dove si può notare che l'inviluppo di un segnale sinusoidale (rectified, visto che è l'operazione di modulo) presenta aliasing in frequenza, ovvero compaiono componenti ripetute che nello spettro del segnale non ci sono e che si ripetono all'infinito con un andamento esponenzialmente decrescente (servono infinite armoniche per rappresentarlo) mentre invece il quadrato dello stesso segnale (squared) non esibisce questo fenomeno di aliasing che vorremmo evitare.

L'aver elevato al quadrato l'inviluppo si traduce quindi in un raddoppio della banda  $B$ , il suo contenuto informativo.

L'aver utilizzato la trasformata di Hilbert ha dei risvolti pratici per la demodulazione in ampiezza con quello che viene definito *one-sided spectrum*, visto che ha solo componenti positive, è antitrasformato dando un segnale complesso nel tempo, che chiameremo segnale analitico. Questo è caratterizzato dall'avere come parte immaginaria la trasformata di Hilbert della sua parte reale e può essere scritto:

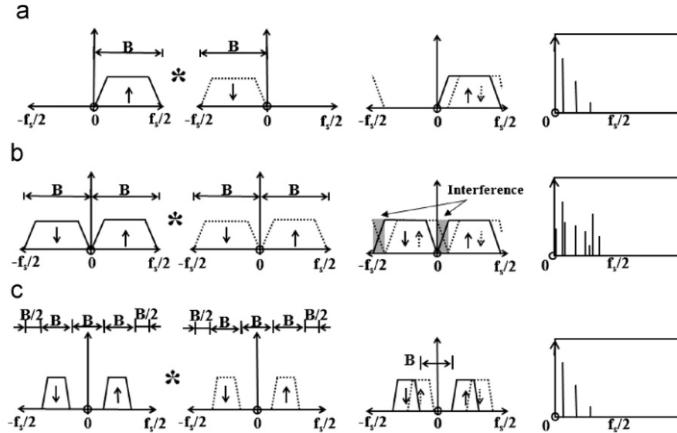
$$x(t) = a(t) + j \cdot b(t) = a(t) + j \cdot \tilde{a}(t) \quad (17)$$

dove con  $\tilde{a}(t)$  indichiamo la trasformata di Hilbert di  $a(t)$ . Quando il segnale è definito solo per frequenze positive, indicando con  $x_a(t)$  il segnale analitico, il suo inviluppo quadrato è dato dalla convoluzione dei rispettivi spettri. Quindi:

$$\mathcal{F}\{a(t) \cdot a^*(t)\} = \mathcal{F}\{a(f)\} * \mathcal{F}\{a^*(f)\} = X_a(f) * X_a^*(f) \quad (18)$$

Effettuare questa convoluzione può dare luogo a diversi risultati come schematizzato in figura. Analizzando la a) si può notare come nel caso di segnale analitico precedentemente definito si

Figure 7: Tre differenti casi di convoluzione: a) segnale analitico one-sided spectrum; b) one-side spectrum equivalente nel campo reale; c) two-sided spectrum; Le frecce discordi indicano la relazione di complesso coniugazione [5].



traduca in frequenze differenza, che contengono l'informazione della modulazione. Tuttavia in b) si può notare come ci sia oltre a quelle del caso precedente, ci siano anche delle componenti che mascherano l'informazione della modulazione. Nella fattispecie sono dovute alla presenza delle componenti negative in frequenza. Allo stesso tempo si può vedere però come in c) la presenza delle componenti negative non sia di intralcio nel caso in cui si dispongano attorno alle frequenze di Nyquist e che nelle altre frequenze sia effettuato un padding di zeri. Questo significa che effettivamente la frequenza di campionamento sia da raddoppiare per la stessa banda di demodulazione e che la dimensione della trasformata debba essere il doppio rispetto allo stesso problema. Tutte queste operazioni possono essere eseguite digitalmente. Infatti si può sostituire la trasformata di Hilbert con la FFT, vista l'uguaglianza delle due trasformazioni nel nostro contesto: segnale reale con assenza di frequenze negative. Come noto, in tal caso la trasformata di Hilbert di un segnale è possibile ottenerla partendo dalla parte reale (o immaginaria) della FFT del segnale.

Riassumendo il procedimento di analisi dell'inviluppo:

Acquisizione segnale grezzo	$v(t)$
Trasformata di Fourier	$v(t) \xrightarrow{\text{FFT}} V(f)$
Filtraggio banda $[f_1, f_2]$ con padding	$V(f) \rightarrow X(f)$
Anti-Trasformata di Fourier	$X(f) \xrightarrow{\text{IFFT}} x(t)$
Calcolo del Modulo	$a(t) =  x(t) $
Trasformata di Fourier	$ a(t)  \xrightarrow{\text{FFT}} A(t)$

Table 1: Schema di analisi dell'inviluppo

#### 2.2.4 L'algoritmo di Diagnosi per cuscinetti

L'algoritmo di diagnosi è semi-automatico, ovvero sono imposti solo pochi parametri, dettati dall'esperienza. Nel caso di velocità costante è possibile saltare l'analisi degli ordini, diversamente è necessaria per eliminare le variazioni di velocità. Qualora vi fossero meccanismi come ingranaggi che ostacolano la diagnosi, è necessario utilizzare i metodi proposti in letteratura (alcuni trattati in questo elaborato: AR, SANC, DRS, ecc..). È possibile eseguire alcuni filtri opzionali, come ad esempio il MED [15] che punta a migliorare l'analisi della scelta della frequenza a cui tagliare il segnale (ad esempio la Spectral Curtosi [16]). Infine l'ultimo step è l'analisi dell'inviluppo, tramite il quale la FFT esibisce le frequenze del guasto. Un esempio di questa procedura può essere trovato in bibliografia [17] ed è riassunto nell'immagine seguente:

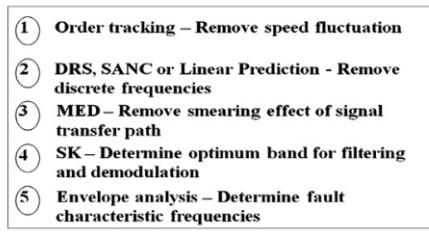


Figure 8: Esempio di algoritmo di diagnostica completo

### 2.3 Il Compressed Sensing

#### 2.3.1 Introduzione

A partire dai primi anni del 2000[18, 19] il Compressed Sensing(CS) ha attirato una considerabile attenzione visto il suo impiego nei più svariati campi, dalla matematica applicata, all'informatica, all'elettronica. L'innovazione di queste tecniche è la possibilità di sorpassare i limiti del campionamento classico, in quanto grazie a questi metodi che il CS utilizza si possono essere impiegate laddove non è possibile impiegare sensori classici. Per molto tempo infatti ci si è attenuti ai risultati di Kotelnikov, Nyquist, Shannon, and Whittaker sul campionamento di segnali a banda limitata, sinteticamente chiamato teorema di Shannon una volta formulato nella sua forma più completa nel 1949. Come è noto questo teorema afferma che per non perdere nessuna informazione e prevenire fenomeni di aliasing, un segnale a banda limitata deve essere campionato ad una frequenza non inferiore a due volte la sua frequenza massima. Le ripercussioni di questo teorema, che a tutti glie effetti è un paradigma di campionamento per i segnali la frequenza massima è molto alta sono intrattabile nemmeno con la crescita esponenziale delle performance dei nuovi hardware: il problema consiste nell'enorme mole enorme di dati frutto del campionamento, che rendono impossibile il problema persino per i sensori di acquisizione dei tempi futuri. Si rende quindi necessario una qualche modifica, accorgimento, al paradigma del campionamento classico e il Compressed Sensing è una risposta a questo problema. Molto spesso è trattato insieme al *sparse recovery* che è essenzialmente il nome per le tecniche di

risoluzione dei problemi formulati dal CS, tanto che sono diventati l'uno il sinonimo dell'altro.

### 2.3.2 Rappresentazioni diverse

I concetti introdotti in questa sezione e nelle prossime, sono molto generali infatti il termine "segna" verrà trattato come vettore generico, sia esso un segnale campionato nel tempo, un audio, un'immagine (vettore di pixel). In tutti questi casi il vettore preso in esame potrà essere scritto come vettore  $\mathbf{x} \in R^N$ , dove  $N$  è il numero delle componenti(o coordinate) del vettore, ovvero la sua dimensione. Rappresentando ogni componente con indici da 1 a  $N$  allora possiamo scrivere come segue:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_N \end{bmatrix}$$

È d'uso comune nell'elaborazione dei segnali esprimere lo stesso vettore tramite le coordinate di una base diversa, definendo appunto una rappresentazione diversa con componenti diverse. Molte possano essere le motivazioni, ad esempio si vuole catturare meglio alcune proprietà o avere facilità di computazione o memorizzazione di esso (idea fondamentale per ogni algoritmo di compressione, si pensi ad un'immagine e quanti pixel servano per memorizzarla). Si tratta quindi scegliere una opportuna collezione di  $N$  vettori  $N$ -dimensionali linearmente indipendenti  $\psi_1, \psi_2, \dots, \psi_N$ . Definendo  $\Psi$  come la matrice che ha in prima colonna il vettore  $\psi_1$ , in seconda colonna  $\psi_2$  e così via, per trovare le nuove coordinate sotto la nuova base che indicheremo con  $\{\psi_i\}_{i=1,2,\dots,N}$  dovremo quindi risolvere il sistema lineare così definito:

$$\Psi X = x \quad (19)$$

Una volta risolto il sistema lineare in  $\mathbf{X}$ , sarà possibile avere la rappresentazione di  $\mathbf{x}$  come combinazione lineare di  $\psi_1, \psi_2, \dots, \psi_N$ .

$$\mathbf{x} = \sum_{i=1}^N X_i \cdot \psi_i \quad (20)$$

Un esempio di base molto ricorrente in molti ambiti è la rappresentazione di Fourier, il cui sistema associato può essere risolto in maniera molto efficiente tramite l'algoritmo di FFT (che sta per Fast Fourier Transform) che ha complessità log-lineare con la dimensione del vettore da trasformare. Introduciamo quindi la matrice di Fourier  $F$  che ci servirà più avanti nella trattazione.

$$F_N = \begin{bmatrix} w_N^{0 \cdot 0} & w_N^{0 \cdot 1} & \dots & w_N^{0 \cdot (N-1)} \\ w_N^{1 \cdot 0} & w_N^{1 \cdot 1} & \dots & w_N^{1 \cdot (N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ w_N^{(N-1) \cdot 0} & w_N^{(N-1) \cdot 1} & \dots & w_N^{(N-1) \cdot (N-1)} \end{bmatrix} = [w_N^{i,j}]_{i,j=0,1,\dots,N-1} \quad (21)$$

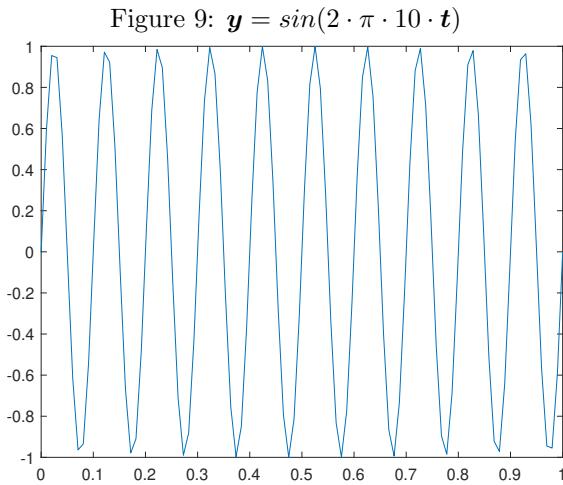
Questa matrice è molto particolare in quanto è anche ortonormale a meno del fattore  $\frac{1}{N}$ , ovvero che la sua inversa è pari a:

$$\mathbf{F}_N^{-1} = \frac{1}{N} \cdot \mathbf{F}_N^* \quad (22)$$

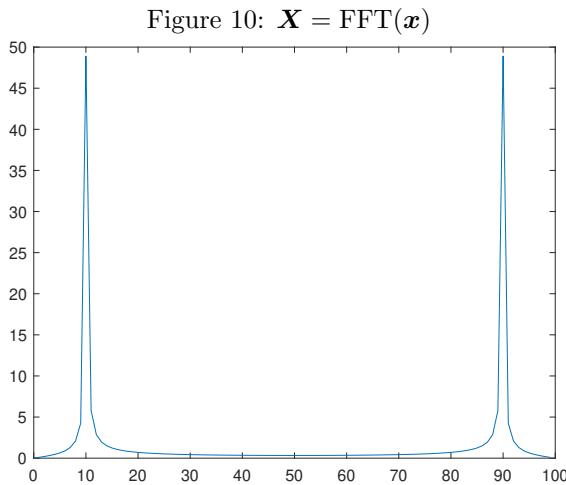
dove  $\star$  indica l'operazione di trasposto coniugato.

### 2.3.3 Sfruttare la sparsità

Un segnale è sparso quando la maggior parte dei suoi coefficienti è pari a zero. In natura non esistono segnali sparsi ma ingegneristicamente esistono segnali che rappresentati con una base opportuna sono approssimabili con un numero esiguo di coefficienti (rispetto alla dimensione del vettore originale). Vediamo un esempio molto semplice: un segnale  $\mathbf{x}$  sinusoidale nel tempo di frequenza 10Hz campionato a 30Hz tra 0s e 1s, genera i 30 campioni in figura:



Come è noto, eseguire la FFT di questo segnale genera un vettore  $\mathbf{X}$  della stessa dimensione di  $\mathbf{x}$  ma che, se rispettiamo il teorema di Shannon-Nyquist precedentemente enunciato presenterà soli due coefficienti in ampiezza diversi da 0: quelli in corrispondenza delle frequenze di 10Hz e di 90Hz.



I segnali sono lo stesso ma espressi tramite due basi diverse: quella canonica e quella di Fourier.

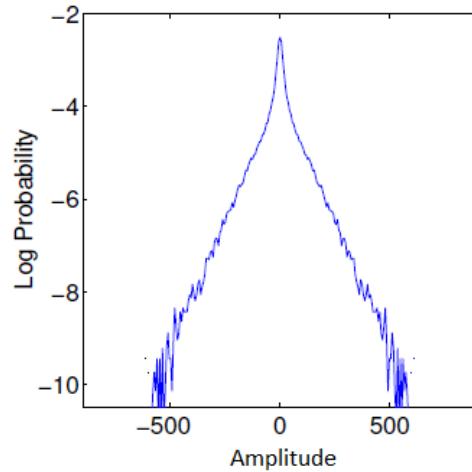
La differenza sta che usando il tempo per descrivere il segnale servano 30 campioni mentre nel dominio delle frequenze due coefficienti (immaginari). Molti esempi possono essere fatti, utilizzando matrici anche più complesse ma il concetto generale di fondo è lo stesso. Si può fare infatti un discorso simile per immagini in cui essenzialmente si utilizza una codifica matematica di un vettore di pixel per permettere di avere pochi coefficienti. Lo standard di compressione JPEG utilizza la trasformata del coseno ad esempio, sotto cui le immagini risultano essere sparse rispetto a una canonica memorizzazione per pixel. Ad esempio esaminando l'immagine seguente tratta da [20]:

Figure 11: Immagine di barche



Utilizzando dei coefficienti Wavelet, senza dilungarsi nella spiegazione di come sia fatta, che caratteristiche abbia e come i suoi coefficienti siano calcolati, possiamo andare a valutare l'ampiezza dei coefficienti in un log-istogramma in figura: l'obiettivo di questo grafico sarà avere un'idea intuitiva sulla la distribuzione delle ampiezze dei pixel che la compongono. Si può notare come i coefficienti di maggior ampiezza (in modulo) siano estremamente ridotti rispetto a quelli nulli e/o quasi, in quanto i primi sono caratterizzati da probabilità estremamente più basse.

Figure 12: Log-Istogramma dei coefficienti Wavelet della figura 11, tratto da [20].



L'immagine infatti utilizzando questi coefficienti è ben approssimabile con il 5% dei coefficienti

totali, un notevole vantaggio per la memorizzazione di esso.

Introduciamo ora il concetto di proiezione random che ci servirà per la trattazione. Sono essenzialmente tecniche per ridurre la dimensione di problemi, vengono utilizzate con l'intento di verificare certe proprietà su insiemi di dati più piccoli per problemi di dimensioni elevate per cui un approccio tradizionale fallisce. Una di queste tecniche prevede l'utilizzo di matrici  $n \times N$  con  $n \ll N$  che sintetizzano il contenuto di matrici(o vettori) in strutture più piccole, caratterizzate da facilità di computazione. Alcuni esempi sono:

- Matrice di Random Sample: È composta da zeri e uni, in particolare su ogni riga c'è un solo uno e la posizione di questo non viene ripetuta in nessuna altra riga. Estraе un numero casuale di righe da un vettore/matrice, motivo per cui ha questo nome. Un esempio è la matrice  $3 \times 4$  seguente:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (23)$$

che estrae la prima, la seconda e la quarta riga di un vettore 4-dimensionale(o nel caso di matrice le righe).

- Matrice Gaussiana: È popolata di numeri distribuiti come una normale. Moltiplicandola per un vettore sintetizzerà in  $n$  righe gli  $N$  coefficienti del vettore, eseguendo una somma pesata da coefficienti distribuiti come una gaussiana (simil discorso nel caso di una matrice).
- Matrice di Bernoulli: I suoi elementi appartengono ad un insieme di cardinalità 2 e ciascuno ha la sua probabilità(da cui il nome).

Il concetto di sparsità e proiezione random sono stati introdotti perchè sotto le opportune ipotesi è possibile sfruttarli per limitare il numero di campioni per un corretto campionamento di segnali.

Supponiamo ora che  $\mathbf{y} \in R^N$  sia un vettore incognito che vorremmo conoscere. Ammettendo di avere una sua proiezione random  $\mathbf{b}$  di dimensione  $n \times N$  data da C, dove  $n \ll N$ , tale che:

$$\mathbf{b} = C\mathbf{y} \quad (24)$$

vorremmo in un qualche modo ricavare il vettore  $\mathbf{y}$  incognito originale. Analizzando il sistema (24) si può notare come sia sottodeterminato in quanto il numero di equazioni sia  $n$  mentre il numero di incognite è  $N$ , con  $n \ll N$ . Siccome una soluzione esiste(nella realtà abbiamo a disposizione il vettore  $\mathbf{b}$  che deriva dai sensori che hanno fatto l'operazione  $C\mathbf{y}$  per noi), allora il sistema sottodeterminato presenterà infinite soluzioni che soddisfano (24). Essenzialmente ora serve qualche artificio matematico che permetta la ricostruzione completa del vettore  $\mathbf{y}$  originale tra gli infiniti possibili vettori consistenti con i dati. È qui che si può sfruttare il concetto di sparsità: ammettendo che il vettore  $\mathbf{y}$  sia sparso su una conveniente base  $\Psi$ , la soluzione al

problema sottodeterminato (24) può essere trovata come la soluzione più sparsa di tutte quelle ammissibili. Ciò è possibile perché i segnali di interesse ingegneristico sotto opportune basi risultano ben approssimabili come sparsi e la soluzione più semplice sotto questo punto di vista risulta essere quella che si sta cercando. Introducendo la misura per la sparsità di un vettore, ovvero la  $\ell_0$ -norma definita:

$$\|\mathbf{s}\|_0 = \sum_{i=1}^N \mathbb{1}_i \quad (25)$$

dove  $\mathbb{1}_{i_0}$  è la funzione indicatrice definita come segue:

$$\mathbb{1}_i = \begin{cases} 1 & \text{se } s_i \neq 0 \\ 0 & \text{se } s_i = 1 \end{cases} \quad (26)$$

si può definire il seguente problema di ottimizzazione:

$$\begin{aligned} \arg \min_{\mathbf{x}} \|\mathbf{x}\|_0 \\ \text{subject to } A\mathbf{x} = \mathbf{b} \end{aligned} \quad (27)$$

Per come è posto il problema di ottimizzazione (27), la strategia di ricerca più efficiente sarebbe di tipo brute-force, non avendo informazioni a priori su  $\mathbf{s}$ . Iterando su ogni possibile K-sparsità si proietterebbe il segnale  $\mathbf{s}$  sulla possibile base scelta. Purtroppo la complessità di questo algoritmo è esponenziale, NP-hard. Essenzialmente (27) è intrattabile e lo sarà sempre.

È possibile rilassare il problema di ottimizzazione (27) ad una minimizzazione che ha la particolarità di essere convessa:

$$\begin{aligned} \arg \min_{\mathbf{x}} \|\mathbf{x}\|_1 \\ \text{subject to } A\mathbf{x} = \mathbf{b} \end{aligned} \quad (28)$$

dove  $\|\cdot\|_1$  è la  $\ell_1$ -norma, data da:

$$\|\mathbf{s}\|_1 = \sum_{i=1}^N |s_i|$$

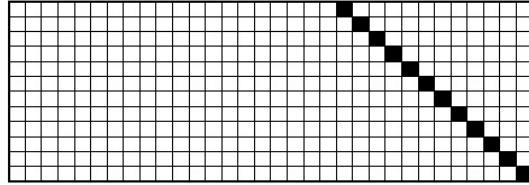
La  $\ell_1$ -norma è conosciuta come norma Manhattan. Quello che ora è il sistema, è un problema di ottimizzazione lineare. Il rilassamento è possibile solo se certe condizioni sono verificate per la matrice C:

- La matrice C deve essere incoerente rispetto alla base  $\Phi$ , ovvero non vi deve essere correlazione tra le righe di C e le colonne di  $\Phi$ .
- Il numero di misure  $p$  deve essere sufficientemente ampio, dell'ordine di:  $O(K \log(n=K))$

Sotto queste ipotesi, che congiunte prendono il nome di *Restricted Isometric Property*, la soluzione per (28) converge alla soluzione (28) con alta probabilità. In realtà la R.I.P. property sarebbe da verificare per ogni coppia di matrice delle misure e di proiezione, ma ai fini pratici le matrici random(3 delle quali presentate prima) con buona probabilità la verificano, per ulteriori

dettagli si veda in bibliografia [21]. Quello che è opportuno sottolineare deve essere che il nuovo campionamento (randomico o con peso dei coefficienti random, come abbiamo visto) deve essere sufficientemente incoerente per l'effettivo funzionamento del framework di cui abbiamo parlato. Se ad esempio si volesse violare questa condizione, ad esempio volendo ottenere il vettore  $\mathbf{y}$  in  $R^N$  utilizzando una matrice  $C$  come in figura: ovvero la matrice identità  $p \times p$  affiancata

Figure 13: Possibile scelta per C che si rivelerrebbe inopportuna



a sinistra da un numero  $N - p$  zeri. Allora qualsiasi segnale  $\mathbf{y}$  che non abbia componenti in queste ultime  $p$  colonne sarebbe completamente invisibile alle misure. Intuitivamente quello che deve succedere è che le misure di cui disponiamo distribuiscano l'uniforme il più possibile. L'ultima cosa che rimane da sottolineare è come ruolo di  $n$  e la sparsità del segnale influisca le prestazioni di questi metodi. Dalla formulazione matematica della R.I.P. property si può derivare come tanto più un segnale sia sparso meno misure serviranno per il corretto recovery del segnale. D'altro canto nel caso in cui il segnale sia ricco di informazioni sotto una certa base, allora sarà necessario alzare il numero di  $n$ . Idealmente infatti qualora si trattasse una base  $\Psi$  che non rende significativamente più sparso il segnale ci si ritroverebbe a risolvere un sistema lineare  $N \times N$  equivalente al problema di partenza.

## 2.4 Gli Algoritmi di ottimizzazione

Rimane adesso da risolvere il problema di ottimizzazione generico (28). Infatti ricordando:

- la formula della  $\ell_1$ -norma per un vettore  $\mathbf{x}$ :

$$\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|$$

- Il problema di ottimizzazione:

$$\begin{aligned} & \arg \min_t \mathbf{1}^T \mathbf{t} \\ & \text{subject to } A\mathbf{x} = \mathbf{b} \\ & |x_i| \leq t_i \quad \forall i \end{aligned}$$

Ponendo  $|x_i| \leq t_i$  è possibile scrivere:

$$\begin{aligned} & \arg \min_t \mathbf{1}^T \mathbf{t} \\ & \text{subject to } A\mathbf{x} = \mathbf{b} \\ & |x_i| \leq t_i \quad \forall i \end{aligned}$$

Siccome  $|x_i| \leq t_i \iff x_i \leq t_i, x_i \geq -t_i$  allora:

$$\begin{aligned} & \arg \min_t \mathbf{1}^T \mathbf{t} \\ & \text{subject to } A\mathbf{x} = \mathbf{b} \\ & \quad x_i \leq t_i \forall i \\ & \quad x_i \geq -t_i \forall i \end{aligned}$$

che può essere scritto:

$$\begin{aligned} & \arg \min_t \mathbf{1}^T \mathbf{t} \\ & \text{subject to } A\mathbf{x} = \mathbf{b} \\ & \quad I\mathbf{x} - It \preceq \mathbf{0} \\ & \quad -I\mathbf{x} - It \preceq \mathbf{0} \end{aligned}$$

che è un problema di *linear programming*. Aver riformulato il problema (27) in uno lineare, che prende il nome di *Basis Pursuit*, ha il vantaggio computazionale di rendere il problema in primis trattabile e affrontabile con gli algoritmi che il *linear programming* mette a disposizione. Infatti l'algoritmo di ottimizzazione con questo rilassamento lavora in un insieme convesso (vincoli solo lineari) per cui è possibile utilizzare gli algoritmi di ottimizzazione conosciuti: interior-points method, projected gradient method e iterative thresholding. Per maggiori dettagli sugli algoritmi di ottimizzazione consultare la bibliografia [22].

È possibile seguire anche un altro tipo di approccio per risolvere il problema (28), ovvero la risoluzione greedy. Questi algoritmi si differenziano da quelli iterativi caratteristici del linear programming per la velocità di computazione molto efficiente e della costruzione di una soluzione fattibile sub-ottima e in particolare generalmente inferiore in qualità rispetto a quella ottenibile con la risoluzione rigorosa iterativa.

Alcuni algoritmi sono i seguenti:

- Matching Pursuit(MP): Reiterà più volte la proiezione del segnale su tutti i vettori che compongono la matrice A, costruendo passo per passo una soluzione con i coefficienti di quei vettori che in modulo hanno la proiezione più grande.
- Orthogonal Matching Pursuit(OMP): Versione del MP dove A è ortonormale e vengono modificati ad ogni passo tutti i coefficienti che compongono la soluzione. Offre migliori risultati ma a un prezzo computazionale maggiore.
- Stagewise OMP (StOMP): Anche questo algoritmo è una variante del MP che utilizza un criterio di thresholding per la selezione degli indici delle componenti. [23].
- CoSAMP: Utilizza l'hard thresholding per la selezione delle M(parametro dell'algoritmo) componenti del vettore da cercare tramite l'applicazione della pseudoinversa [24].
- Generalized OMP (gOMP): Generalizzazione dell'OMP, invece che selezionare un solo componente alla volta ne seleziona anche altri [25].

---

**Algorithm 1:** Matching Pursuit

---

```

1 Input matrice  $A$  con colonne normalizzate  $c_i$ , segnale  $s$ ;
2  $R_1 \leftarrow s$ ;
3  $n \leftarrow 1$  ;
4 repeat
5   Find column  $c \in A$  with maximum inner product  $\|\langle c, s \rangle\|$  ;
6    $a_n \leftarrow \langle R_n, c \rangle$  ;
7    $R_{n+1} \leftarrow R_n - a_n c$  ;
8    $n \leftarrow n + 1$  ;
9 until  $\|R_n\| < threshold$ ;

```

---

- Multipath Matching Pursuit (MMP): Affronta il problema con una ricerca in un albero con algoritmi greedy [26].

#### 2.4.1 Ulteriori Formulazioni

I segnali del mondo reale sono caratterizzati dalla presenza di rumore. Un approccio come quello appena trattato, che risolve l'uguaglianza tra  $Ax$  e  $b$  osservato, senza discernere dal segnale d'interesse il rumore che si sovrappone. La soluzione a questo problema è cercare formulazioni in cui la soluzione è un trade-off tra fit e sparsità della soluzione, premiando le soluzioni sparse a discapito di quelle che si adattano meglio ai dati. Questo perchè la presenza del rumore non fa altro che peggiorare la sparsità della soluzione trovata, identificando più coefficienti di quelli che la soluzione pulita senza rumore indentificherebbe. Da qui quindi la necessità di qualche modo per regolarizzare la soluzione trovata. È possibile regolarizzare la soluzione formulando il problema in maniera diversa, una prima via è modificare la funzione da minimizzare del problema (27) promuovendo la sparsità della soluzione. Mantenendo la forma di minimizzazione, si può quindi pensare di aggiungere un parametro regolarizzatore positivo che si sottrae all'errore quadratico medio e viene moltiplicato per la sparsità del vettore  $x$ . Scrivendo in forma matematica allora:

$$\begin{aligned} & \underset{\mathbf{x}}{\text{minimize}} \quad \frac{1}{2} \|A\mathbf{x} - \mathbf{b}\|_2^2 + \alpha \|\mathbf{x}\|_1 \\ & \text{subject to} \quad A\mathbf{x} = \mathbf{b} \end{aligned} \tag{29}$$

Si può quindi notare come il trade-off sparsità-errore quadratico medio della soluzione sia dettato da  $\alpha$  che interviene a penalizzare soluzione complesse e poco sparse che si adattano ai dati rumorosi. Esiste un'altra formulazione per questo problema equivalente ed è la seguente con  $\sigma > 0$ :

$$\begin{aligned} & \underset{\mathbf{x}}{\text{minimize}} \quad \|\mathbf{x}\|_1 \\ & \text{subject to} \quad \|A\mathbf{x} - \mathbf{b}\|_2^2 \leq \sigma \end{aligned} \tag{30}$$

Con questa formulazione che è molto simile, che minimizza la stessa funzione del (27) non si risolve più l'uguaglianza stretta ma si permette alla soluzione di non soddisfarla entro una

certa soglia  $\sigma$ . Si può dimostrare che le due formulazioni sono equivalenti, ovvero (29) e (36), per  $\alpha$  e  $\sigma$  opportuni, ammettano la stessa soluzione. In generale per risolvere questi problemi di ottimizzazione, che prendono il nome di Basis Pursuit Denoising(BPDN), persa la linearità della forma (28), è necessario utilizzare algoritmi di risoluzione per problemi convessi quadratici come i metodi dei punti interni (*interior-points*) che sono descritti in [27]. Ne esistono anche altri più efficienti, soprattutto quando si trattano problemi ampi.

Un altro modo di regolarizzare la soluzione in caso di rumore è per esempio vincolare la sparsità della soluzione. Ad esempio imporre come vincolo che la  $\ell_1$ -norma del vettore sia al più  $\tau$ . In tal caso il parametro va a "costringere" la soluzione a presentare una  $\ell_1$ -norma che al più vale  $\tau$ . Supponendo  $\tau > 0$ , si può quindi scrivere:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & \frac{1}{2} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 \\ \text{subject to} \quad & \|\mathbf{x}\|_1 \leq \tau \end{aligned} \tag{31}$$

Questa formulazione prende il nome di *LASSO* [28] dove viene anche descritto l'algoritmo per la risoluzione di un problema formulato così.

Quelli visti fin ora sono tutte risoluzioni iterative che migliorano ad ogni passo la soluzione. Esistono però anche approcci euristici che trovano soluzioni sub-ottime. I vantaggi di queste soluzioni sono sicuramente l'efficienza visto che utilizzano euristiche e possono trovare impiego laddove magari si progetti un algoritmo iterativo di raffinamento della soluzione, in quanto è possibile utilizzare come soluzione iniziale da migliorare quella trovata da un metodo euristico. Oltre a quelli introdotti nella sezione precedente (OMP e le sue varianti) si può utilizzare ad esempio In-Crowd [29] che risolve il BPDN su problemi iterativamente più grandi:

---

**Algorithm 2:** In-Crowd

---

- 1** **Input** matrice  $A$  con colonne normalizzate  $c_i$ , segnale  $s$ ;
  - 2**  $x \leftarrow 0$ ;
  - 3**  $r \leftarrow s$  ;
  - 4**  $I^c \leftarrow \emptyset$ ;
  - 5** Compute "usefulness"  $u_j = |\langle r, s \rangle|$  for each  $I^c$ ;
  - 6** If on  $I^c$ , no  $u_j > \alpha$  terminate ;
  - 7** Otherwise add L components to  $I^c$  based on their usefulness ;
  - 8** Solve BDPN on  $I^c$  ;
  - 9** Update  $r = y - Ax$  ;
  - 10** Go to step 5 ;
- 

Come si può vedere,  $\alpha$  è il parametro di regolarizzazione per il generico problema di BPDN che viene affrontato più volte su insiemi più grandi di colonne di  $A$ . Un altro algoritmo è per esempio il BLITZ [30] che divide il problema in sottoproblemi utilizzando una tecnica simile a quella dell'ottimizzazione lineare (si serve dei constraints attivi).

Sono possibili ulteriori formulazioni, ad esempio una è il Dantzing Selector (DS) [31] che è

formulata come segue:

$$\min_{\beta \in \mathbf{x}^p} \|x\|_{\ell_1} \quad \text{subject to} \quad \|X^*r\|_{\ell_\infty} \leq (1 + t^{-1})\sqrt{2 \log p} \cdot \sigma,$$

#### 2.4.2 Lista di solver sperimentati

Sono disponibili su MATLAB svariate librerie open-source che implementano gli algoritmi per la risoluzione delle varie formulazioni che si fare per i problemi del CS.

- SPGL1 [2] (che verrà usato per i risultati di questo elaborato)
- L1-Magic [32]
- CVX [33]
- YALL1 [34]



### 3 Setup Sperimentale

Questa sezione tratterà la presentazione del macchinario, fornendo in primis una descrizione e successivamente elencando i sensori presenti sul macchinario e le relative caratteristiche.

#### 3.1 Descrizione della macchina

Il lavoro è stato effettuato su una macchina Spark X 2100 e consiste in un centro di lavoro sviluppato da Mandelli Sistemi per mercati come possono essere l'aeronautico e l'energetico, caratterizzati da materiali di difficile lavorazione. La particolarità di questo macchinario è la precisione con cui opera anche in condizioni di applicazione di coppie elevate. Il sistema è comprensivo anche di un encoder ottico ad alta definizione, due coppie di cuscinetti ceramici molto precisi e bilanciati dinamicamente.

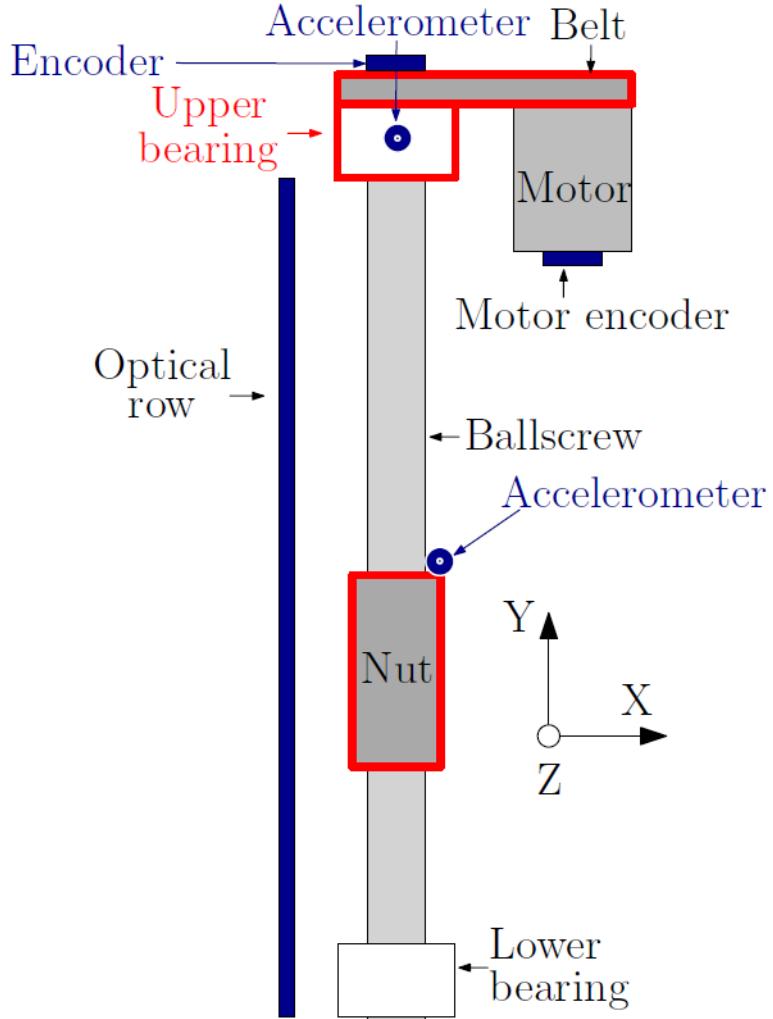
Figure 14: Spark 2100 [1]



#### 3.2 Descrizione dell'asse Y

È stato considerato solo l'asse Y della macchina, rappresentato in figura 15.

Figure 15: Asse Spark 2100 [1]



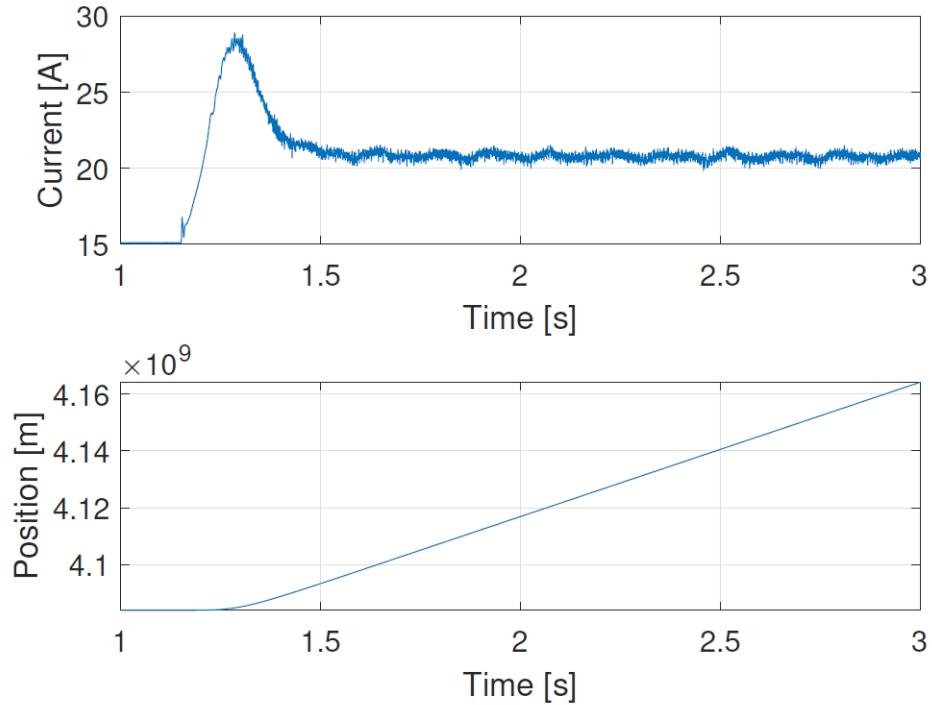
Il sistema è composto da un motore BLDC trifase collegato all'asse verticale tramite una cinghia, 4 cuscinetti assiali NSK sono posti tra cinghia e vite, in un package per evitare il contatto con l'esterno. Ogni cuscinetto è composto da due anelli, due corone di sfere e due gabbie. La vite a ricircolo di sfere dispone di una corsa massima di 1500mm, fino al cuscinetto inferiore. Per la diagnosi dei guasti si è deciso di concentrarsi su alcuni componenti, considerati più critici (evidenziati in rosso in figure 15) ovvero: cuscinetto superiore, vite a ricircolo di sfere e cinghia. Sulla base di questa scelta infatti sono stati posizionati "sensori extra". Questi componenti infatti risultano quelli più soggetti all'usura pertanto è utile prenderli in considerazione come casi studio per verificare lo stato di salute della macchina. La macchina CNC (calcolo numerico computerizzato) dispone già di alcuni sensori utilizzati per misurare l'andamento delle grandezze fisiche legate ad esso (ad esempio le correnti di fase, encoder, riga ottica, ecc.). Questi file venivano salvati in formato .xml sulla console della macchina e sono stati estratti con chiavetta USB. La macchina ha un limite della dimensione del file di salvataggio che dipende dal numero di variabili salvate e della frequenza di campionamento, rendendo quindi necessario l'esecuzione di più prove per un campionamento completo delle grandezze in un intervallo pre-

fissato. Delle molteplici variabili che si potevano salvare dal CNC sono state scelte le seguenti come le più indicate da analizzare:

- Corrente in quadratura del motore(IQ).
- Encoder del motore, prima della trasmissione.
- Riferimento dell'encoder.
- Riga ottica.

Dopo alcune prove si è deciso di utilizzare come tempo di campionamento  $T_s=3\text{s}$  a  $f_s=4000\text{Hz}$  per la singola prova, un esempio di queste variabili(nella fattispecie IQ e encoder) può essere trovato in figura 16. Sono però necessarie più prove per ogni variabile in quanto la memoria del buffer dell'azionamento si esaurisce.

Figure 16: Esempio variabili estratte dal CNC con  $T_s=3\text{s}$  e  $f_s=4000\text{Hz}$  [1]



### 3.3 Sistemi di acquisizione dati

Il sistema di acquisizione dati è il National Instruments CompactDAQ, che presenta moduli per i relativi sensori (sui vari assi, temperatura, corrente...). È stato collegato ad un pc con Labview per acquisire i vari dati.

Figure 17: Chassis NI CompactDAQ per i moduli di I/O

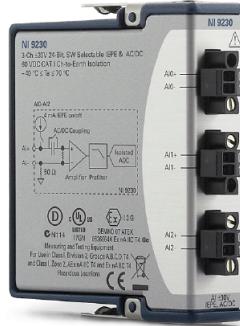


### 3.3.1 cDAQ

Per quanto concerne l'acquisizione e l'elaborazione dei segnali, è stato utilizzato un *National Instruments CompactDAQ*. Il modello impiegato è il CDAQ-9184: chassis che si collega al pc tramite USB e integra più moduli I/O che forniscono connettività diretta ai sensori applicati alla macchina. I moduli presenti sono i seguenti:

- Modulo NI-9230 in ingresso sound e vibration, a 3 canali 12.8kS/s per canale,  $\pm 30V$ . È stato utilizzato per il segnale accelerometrico sui 3 assi.

Figure 18: NI-9230



- Modulo NI-9211 di input temperatura C Series a 4 canali, 14 S/s aggregato,  $\pm 80mV$ . Sono inclusi filtri di *anti-aliasing*, rilevamento a termocoppia aperta e compesazione del giunto freddo per misure di termocoppia ad accuratezza elevata. Nella pagina successiva è possibile vedere il modulo, in figura 19.
- Modulo digitale NI-9401 per ingresso o uscita in incrementi a 4-bit (8 canali bidirezionali), con 5V/TTL, 100 ns; questo modulo in ingresso presenta il segnale dell'encoder interno al motore. È possibile vedere il modulo alla pagina successiva in figura 3.3.1.



Figure 19: Modulo NI-9211



Figure 20: Modulo NI-9401

### 3.3.2 Encoder

L'encoder utilizzato è un ROD 486 (in figura 21) in concomitanza col modulo NI-9411 e al cDAQ. È un encoder di tipo incrementale e costituito da due linee di impulsi (che vengono identificate con la lettera A e la lettera B rispettivamente) le quali sono sfasate di 90°. Un terzo canale di impulsi (chiamato R) serve per la trasmissione dell'impulso di riferimento per ogni rotazione completa. I segnali in uscita sono sinusoidali e perciò l'encoder e il cDAQ è su un circuito raddrizzatore TTL per convertire i segnali in onde quadre. La posizione dell'encoder è stato montato sul lato vite dopo la cinghia ma prima della chiocciola.

### 3.3.3 Termocoppia

Per quanto riguarda la termocoppia, questa è un trasduttore di temperatura basato sull'effetto *Seebeck*, ovvero in presenza di una differenza di temperatura produce elettricità. Nel particolare ci sono due conduttori generalmente di natura diversa e saldati insieme ad un'estremità chiamato giunto caldo. Una volta che il punto di misura viene riscaldato si misura la tensione sulle estremità del cavo (giunto freddo). È proprio questa differenza di potenziale che permette di generare elettricità. Tale tensione, chiamata Forza Elettro Motrice (FEM), viene prodotta



Figure 21: Encoder ROD 486

grazie alla diversa densità di elettroni dei due conduttori in metallo diversi del cavo impiegato, in combinazione con la differenza di temperatura tra il punto di misura e il giunto freddo. La termocoppia viene utilizzata insieme al modulo NI-9211 ed è montata nella vite e nel cuscinetto superiore.



Figure 22: Termocoppia

### 3.3.4 Accelerometro

L'accelerometro impiegato è il modello Dytran della serie KS903-10 IIEPE( Integrated Electronics Piezo-Electric) mostrato in figura 23. È del tipo *charge mode* ovvero genera un'uscita elettricamente proporzionale all'accellerazione rilevata. Sono presenti 4 pin d'uscita X,Y,Z e GND. Per posizionarlo in generale è possibile utilizzare materiale adesivo oppure un contatto magnetico oppure utilizzare una vite. Il modo in cui viene applicato influenza il range di frequenze misurabili dall'accelerometro. Le specifiche del sensore sono le seguenti:

- Sensitività  $100 \frac{mV}{g}$ .
- Intervallo di misure  $\pm 50g$ .
- Intervallo di frequenze ( $\pm 5\%$ ) $0.5-3000$  Hz..

Figure 23: Accellerometro



In questo elaborato è stato utilizzato l'accelerometro montato sul cuscinetto superiore ed era disponibile anche quello sulle vite a ricircolo di sfere, collegati al modulo NI-9230, come mostrato in figura 24.

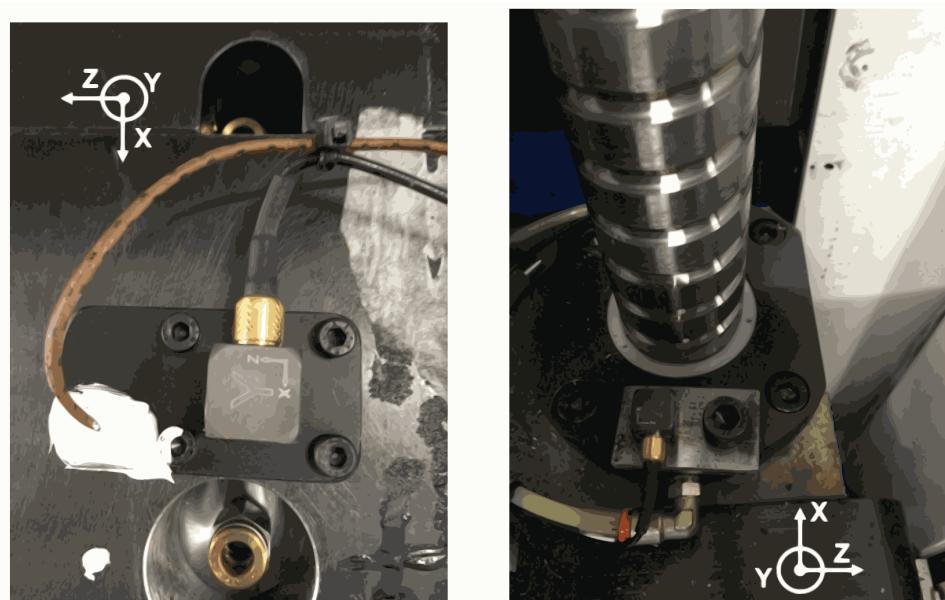
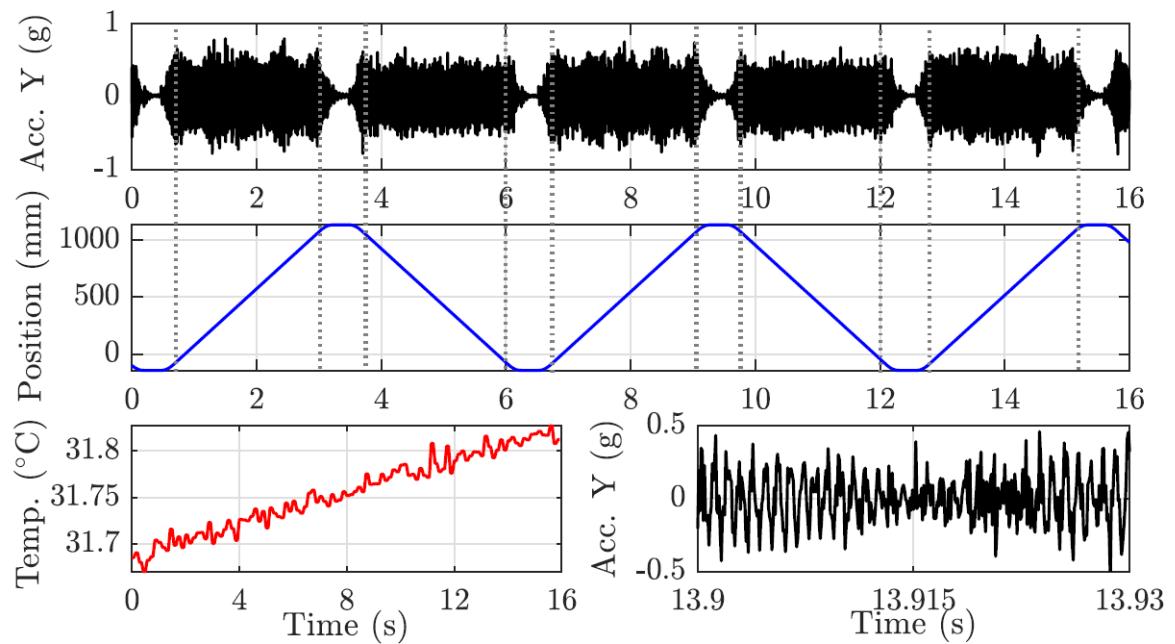


Figure 24: Sensori accelerometrici a disposizione. A sinistra: accellerometro e termocoppia posto sul package dei cuscinetti superiore. A destra: accelerometro posto sulla vite a ricircolo di sfera [1]

Componente	Variabile Misurata	Sensore	HW di acquisizione	Frequenza campionamento
Cuscinetto	Accelerazione	Accelerometro triassiale Dy-tran KS903.100	NI CDAQ-9184 Modulo NI9230	12800Hz
Cuscinetto	Temperatura	Termocoppia tipo J	NI CDAQ-9184 + Modulo NI9211	14 Hz
Vite a ricircolo di sfere	Accelerazioni	Accelerometro triassiale Dy-tran KS903.100	NI CDAQ-9184 + Modulo NI-9230	12800 Hz
Vite a ricircolo di sfere	Posizione angolare	Encoder dopo trasmissione cinghia, Heidenhain ROD 486 + raddrizzatore	NI CDAQ-9184 + Modulo NI9411	12800 Hz
EMA	Corrente IQ	Interno ad azionamento EMA	Azionamento EMA	4000 Hz
EMA	Posizione angolare	Riga ottica	Azionamento EMA	4000 Hz
Vite a ricircolo di sfere	Posizione lineare	NI CDAQ-9184 + Modulo NI-9411	Azionamento EMA	4000 Hz
EMA	Riferimento Posizione angolare	Nessuno	Azionamento EMA	4000 Hz

Table 2: Tabella riassuntiva sui sensori a disposizione.

Figure 25: Esempio rilevazioni sensori [1]





## 4 Risultati

In questa sezione verranno commentati i risultati del lavoro eseguito, passo per passo. Si partirà ovviamente dall'acquisizione dei dati e si procederà con l'esecuzione dell'algoritmo di diagnostica classico rivisitato con l'utilizzo del CS.

### 4.1 Acquisizione dati

Ricordando che il difetto preso in esame è il pitting sulla pista interna, l'esecuzione dell'algoritmo discusso nella precedente sezione, che per brevità chiameremo l'algoritmo di diagnosi classico, identifica componenti aggiuntive in frequenza multiple di 192Hz sui nostri dati a disposizione, in accordo con le frequenze teoriche calcolate a partire dalle formule teoriche(considerando che le prove sono effettuate con velocità 30000mm/min), come verrà mostrato nelle prossime righe. Mostriamo ora quindi, fatte le premesse su cosa ci stiamo apprestando a fare, i dati accelerometrici disponibili.

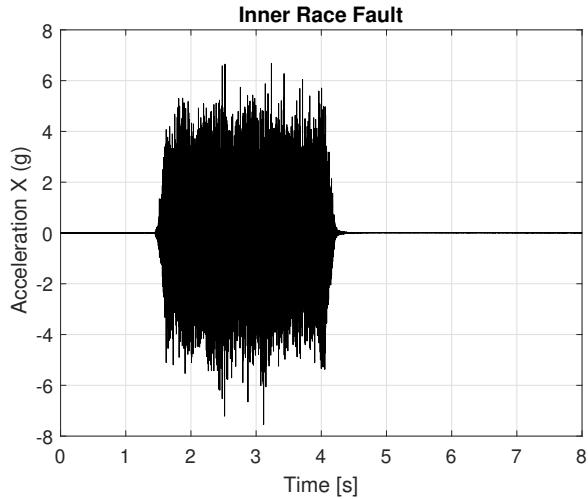


Figure 26: Dati accelerometrici su asse X

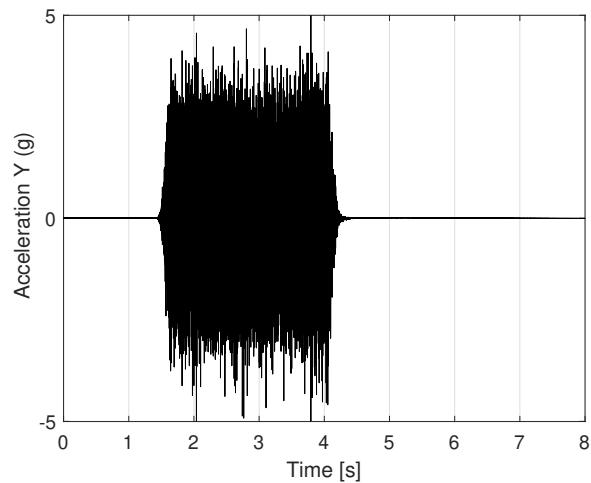


Figure 27: Dati accelerometrici su asse Y

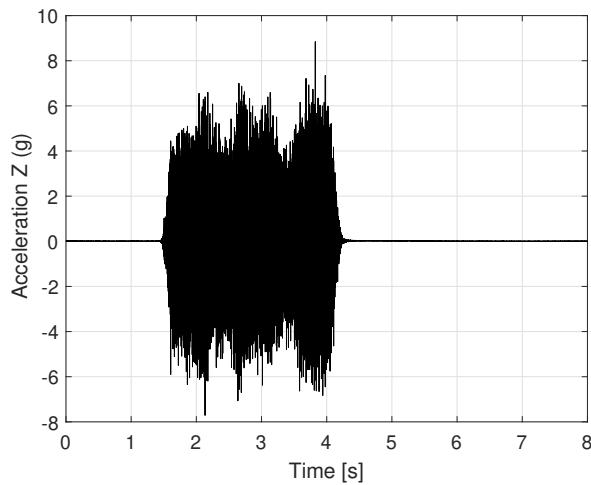


Figure 28: Dati accelerometrici su asse Z

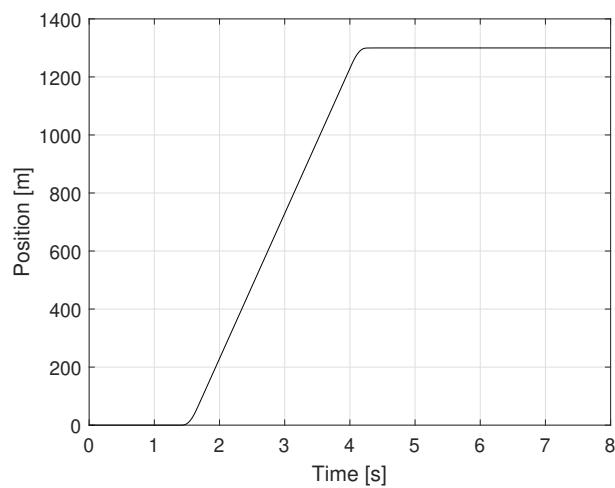


Figure 29: Dati accelerometrici sulla posizione

I dati grezzi originali appena mostrati sono il frutto dell'acquisizione di 8 secondi del segnale accelerometrico ad una frequenza di 12800 Hz sui vari assi ma solo quello sull'asse Y è di nostro

interesse, in quanto assumiamo più esplicativo per quanto concerne il guasto e la sua corretta identificazione. Lo stesso procedimento può essere utilizzato anche sugli altri assi e può portare agli stessi risultati. Sapendo già che è presente un guasto e che ci aspettiamo di trovarlo grazie ai picchi nelle frequenze multiple di 192, data la geometria del cuscinetto e condizioni operative (ovvero: 192Hz, 384Hz, 576Hz, 768Hz, 960Hz, ecc..). Ai fini diagnostici, con dati reali, trovare già le prime due appena elencate è più che sufficiente, considerando che probabilmente già dalla terza frequenza il segnale di guasto (quello che si ripete in linea teorica all'infinito, come le simulazioni teoriche su calcolatori mostrano) può essere mascherata nel rumore di fondo che caratterizza gli spettri derivati da segnali reali. Come si può notare dall'ultimo grafico nella figura 4.1 si possono distinguere 3 fasi: una prima fase iniziale in cui la velocità di rotazione cresce, una seconda in cui è costante e una terza in cui decresce. Di nostro interesse sarà la fase centrale a velocità costante, in quanto unica che permette il corretto funzionamento dei nostri algoritmi di diagnostica (sia classici, sia quello presentato in questo elaborato) senza dover fare un'analisi degli ordini. Si deciderà quindi di utilizzare i dati nella finestra temporale che va dal secondo 2.6 al secondo 4(1.4 secondi campionati a 12800Hz). Il vettore generato e che sarà preso in esame per i prossimi step dell'algoritmo diagnostico avrà quindi 17921 componenti. Purtroppo l'applicazione della FFT diretta a questo vettore non evidenzia le frequenze teoriche tipiche del guasto, rendendo quindi necessario l'esecuzione di diversi stadi di preprocessing dei dati.

## 4.2 Filtraggio e calcolo Inviluppo

Una volta acquisiti i dati delle vibrazioni come appena spiegato si può procedere col filtraggio passa banda, in cui è necessario definire una frequenza centrale e una larghezza di banda a cui sottoporre il segnale grezzo. Ci sono vari modi per ottenere dei suggerimenti sul range di frequenze a cui tagliare il segnale accelerometrico sull'asse Y (ad esempio si può utilizzare l'algoritmo di Spectral Kurtosi). Nel nostro caso si è optato per un filtro FIR del primo ordine con  $f_c = 800$  Hz e larghezza banda  $BW = 800$  Hz, ottenendo così un nuovo vettore, che utilizzeremo per il calcolo dell'inviluppo. La scelta di tale  $f_c$  e  $BW$  è il frutto di algoritmi precedentemente eseguiti su questi stessi dati, che quindi prendiamo come parametri anche per i nostri scopi. Filtrati i dati col filtro appena sopra, si può procedere al calcolo dell'inviluppo  $h(t)$ , che ci servirà per il prossimo step dell'algoritmo di diagnostica. Per il calcolo dell'inviluppo si procede con il padding di zeri della rappresentazione in frequenza dei dati grezzi, raddoppiando la banda del segnale. Viene poi eseguita la IFFT del vettore appena ottenuto e di cui verrà calcolato il modulo. Con questi passaggi è poi possibile esibire il guasto riapplicando la FFT e mostrando il vettore generato in un grafico frequenza-modulo.

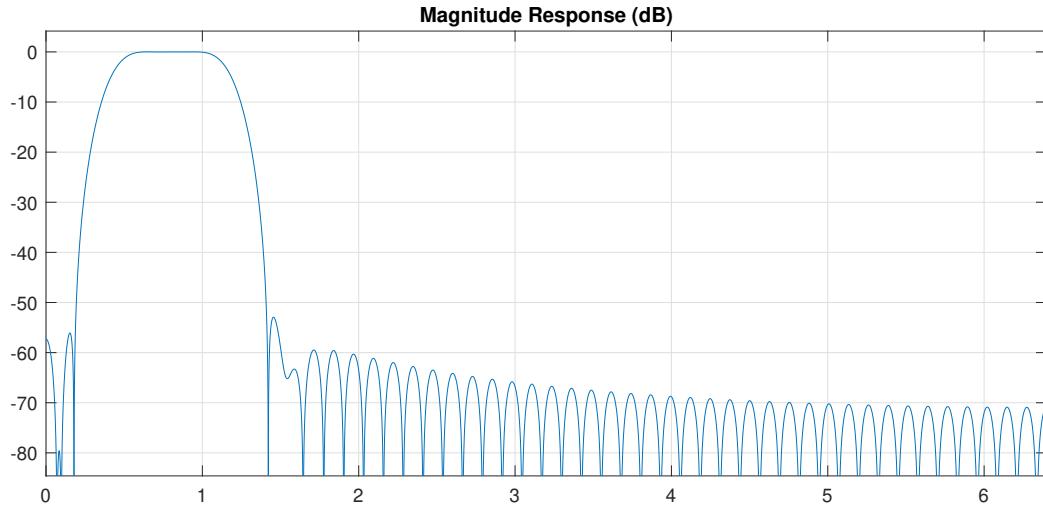


Figure 30:  $f_c=800$   $BW=800$

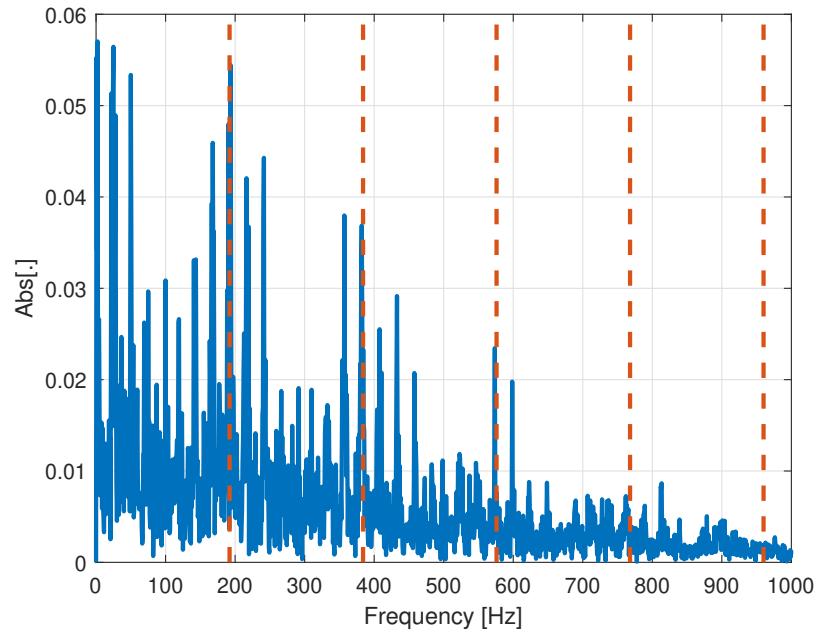


Figure 31: Rappresentazione in frequenza della soluzione trovata dall'algoritmo classico

### 4.3 Applicazione dei metodi di Compressed Sensing

Quello finora presentato come si avrà notato non è altro che una parte dell'algoritmo classico, che infatti viene eseguito fino a questo momento con gli stessi parametri  $f_c$ ,  $BW$  e nello stesso ordine. Alla prossima pagina in figura 32 è possibile trovare l'algoritmo completo, comprensivo di dettagli sulle variabili e la loro natura, che sono stati utilizzati. In questo momento, confrontando l'algoritmo schematizzato in figura 32 e quello finora esposto, mancherebbe solo l'applicazione diretta della FFT all'inviluppo  $\mathbf{b}(t)$  per ottenere  $\mathbf{X}(t)$ . Ricordando che  $\mathbf{X}(t)$  è un vettore di coefficienti reali (l'ultima operazione del calcolo dell'inviluppo prevede il calcolo del modulo di un vettore), quando si va a fare la FFT di  $\mathbf{X}(t)$  si ottiene un vettore immaginario. Scegliendo di plottare il vettore  $\mathbf{X}(t)$  in un grafico dove in ascissa è rappresentata la

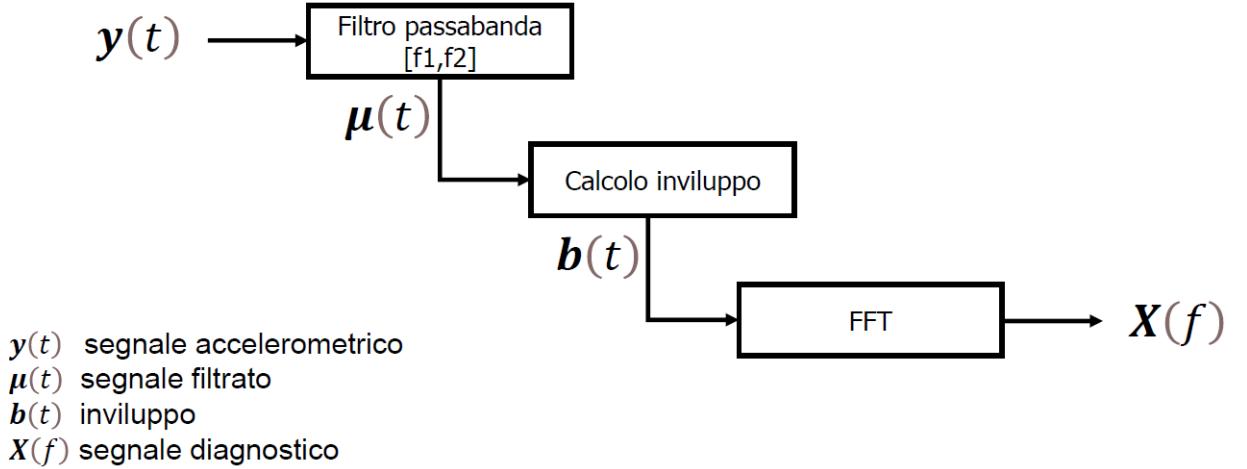
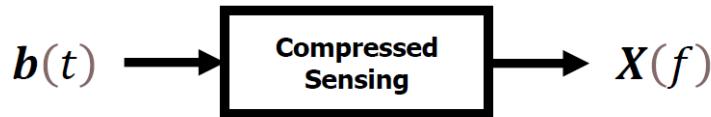


Figure 32: Algoritmo classico schematizzato

frequenza dei coefficienti e in ordinata il loro modulo, per diagnosticare il guasto basterebbe ora notare i picchi nelle frequenze di guasto (che ricordiamo essere 192Hz, 384Hz, 576Hz ecc) o la loro assenza nel caso di cuscinetto sano. Quello che si vorrà andare a fare ora invece, idea di tutto l'elaborato, è sfruttare la sparsità dei picchi dell'inviluppo nel dominio della frequenza e andare ad applicare i metodi di recovery del CS per evidenziare il guasto. In realtà l'inviluppo non è propriamente sparso in frequenza nel senso matematico del termine in quanto il valore assoluto dei coefficienti nelle frequenze non di guasto non è esattamente nullo come si avrà notato (il vettore diagnostico  $X(t)$  ricordiamo essere mostrato in figura 31), ma piuttosto è la parte più significativa della rappresentazione in frequenza dell'inviluppo dei dati filtrati ad essere concentrata intorno alle frequenze tipiche del guasto. Tutto ciò che è invece esterna ad essa è assimilabile ad un rumore che si sovrappone al segnale di interesse.

Riassumendo tutto quindi ci troviamo ora in una situazione di questo genere:



dove utilizzando una concezione più affine ai vettori e ai sistemi lineari e meno ai segnali gli elementi in gioco sono:

- $b$  vettore dell'inviluppo dei dati filtrati;
- $x$  vettore diagnostico che mostra lo stato di salute del componente;

Vorremmo utilizzare le tecniche del CS per calcolare  $x$  a partire dal vettore  $b$ .

#### 4.4 Risolvere il sistema sottodeterminato

Si tratta quindi ora di impostare il problema come segue, utilizzando una sintassi molto simile a quella dei sistemi lineari:

$$\begin{aligned} & \arg \min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1 \\ & \text{subject to } A\boldsymbol{x} = \boldsymbol{b} \end{aligned} \tag{32}$$

La caratteristica che deve presentare il sistema è che sia sottodeterminato con infinite soluzioni, ovvero con meno equazioni che incognite, dove:

- $A$ : è una matrice che rende sparso l'inviluppo (ve ne possono essere diverse, quanto più  $A$  rende sparso l'inviluppo meno campioni rispetto al totale serviranno);
- $\boldsymbol{x}$ : è la rappresentazione sparsa di esso;
- $\boldsymbol{b}$ : è appunto l'inviluppo calcolato dai dati grezzi filtrati;

Come matrice "sparsizzante" possiamo prendere la matrice inversa di Fourier, infatti con qualche rapido passaggio di algebra lineare ci si può convincere che questa permette al vettore  $\boldsymbol{x}$  del sistema lineare appena definito di essere sparso.

Quando nell'algoritmo si esegue la trasformata di Fourier dell'inviluppo, non si fa altro che un'operazione lineare, rappresentabile come il prodotto tra  $Fh$ , tra matrice di Fourier a  $N$  righe e  $N$  colonne e il  $\boldsymbol{x}$  vettore inviluppo dei dati filtrati  $\boldsymbol{h}$   $N \times 1$ , ottenendo così un vettore  $X$  di dimensione  $N \times 1$ . Scriviamo per chiarezza quindi l'equazione:

$$F\boldsymbol{b} = \boldsymbol{x}$$

Come abbiamo spiegato nella precedente sezione,  $\boldsymbol{H}$  è considerabile ai nostri fini come sparso.

Quindi moltiplicando per l'inversa della matrice di Fourier a sinistra entrambi i membri dell'equazione si ottiene:

$$FF^{-1}\boldsymbol{b} = F^{-1}\boldsymbol{x} \tag{33}$$

da cui quindi:

$$\boldsymbol{b} = F^{-1}\boldsymbol{x} \tag{34}$$

Come si può vedere la forma è quella di un sistema lineare  $A\boldsymbol{x} = \boldsymbol{b}$  a patto di chiamare  $A = F^{-1}$ , dove  $\boldsymbol{x}$  è una soluzione sparsa come appunto già discusso.

Affinchè i metodi funzionino non resta quindi che scegliere un'opportuna matrice  $\psi$  che innanzitutto selezioni delle righe casuali del sistema lineare e che ci permetta la risoluzione del sistema sottodeterminato (i.e. che rispetti la RIP property) che ci siamo prefissi di risolvere. Una possibile scelta per  $\psi$  può essere la matrice di proiezione random  $N \times N$  che essenzialmente estrapola da una matrice  $N \times N$  un numero  $n$  inferiore ad  $N$  di righe, mentre applicata ad un vettore estrae  $n$  componenti delle  $N$  possibili.

Un esempio di matrice di proiezione random  $3 \times 5$  è quella che segue:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

che essenzialmente se moltiplichiamo per un vettore  $5 \times 1$ , estrae primo, quarto e quinto elemento, mentre se invece moltiplichiamo per una matrice  $5 \times 5$  estrae similmente prima, quarta e quinta riga. Nel nostro caso la matrice di proiezione scelta sarà una matrice con molte meno righe rispetto alle colonne con la caratteristica di avere uno e uno solo 1 in ogni riga e ognuno di questi non è presente nella stessa colonna di righe diverse (è un sample randomico). È possibile anche utilizzare altri tipi di matrici ma è preferibile questa scelta a livello implementativo, come si vedrà.

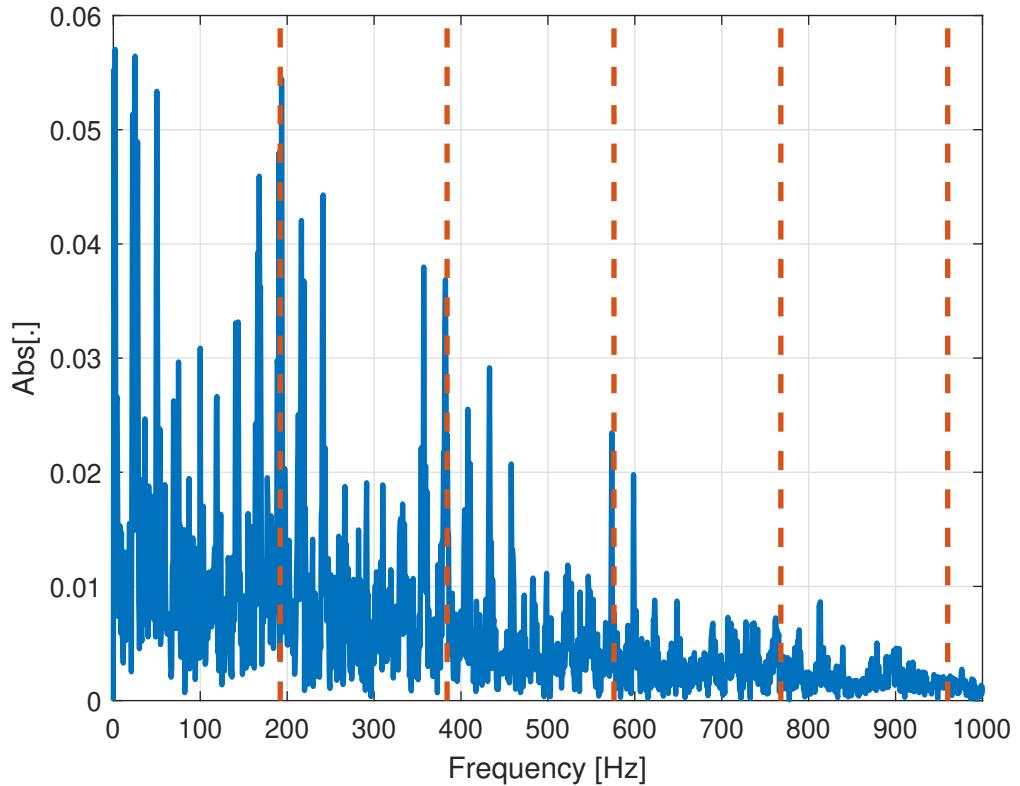
Ora che abbiamo tutti i parametri non ci resta che decidere l'entità di  $n$ , decidiamo quindi di prendere  $n$  uguale a  $N/3$ , quindi circa 3000 campioni dei 9000 che compongono il vettore inviluppo. A questa scelta si è giunti utilizzando in realtà un approccio in prima battuta con un numero elevato di  $n$  che poi si è andato a diminuire progressivamente ed eseguito più volte per valutare la robustezza dei risultati al diminuire di  $n$ .

Istanziare la matrice  $A$  può essere impegnativo per quanto riguarda sia il tempo di calcolo sia lo spazio che la matrice occupa (1.4s campionati a 12800Hz, utilizzati per l'algoritmo classico, portano ad avere in totale quasi 18.000 dati, il che vorrebbe dire calcolare 18.000 esponenziali complessi a 18.000 istanti diversi, ovvero 324.000.000 coefficienti, cosa impegnativa a livello sia computazionale sia dal punto di vista della memoria). Per affrontare questo problema si può ridurre il tempo di acquisizione dati a un intervallo contenuto in quello iniziale (opteremo per una riduzione della finestra temporale, andandola a dimezzare e quindi prendere in considerazione 0.7s dal secondo 2.95 a quello 3.65) e al contempo generare solo le righe della matrice inversa di Fourier che serviranno al calcolo invece di calcolare tutta la matrice. La prima miglioria permette una riduzione di un fattore 4 degli elementi di  $A$  (in quanto la matrice è quadrata), mentre la seconda permette di generare solo le righe che servono per il calcolo effettivo senza doverle generare tutte, che nel nostro caso ricordiamo essere molte meno rispetto a quelle totali della matrice  $A$ . Questa strategia rende possibile la risoluzione di un sistema  $A\mathbf{x} = \mathbf{b}$  sottodeterminato per un calcolatore in quanto la matrice  $\psi$  non fa altro che selezionare alcune righe del sistema completo, altre scelte per  $\psi$  (Bernoulli, Gaussiana, ecc...) porterebbero a dover calcolare ogni riga tramite tutti i coefficienti e sono eseguite a livello hardware e non

software (i sensori misurano quelli che vengono chiamati *linear measurements* in quanto frutto di un'operazione lineare). Precisiamo ancora che una matrice che rende sparso l'inviluppo permette di utilizzare un  $n$  minore ma se questo è troppo piccolo si rischia che la soluzione non converga a quella reale. D'altro lato un numero  $n$  maggiore permette una soluzione più precisa ma ad un costo computazionale maggiore.

Ora che abbiamo tutti i parametri si può utilizzare il solver con i parametri opportuni e vedere la soluzione trovata su un grafico, in cui rappresentiamo il valore assoluto dei coefficienti trovati, rappresentato in ascissa le frequenze.

Figure 33: Rappresentazione in frequenza per la soluzione al sistema  $Ax = b$



Come si può notare con la linea tratteggiata nelle frequenze chiave per la diagnostica del guasto (che ricordiamo essere rispettivamente 192Hz, 384Hz e 476Hz), il cuscinetto è guasto. Purtroppo però, pur utilizzando  $n$  piccoli e le altre accortezze usate, l'esecuzione sarà comunque lenta, richiedendo parecchi minuti prima di terminare contro la decina di secondi dei metodi standard. Ciò è dovuto dal fatto che eseguire la trasformata di Fourier per vettori è risaputamente un'operazione loglineare particolarmente efficiente mentre gli algoritmi di ottimizzazione non è detto che riescano a competere con questa velocità soprattutto per sistemi lineari di comunque grande dimensione.

Nota: l'unica metrica con cui si possano confrontare l'algoritmo classico e gli algoritmi presentati in questo elaborato è il tempo impiegato, in quanto nell'unico passaggio differente si stanno effettuando due cose differenti: nell'algoritmo classico infatti si sta utilizzando la DFT di un vettore, che ha complessità loglineare rispetto alla dimensione dello stesso, nel Compressed

Sensing invece si eseguono algoritmi iterativi di ottimizzazione in cui bisognerebbe confrontare il tasso di convergenza dell'errore ad ogni iterazione(ovvero quanto si riduce l'errore, o un suo stimatore. ad ogni iterazione).

## 4.5 Basis Pursuit Denoising

Si può quindi modificare questo approccio per renderlo trattabile utilizzando delle forme di regolarizzazione. Queste non sono altro che modi per ottenere una soluzione che meno inseguiva l'errore(che caratterizza i coefficienti nelle frequenze che non sono di guasto), perché per quanto concerne a noi non interessa ottenere la FFT esatta dell'inviluppo, sarebbe molto più importante individuare solo i picchi periodici caratteristici del guasto e che idealmente quello in mezzo vorremmo fosse nullo, soprattutto se questo richiedesse meno tempo rispetto a prima. Un primo approccio potrebbe essere formulare il problema diversamente, invece che minimizzare la sparsità soddisfacendo l'equazione si potrebbe ridefinire il problema di ricerca permettendo di promuovere soluzioni più sparse piuttosto che adatte al fit lineare (a patto che riconoscano il guasto o la sua assenza). Si può quindi per esempio andare a modificare leggermente la funzione da minimizzare, permettendo appunto al fit un certo errore, a patto di minimizzare la sparsità della soluzione trovata. Matematicamente si può quindi scrivere:

$$\begin{aligned} & \underset{\mathbf{x}}{\text{minimize}} \quad \frac{1}{2} \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \alpha \|\mathbf{x}\|_1 \\ & \text{subject to} \quad \mathbf{Ax} = \mathbf{b} \end{aligned} \tag{35}$$

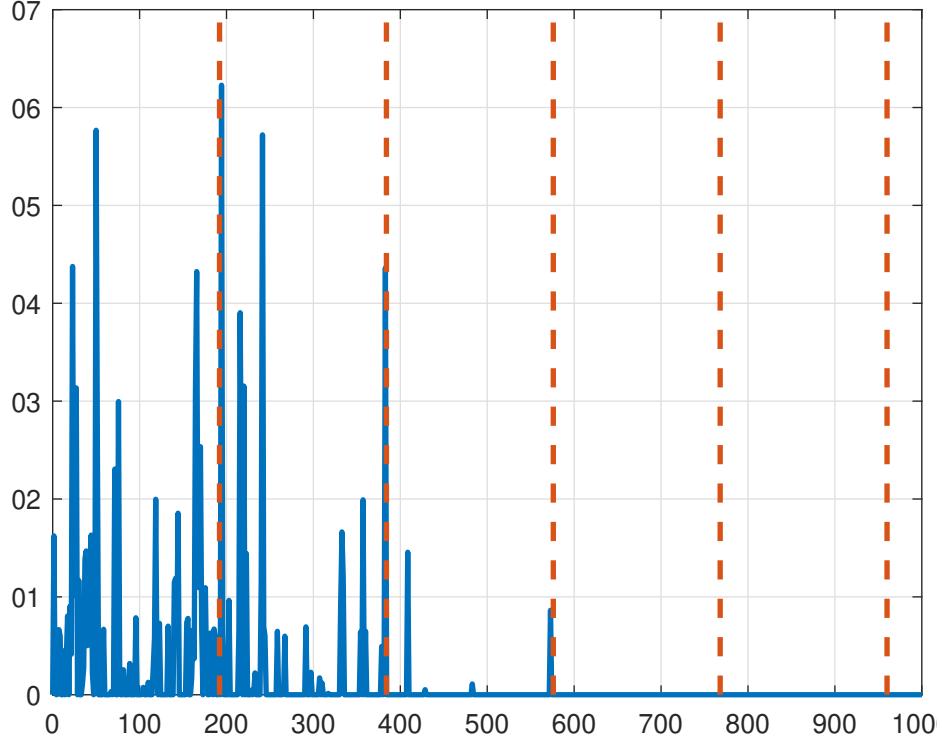
Questo tipo di problema in letteratura viene chiamato Basis Pursuit Denoising. Storicamente è noto anche con un'altra forma simile che utilizza il parametro  $\alpha$  in maniera diversa ma analoga:

$$\begin{aligned} & \underset{\mathbf{x}}{\text{minimize}} \quad \|\mathbf{x}\|_1 \\ & \text{subject to} \quad \|\mathbf{Ax} - \mathbf{b}\|_2^2 \leq \sigma \end{aligned} \tag{36}$$

Dove si minimizza la sparsità della soluzione ma regolarizzando tramite  $\alpha$  che è una sorta di tolleranza che permettiamo alla soluzione di avere (permessi) alla soluzione di avere un errore quadratico medio al più pari a  $\alpha$ . Il solver SPGL1 prevede un comando per la risoluzione di questo tipo di problema che produce la soluzione seguente per  $\sigma$  pari a 11.

Anche in questo caso si diagnostica correttamente, vedendo come l'azione regolarizzatrice del parametro permetta anche di avere un risultato con meno rumore rispetto all'algoritmo classico.

Figure 34: Soluzione ottenuta per problema BPDN con  $\sigma = 11$



## 4.6 LASSO

Un'altra possibile modifica della formulazione che rende il problema trattabile è per esempio imporre alla soluzione da cercare di avere una norma pari ad un parametro  $\tau$ , andando a minimizzare l'errore quadratico medio di essa, in pratica di tutte le possibili soluzioni che hanno norma pari a  $\tau$  prendere quella che si adatta meglio ai dati. Matematicamente si ha quindi:

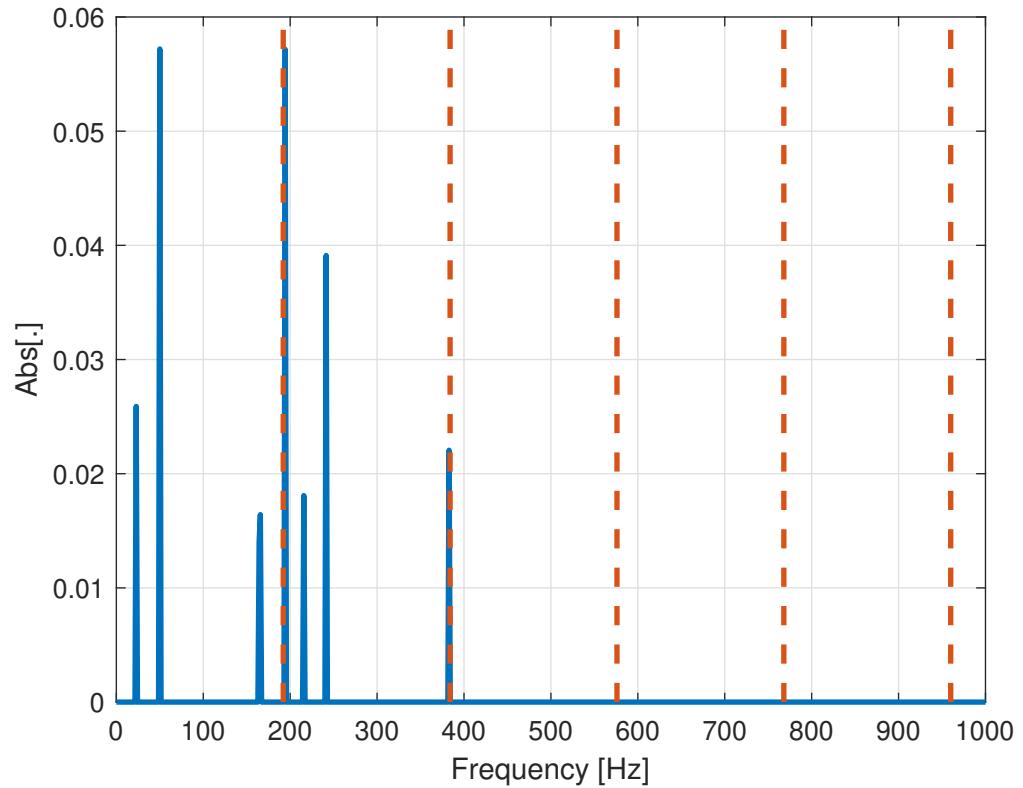
$$\begin{aligned} & \underset{\boldsymbol{x}}{\text{minimize}} \quad \frac{1}{2} \|\boldsymbol{Ax} - \boldsymbol{b}\|_2^2 \\ & \text{subject to} \quad \|\boldsymbol{x}\|_1 \leq \tau \end{aligned} \tag{37}$$

Questo problema in letteratura viene chiamato LASSO. Anche qui SPGL1 ha un comando per la risoluzione di un problema formulato così che lanciato con  $\tau = 4$  produce il seguente risultato: Il guasto viene diagnosticato correttamente, anche più velocemente rispetto al BPDN e come si può notare la soluzione è parecchio più pulita rispetto a prima.

## 4.7 Un'alternativa: il Downsampling

Un altro possibile modo di affrontare il problema è effettuare un downsampling dei dati, piuttosto che limitare la finestra temporale. In altre parole si tratta di utilizzare meno dati di quelli disponibili riducendo di un fattore maggiore di 1 gli stessi. Si potrebbe quindi usare ad esempio un dato ogni 2, oppure ogni 3 e così via. Optando per questa strategia sarebbe necessario verificare due cose. La prima è che la banda del filtraggio passa banda sarebbe da cambiare, la seconda è che i picchi potrebbero cambiare frequenza. Per ovviare questi problemi si è optato per una strategia di controllo con varie combinazioni di frequenza centrale e larghezza di banda

Figure 35: Soluzione ottenuta per problema LASSO con  $\tau = 4$



eseguite con l'algoritmo classico, alla ricerca di come si distribuiscono i picchi in frequenza, i cui risultati, nel caso di downsample di ordine 2, di questa ricerca sono mostrati in figura.

Figure 36:  $f_c = 400\text{Hz}$

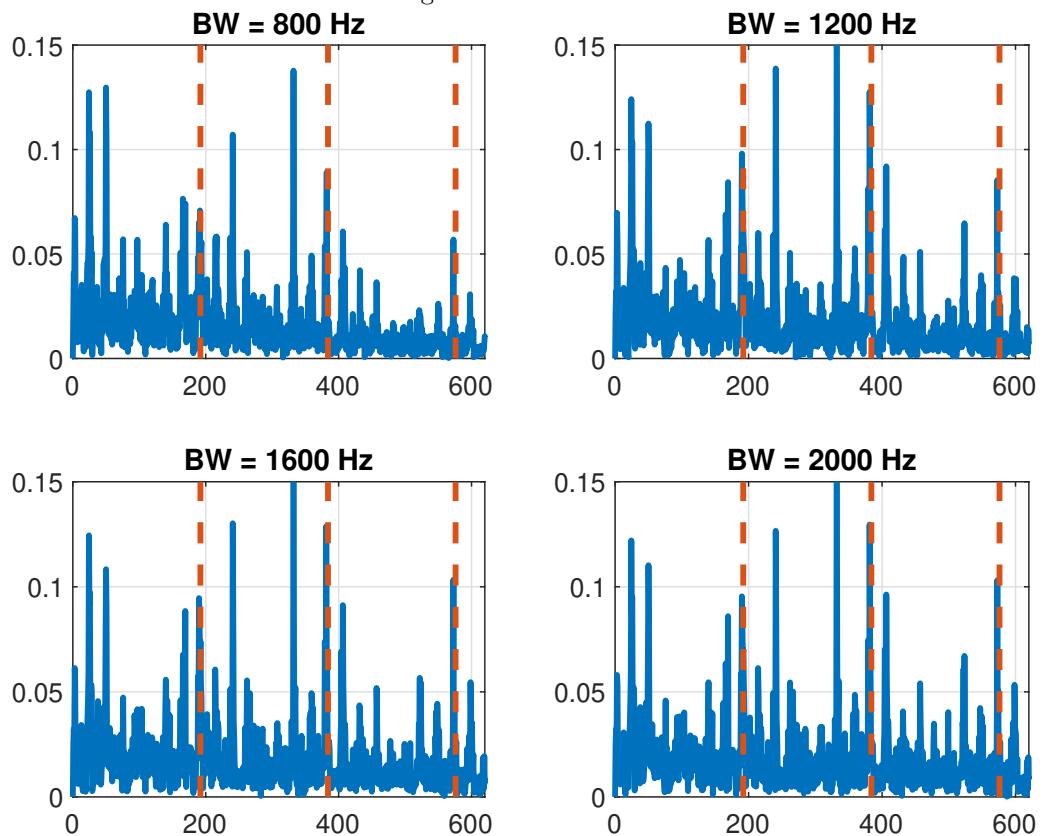


Figure 37:  $f_c = 650\text{Hz}$

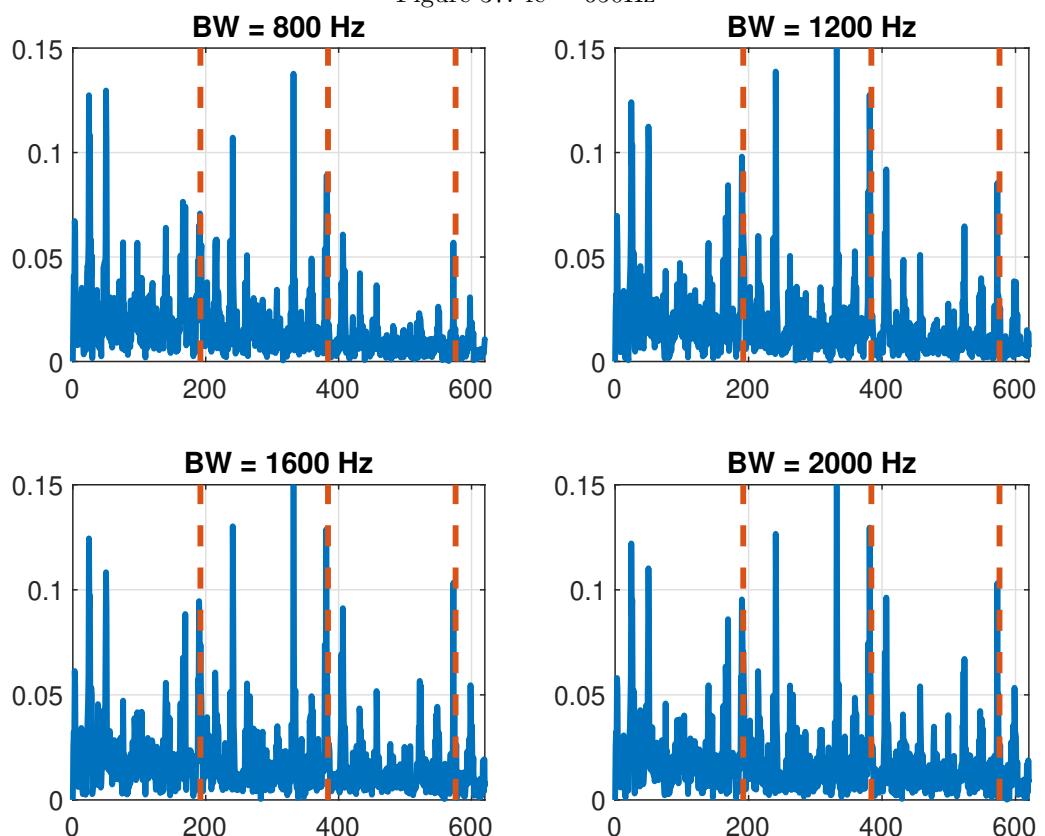


Figure 38:  $f_c = 900\text{Hz}$

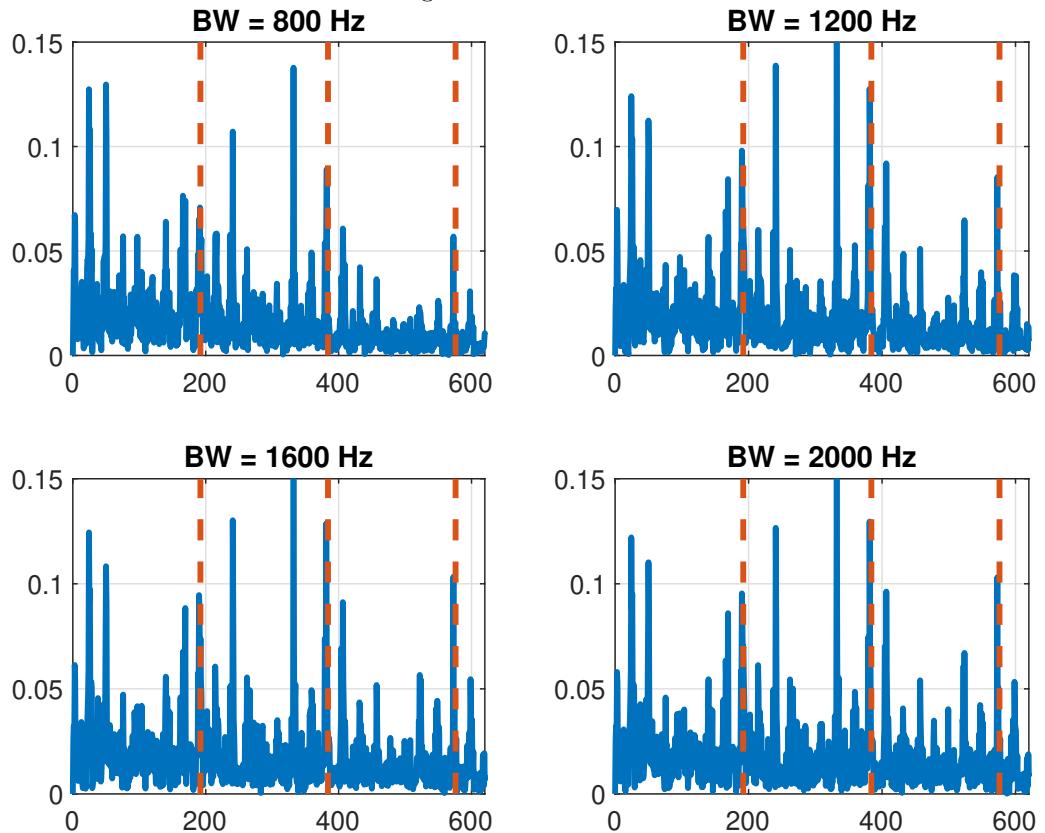
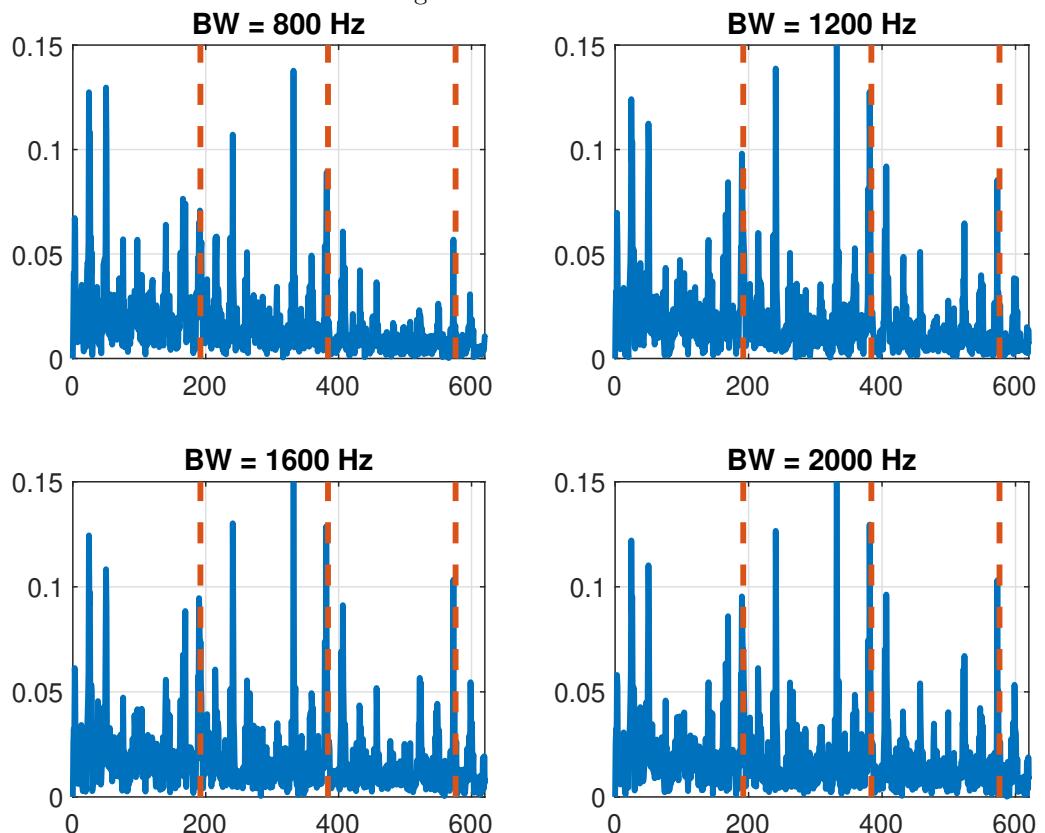


Figure 39:  $f_c = 1150\text{Hz}$



Quelli appena mostrati sono solo alcune delle varie coppie di combinazioni di frequenza centrale e larghezza di banda realmente provate, queste sono solo a scopo illustrativo per non rendere troppo pesante questa parte. Dall'osservazione di grafici simili a questo, ci si è quindi convinti che il downsampling non cambia la posizione dei picchi e che in generale la banda a cui bisogna tagliare differisce rispetto al metodo precedentemente illustrato: si è optato per una frequenza centrale  $fc = 650$  e una larghezza di banda  $BW = 1100$ . Una volta scelta la  $fc$  e la  $BW$  che esibiscono meglio il guasto, scelta che precisiamo non essere univoca e che si può ottimizzare ai fini della ricerca del guasto anche a posteriori per evidenziare più picchi, il procedimento non cambia, conducendo alle stesse conclusioni del procedimento eseguito precedentemente nell'elaborato. Con gli stessi passaggi già eseguiti prima si arriva a questi risultati, che sono gli stessi della trattazione precedente.

Figure 40: Risultato per  $Ax = b$  con DS di ordine 2

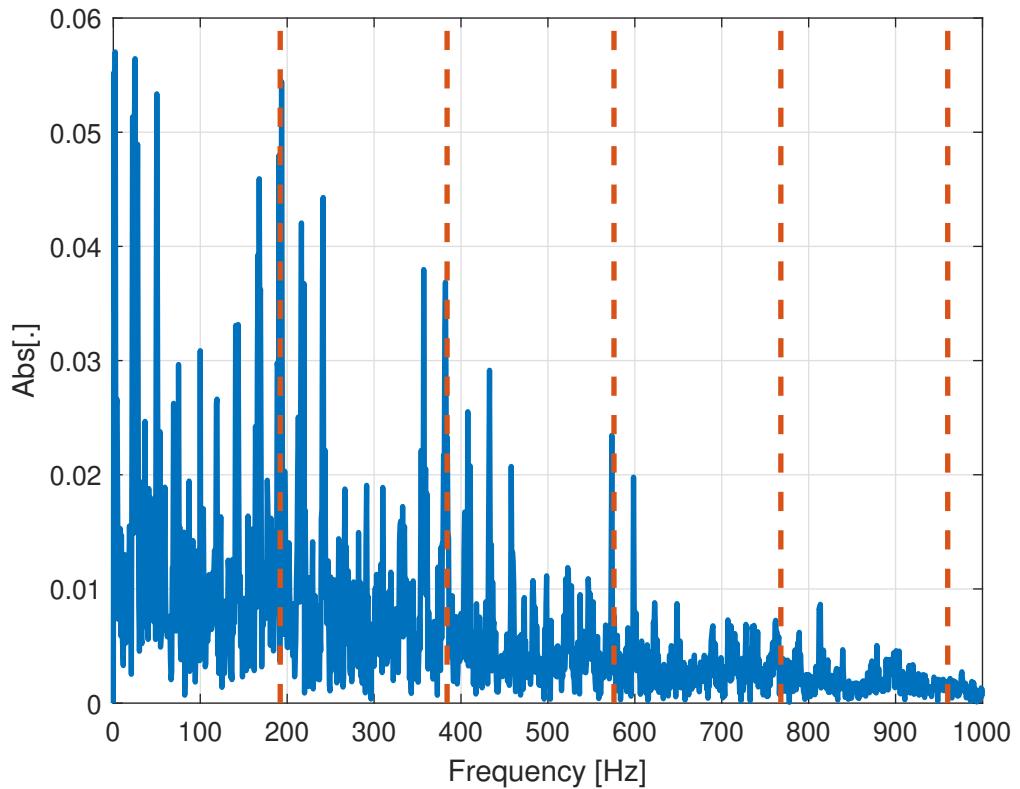


Figure 41: Risultato per problema BPDN con  $\mu = 10$  con DS di ordine 2

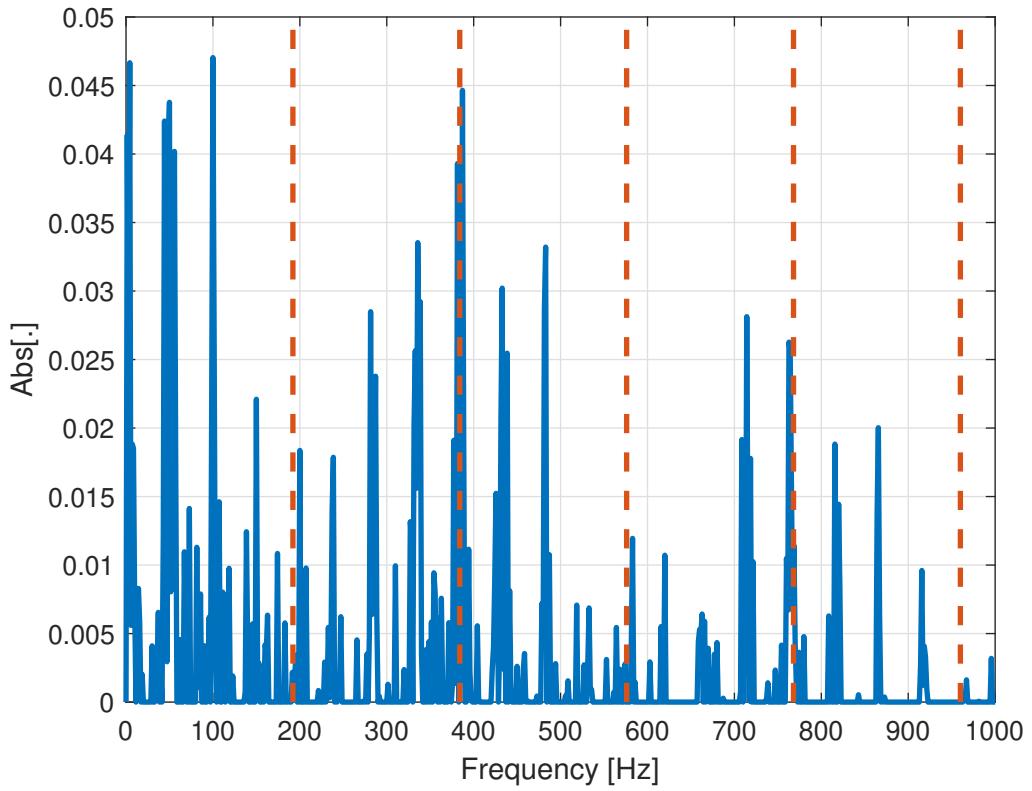
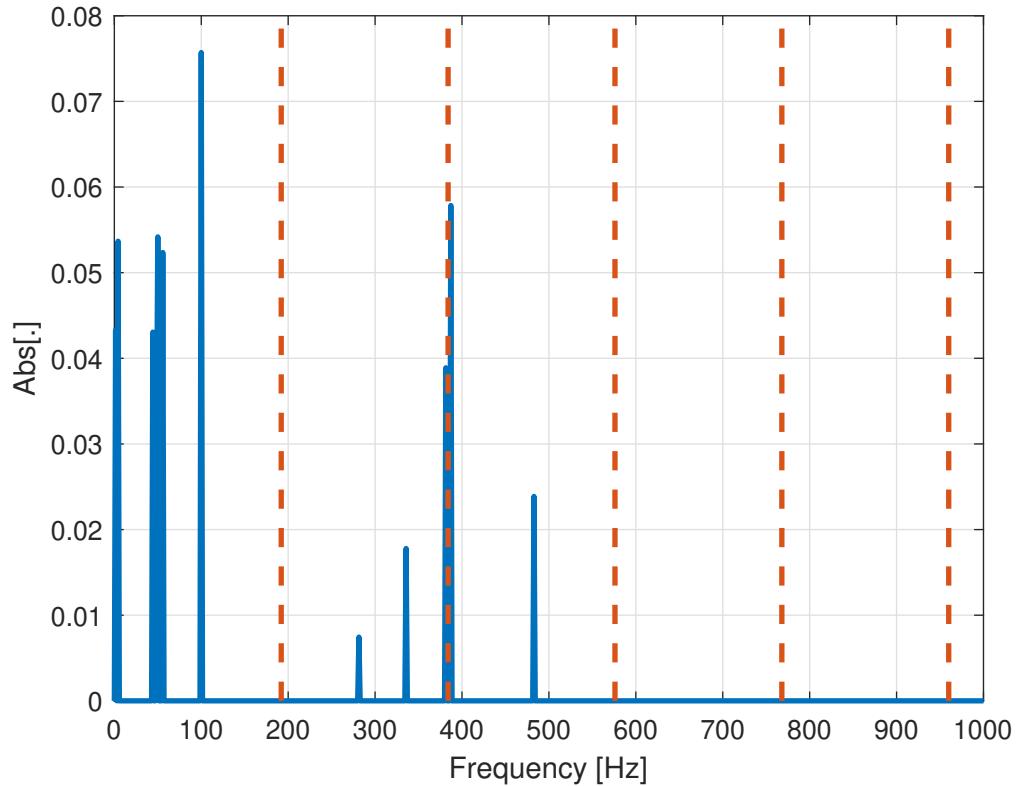


Figure 42: Risultato per Lasso con  $\tau = 5$  con DS di ordine 2



Come si può vedere anche utilizzando il DS è possibile diagnosticare correttamente il guasto. Questo approccio è già preventivabile come più efficiente rispetto a prima, basti pensare anche

solo alla mole di dati che viene ridotta di un fattore K. Ovviamente in caso di K troppo elevato è possibile fallire, soffrendo di problematiche di aliasing in frequenza.



## 5 Conclusioni

In questa sezione verrà condotta un analisi dei risultati, confrontando il metodo classico e analizzando anche il ruolo che i parametri giocano. La sezione si chiude poi con degli spunti per lo sviluppo futuro.

### 5.1 Analisi risultati

I metodi presentati funzionano e, escludendo quelli che prevedono la risoluzione esatta dell'equazione  $\mathbf{Ax} = \mathbf{b}$  sottodeterminata, sono anche veloci e confrontabili col metodo classico presentato in letteratura. Probabilmente per un confronto migliore bisognerebbe utilizzare un maggior numero di dati ma non è assolutamente necessario perché come si è potuto notare con le finestre temporali prese in esame si è potuto individuare con esattezza il guasto, nostro scopo principale.

### 5.2 Confronto con l'algoritmo classico

Come appena esposto nella precedente sezione e mostrato nell'elaborato con opportuni grafici, il metodo proposto funziona. È quindi necessario un confronto col metodo classico in termini di prestazioni. L'esecuzione dell'algoritmo è computazionalmente simile a quella dell'algoritmo classico, il vantaggio principale è l'utilizzo di meno dati, visto che l'algoritmo funziona con  $\frac{1}{5}$  dei dati disponibili o anche meno. Il metodo classico ha come grande vantaggio l'utilizzo di computazioni molto efficienti(FFT in primis) e quindi l'impiego di meno dati non è così tangibile come succede in altri ambiti dove trasformazioni lineari di grandi moli di dati sono computazionalmente onerose e che con un approccio di tipo CS viene subito apprezzata l'efficacia computazionale.

### 5.3 Ruolo Parametri

Durante la trattazione si è volutamente omessa la strategia per la scelta dei parametri  $\sigma$  per il BPDN e  $\tau$  per la LASSO che servono all'esecuzione del solver. Nel nostro caso, ai meri fini puramente diagnostici, si è infatti optato per una strategia di esecuzione molto semplice in cui si eseguiva il solver più volte con dei parametri diversi equispaziati tra loro, graficando i risultati nel dominio delle frequenze per trovarne subito a primo impatto uno più rappresentativo di altri e per vedere come la soluzione cambiasse a seconda dei vari parametri scelti. Come si può vedere nella figura 5.3 in alto non vi è altro che il risultato dell'algoritmo classico per poter effettuare un confronto diretto con esso. Si può notare che all'aumentare di  $\sigma$  la soluzione tenda sempre di più a perdere quel rumore che caratterizza le frequenze non di guasto, finché non diventa addirittura identicamente nulla. Ciò non deve sorprendere perché ricordando che il BPDN non è altro che un trade-off tra fit lineare e sparsità, dove per  $\sigma$  crescenti in pratica permette alla soluzione di discostarsi più dal modello a patto che sia sparsa, si capisce come per  $\sigma$  bassi la soluzione tenda a quella classica mentre per  $\sigma$  alti si tenda a soluzioni nulle, che sono sì molto sparse ma anche altrettanto poco significative per i nostri fini. Per la risoluzione

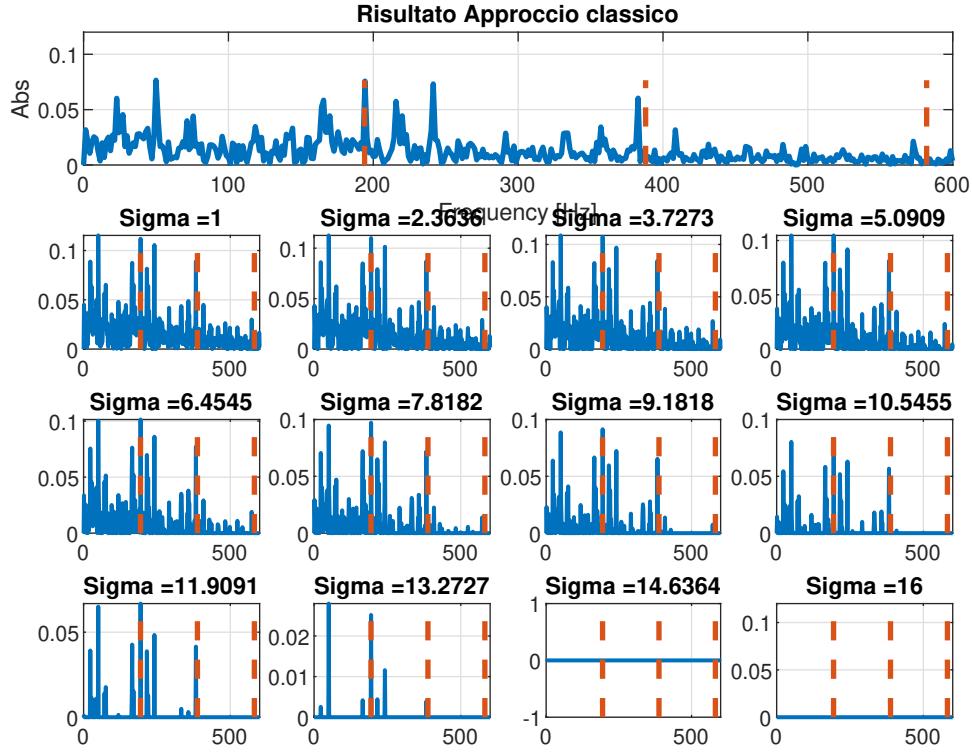


Figure 43: Risultato BPDN al variare di  $\sigma$ .

del problema LASSO si è optato per una strategia identica, ovvero 12 esecuzioni differenti per i parametri  $\tau$  come appunto si può vedere nella prossima figura.

Anche nella figura 5.3, in alto vi è il risultato dell'algoritmo classico per un confronto diretto. Nelle altre 12 immagini si può osservare il risultato delle altrettante esecuzioni con  $\tau$  diversi. Si può notare che al crescere di  $\tau$  la soluzione diventa sempre più complessa. Anche qui bisogna ricordarsi che  $\tau$  non è altro che il parametro che regola la complessità del modello. Analoghi ragionamenti sono stati presi pure nel caso in cui si optasse per il Downampling.

## 5.4 Sviluppi futuri

Vi sono parecchi modi ulteriori per affrontare il problema della diagnostica di un cuscinetto guasto con metodi di Compressed Sensing: vi sono ad esempio algoritmi greedy che potrebbero risolvere il problema molto efficientemente o altri algoritmi di ottimizzazione con funzioni obiettivo formulate in modo diverso che potrebbero produrre risultati interessanti. Vi è anche la possibilità di utilizzare approcci misti che prevedono l'utilizzo di entrambe le tipologie di soluzioni offerte, ad esempio è possibile utilizzare algoritmi di raffinamento di soluzioni inizialmente fornite da algoritmi greedy.

Un altro modo di cambiare il metodo proposto può essere quello di utilizzare una nuova matrice  $A$  che renda ancora più sparsa la soluzione del sistema. Ciò permetterebbe un'esecuzione più rapida in quanto si potrebbero usare meno righe di quelle proposte in questo elaborato (ricordando che la robustezza del metodo si basa su una scelta sufficientemente grande per  $n$ ). Una

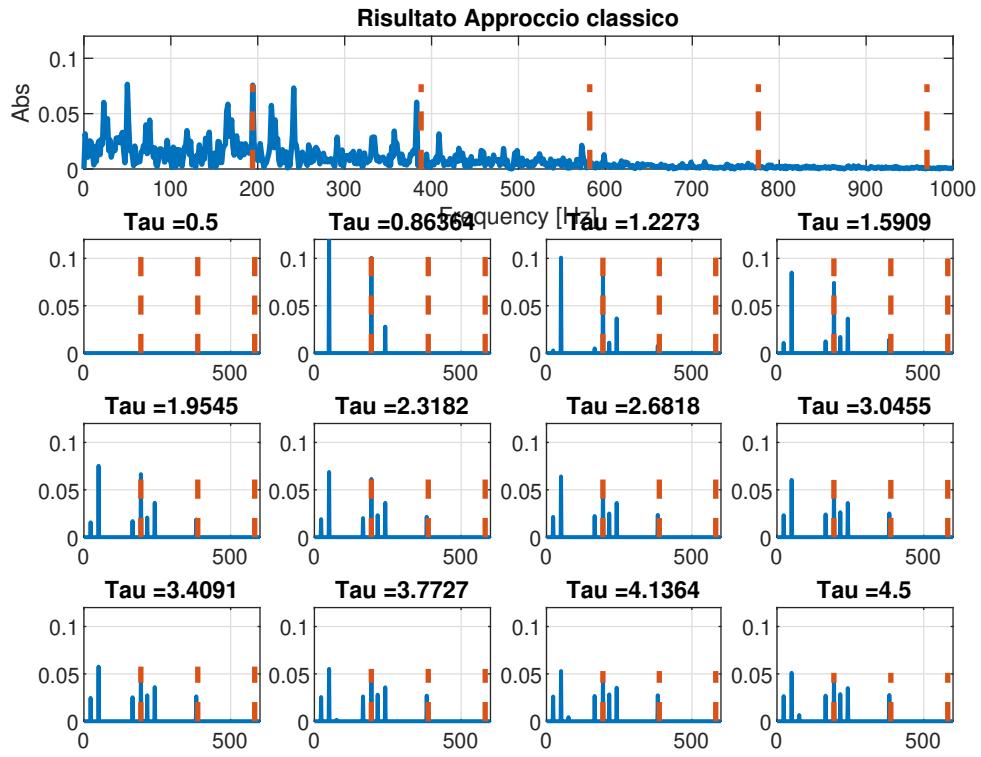


Figure 44: Risultato LASSO al variare di  $\tau$ .

s

possibilità per questa variazione potrebbe essere definire una matrice  $A$  con esponenziali complessi diversi dalla matrice inversa di Fourier (ad esempio utilizzando le frequenze che partano da 150 Hz in poi e vedere come il segnale si distribuisce, creando una FFT ad-hoc per il nostro problema).

Infine per risolvere il problema può essere utilizzare solo ed unicamente il CS, impostando un sistema lineare  $Ax = b$  dove  $A$  è la matrice che tiene traccia di tutte le operazioni eseguite(filtro frequenza, padding zeri, IFFT, modulo e FFT finale) e confrontare in termini di efficienza computazionale i risultati.



## A Codice

In questa sezione verrà esposto brevemente il codice utilizzato per produrre i risultati della precedente sezione.

Una volta acquisito il vettore dei dati completo nel file '`'data.mat'`' si può quindi selezionare la finestra temporale di interesse andando ad operare sul vettore dei tempi che '`'data.mat'`' contiene. Utilizzando `find(x, num)` che cerca la prima occorrenza di `num` nel vettore `x` e ne restituisce l'indice, è possibile infatti ottenere indice di inizio e fine per la finestra temporale scelta.

```
1 ta=2.95;tb=3.65; % istanti di tempo presi in considerazione
2 ta_index = find(t == ta, 1); % indice per ta nel vettore dei tempi completo
3 tb_index = find(t == tb, 1); % indice per tb nel vettore dei tempi completo
```

Tramite questi due indici è quindi possibile selezionare la finestra opportuna per i dati delle vibrazioni (sull'asse Y), mentre per il filtraggio nella banda di interesse è stato utilizzato in primis `designfilt` che crea il filtro e `filter` che utilizzando il valore restituito da `designfilt` filtra il vettore `y` nella frequenza di taglio.

```
1 bpf = designfilt('bandpassfir', 'FilterOrder', 100, 'CutoffFrequency1', 400, ...
2   'CutoffFrequency2', 1200, 'SampleRate', 12800);
3 yBFI = filter(bpf, y);
4 [pEnvBFI_sk, fEnvBFI_sk, ~, ~] = envspectrum(yBFI, fs, ...
5   'FilterOrder', 100, 'Band', [400 1200]);
```

A questo punto per il calcolo dell'inviluppo si può quindi usare appunto `envspectrum` con gli opportuni parametri di frequenza, ordine del filtro e banda. In alternativa è anche possibile utilizzare in serie le operazioni elencate nelle precedenti sezioni per il calcolo dell'inviluppo (DFT, zero padding, IFFT, modulo).

I dati, dopo questi stadi di preprocessing, sono idonei per l'utilizzo delle tecniche di CS, che verranno eseguite dal solver SPGL1. È necessario però prima costruire correttamente la matrice che renda sparso, che come abbiamo visto è la matrice inversa di Fourier, e selezionare un numero minore di colonne rispetto alle righe. Per eseguire quanto appena detto è sufficiente:

```
1 ii = randperm(N); % Generazione vettore 1xN di indici casuali non ripetuti ...
  compresi tra 1 e N
2 ii = ii(1:n); % selezione dei primi n coefficienti
3 A = dftmtx(N)'/N; % costruzione matrice che rende sparso il segnale (inversa ...
  di Fourier)
4 A = A(ii,:); % selezione di solo n righe casuali delle N possibili
5 b = y(ii); % e di conseguenza selezione per il vettore dei termini noti
```

Per una migliore efficienza è possibile anche generare solo i coefficienti utilizzati e non tutta

la matrice come fa il comando `dftmtx`.

Installato il pacchetto correttamente, per eseguire il solver SPGL1 è sufficiente utilizzare i seguenti comandi che sono a disposizione:

- `x_bp = spg_bp(A, b)` risolve il problema di Basis Pursuit  $Ax = b$ ;
- `x_bpdn = spg_bpdn(A, b, alpha)` risolve il problema di Basis Pursuit Denoising  $Ax = b + w$  dove  $w$  è il rumore;
- `x_lasso = spg_lasso(A, b, tau)` risolve il problema di LASSO;

Ricordando che ai nostri fini:

- `x_bp, x_bpdn, x_lasso` sono i vettori che conterranno la soluzione e che sarà il nostro strumento di diagnostica, dove quello che segue l'underscore indica quale metodo è stato utilizzato per generarli;
- `alpha` è il coefficiente di regolarizzazione per il problema BPDN;
- `tau` è il coefficiente di regolarizzazione per il problema LASSO;

Nel caso di Downsample invece è stato utilizzato il comando built-in di MATLAB `y = ... resample(x, fs_new, fs)` che esegue il resample del vettore `x`, ottenuto campionando a frequenza `fs`, in uno nuovo `y` a frequenza `fs_new`. È stato utilizzato direttamente sui dati grezzi e il codice che segue l'esecuzione di questo comando è identico a prima (cambiando opportunamente i valori di finestra temporale, banda di taglio e gli eventuali di regolarizzazione).

Per vedere quindi i risultati è quindi necessario visualizzare la rappresentazione in frequenza dei risultati mediante l'utilizzo del `plot()` di quello che restituisce il comando `fft()`, rappresentando in ascissa la corretta frequenza.



## References

- [1] L. Pitturelli. *Sviluppo e validazione di un metodo di diagnosi e monitoraggio dei guasti per centri di lavoro ad alte prestazioni*. Dec. 2019.
- [2] E. van den Berg and M. P. Friedlander. *SPGL1: A solver for large-scale sparse reconstruction*. <https://friedlander.io/spgl1>. Dec. 2019.
- [3] R. Iserman. *Fault-Diagnosis Systems: an introduction from fault detection to fault tolerance*. Springer Science & Business Media, 2006.
- [4] Z. Gao, C. Cecati, and S. X. Ding. “A Survey of Fault Diagnosis and Fault-Tolerant Techniques—Part I: Fault Diagnosis With Model-Based and Signal-Based Approaches”. In: *IEEE Transactions on Industrial Electronics* 62.6 (2015), pp. 3757–3767. DOI: 10.1109/TIE.2015.2417501.
- [5] Robert B. Randall and Jérôme Antoni. “Rolling element bearing diagnostics—A tutorial”. In: *Mechanical Systems and Signal Processing* 25.2 (2011), pp. 485–520. ISSN: 0888-3270. DOI: <https://doi.org/10.1016/j.ymssp.2010.07.017>. URL: <https://www.sciencedirect.com/science/article/pii/S0888327010002530>.
- [6] K.R. Fyfe and E.D.S. Munck. “ANALYSIS OF COMPUTED ORDER TRACKING”. In: *Mechanical Systems and Signal Processing* 11.2 (1997), pp. 187–205. ISSN: 0888-3270. DOI: <https://doi.org/10.1006/mssp.1996.0056>. URL: <https://www.sciencedirect.com/science/article/pii/S0888327096900564>.
- [7] P. Borghesani et al. “Order tracking for discrete-random separation in variable speed conditions”. In: *Mechanical Systems and Signal Processing* 30 (2012), pp. 1–22. ISSN: 0888-3270. DOI: <https://doi.org/10.1016/j.ymssp.2012.01.015>. URL: <https://www.sciencedirect.com/science/article/pii/S0888327012000167>.
- [8] G. K. Chaturvedi and D. W. Thomas. “Bearing Fault Detection Using Adaptive Noise Cancelling”. In: *Journal of Mechanical Design* 104.2 (Apr. 1982), pp. 280–289. ISSN: 0161-8458. DOI: 10.1115/1.3256337. eprint: [https://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/104/2/280/5809035/280\\\_1.pdf](https://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/104/2/280/5809035/280\_1.pdf). URL: <https://doi.org/10.1115/1.3256337>.
- [9] CC Tan. *An Adaptive Noise Cancellation Approach for Condition Monitoring of Gear Box Bearings*. 1987. URL: <https://search.informit.org/doi/10.3316/informit.435433312546041>.
- [10] D.Ho. “Bearing diagnostics and self-adaptive noise cancellation”. In: (1999).
- [11] J. Antoni and R.B. Randall. “Unsupervised noise cancellation for vibration signals: part II—a novel frequency-domain algorithm”. In: *Mechanical Systems and Signal Processing* 18.1 (2004), pp. 103–117. ISSN: 0888-3270. DOI: [https://doi.org/10.1016/S0888-3270\(03\)00013-X](https://doi.org/10.1016/S0888-3270(03)00013-X). URL: <https://www.sciencedirect.com/science/article/pii/S088832700300013X>.

- [12] Eric Bechhoefer and Michael Kingsley. “A Review of Time Synchronous Average Algorithms”. In: (Jan. 2009).
- [13] R.B. Randall. “Detection and diagnosis of incipient bearing failure in helicopter gearboxes”. In: *Engineering Failure Analysis* 11.2 (2004). Papers presented at the International Conference on Failure Analysis ICFA, pp. 177–190. ISSN: 1350-6307. DOI: <https://doi.org/10.1016/j.engfailanal.2003.05.005>. URL: <https://www.sciencedirect.com/science/article/pii/S1350630703000827>.
- [14] D. HO and R.B. RANDALL. “OPTIMISATION OF BEARING DIAGNOSTIC TECHNIQUES USING SIMULATED AND ACTUAL BEARING FAULT SIGNALS”. In: *Mechanical Systems and Signal Processing* 14.5 (2000), pp. 763–788. ISSN: 0888-3270. DOI: <https://doi.org/10.1006/mssp.2000.1304>. URL: <https://www.sciencedirect.com/science/article/pii/S0888327000913049>.
- [15] Ralph A Wiggins. “Minimum entropy deconvolution”. In: *Geoexploration* 16.1 (1978), pp. 21–35. ISSN: 0016-7142. DOI: [https://doi.org/10.1016/0016-7142\(78\)90005-4](https://doi.org/10.1016/0016-7142(78)90005-4). URL: <https://www.sciencedirect.com/science/article/pii/0016714278900054>.
- [16] Jérôme Antoni. “The spectral kurtosis: a useful tool for characterising non-stationary signals”. In: *Mechanical Systems and Signal Processing* 20.2 (2006), pp. 282–307. ISSN: 0888-3270. DOI: <https://doi.org/10.1016/j.ymssp.2004.09.001>. URL: <https://www.sciencedirect.com/science/article/pii/S0888327004001517>.
- [17] N. Sawalhi, R.B. Randall, and H. Endo. “The enhancement of fault detection and diagnosis in rolling element bearings using minimum entropy deconvolution combined with spectral kurtosis”. In: *Mechanical Systems and Signal Processing* 21.6 (2007), pp. 2616–2633. ISSN: 0888-3270. DOI: <https://doi.org/10.1016/j.ymssp.2006.12.002>. URL: <https://www.sciencedirect.com/science/article/pii/S0888327006002603>.
- [18] D. L. Donoho. “Compressed sensing”. In: *IEEE Transactions on Information Theory* 52.4 (2006), pp. 1289–1306. DOI: [10.1109/TIT.2006.871582](https://doi.org/10.1109/TIT.2006.871582).
- [19] Emmanuel J. Candès, Justin K. Romberg, and Terence Tao. “Stable signal recovery from incomplete and inaccurate measurements”. In: *Communications on Pure and Applied Mathematics* 59.8 (2006), pp. 1207–1223. DOI: <https://doi.org/10.1002/cpa.20124>. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.20124>. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.20124>.
- [20] Steven L. Brunton and J. Nathan Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2019. DOI: [10.1017/9781108380690](https://doi.org/10.1017/9781108380690).
- [21] Abhiram Natarajan and Yi Wu. “Computational Complexity of Certifying Restricted Isometry Property”. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*. Ed. by Klaus Jansen et

- al. Vol. 28. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2014, pp. 371–380. ISBN: 978-3-939897-74-3. DOI: 10.4230/LIPIcs.APPROX-RANDOM.2014.371. URL: <http://drops.dagstuhl.de/opus/volltexte/2014/4709>.
- [22] Simon Foucart and Holger Rauhut. *A Mathematical Introduction to Compressive Sensing*. Birkhäuser Basel, 2013. ISBN: 0817649476.
- [23] David Donoho et al. “Sparse Solution of Underdetermined Systems of Linear Equations by Stagewise Orthogonal Matching Pursuit”. In: *IEEE Transactions on Information Theory* 58 (Feb. 2012), pp. 1094–1121. DOI: 10.1109/TIT.2011.2173241.
- [24] D. Needell and J.A. Tropp. “CoSaMP: Iterative signal recovery from incomplete and inaccurate samples”. In: *Applied and Computational Harmonic Analysis* 26.3 (2009), pp. 301–321. ISSN: 1063-5203. DOI: <https://doi.org/10.1016/j.acha.2008.07.002>. URL: <https://www.sciencedirect.com/science/article/pii/S1063520308000638>.
- [25] Jian Wang, Seokbeop Kwon, and Byonghyo Shim. “Generalized Orthogonal Matching Pursuit”. In: *IEEE Transactions on Signal Processing* 60.12 (2012), pp. 6202–6216. DOI: 10.1109/TSP.2012.2218810.
- [26] Suhyuk Kwon, Jian Wang, and Byonghyo Shim. “Multipath Matching Pursuit”. In: *IEEE Transactions on Information Theory* 60.5 (2014), pp. 2986–3001. DOI: 10.1109/TIT.2014.2310482.
- [27] Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. “Atomic Decomposition by Basis Pursuit”. In: *SIAM J. Sci. Comput.* 20.1 (Dec. 1998), pp. 33–61. ISSN: 1064-8275. DOI: 10.1137/S1064827596304010. URL: <https://doi.org/10.1137/S1064827596304010>.
- [28] Robert Tibshirani. “Regression Shrinkage and Selection via the Lasso”. In: *Journal of the Royal Statistical Society. Series B (Methodological)* 58.1 (1996), pp. 267–288. ISSN: 00359246. URL: <http://www.jstor.org/stable/2346178>.
- [29] P. R. Gill, A. Wang, and A. Molnar. “The In-Crowd Algorithm for Fast Basis Pursuit Denoising”. In: *IEEE Transactions on Signal Processing* 59.10 (2011), pp. 4595–4605. DOI: 10.1109/TSP.2011.2161292.
- [30] Tyler Johnson and Carlos Guestrin. “Blitz: A Principled Meta-Algorithm for Scaling Sparse Optimization”. In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 1171–1179. URL: <http://proceedings.mlr.press/v37/johnson15.html>.
- [31] Emmanuel Candes and Terence Tao. “The Dantzig selector: Statistical estimation when p is much larger than n”. In: *The Annals of Statistics* 35.6 (2007), pp. 2313–2351. DOI: 10.1214/009053606000001523. URL: <https://doi.org/10.1214/009053606000001523>.

- [32] Emmanuel Candes and Justin Romberg.  *$l_1$ -MAGIC: Recovery of Sparse Signals via Convex Programming*. Jan. 2005.
- [33] Michael Grant and Stephen Boyd. *CVX: Matlab Software for Disciplined Convex Programming, version 2.1*. <http://cvxr.com/cvx>. Mar. 2014.
- [34] Yin Zhang. “User’s Guide for YALL1: Your ALgorithms for L1 Optimization”. In: *Tech-nique Report* (Jan. 2009).