ALMA MATER STUDIORUM
UNIVERSITA' DI BOLOGNA

SCUOLA DI INGEGNERIA E ARCHITETTURA
- Sede di Forlì –

CORSO DI LAUREA
IN INGEGNERIA MECCANICA
Classe LM-33

TESI DI LAUREA in
CONTROLLO DEI MOTORI A COMBUSTIONE INTERNA

SVILUPPO DI SISTEMI DI CONDITION MONITORING SU PIATTAFORMA REAL-TIME/FPGA PER TEST IN SALA PROVE

Candidato:
MARCO ABBONDANZA

Relatore:
Prof. Ing. **ENRICO CORTI**

ANNO ACCADEMICO 2016-2017

INDICE

SIGLE E ABBREVIAZIONI

ω	Pulsazione
μ	Media
σ	Deviazione standard
FIFO	First In First Out
Fs	Sampling Frequency
TPS	Throttle Position Sensor
NaN	Not a Number
STFT	Short Time Fourier Trasform
RAM	Random Access Memory
IMEP	Pressione media indicata
RPM	Giri al minuto
RMS	Root Mean Squared, valore quadratico medio
VI	Virtual Instrument (Labview)
WOT	Whole Open Throttle
ECU	Engine Control Unit
CdP	Curva di Potenza

INTRODUZIONE

L'attività di tesi nasce dall'esigenza presente in sala prove motori di monitorare in modo continuativo l'affidabilità del propulsore, con l'obiettivo di ottenere valutazioni predittive rispetto ai guasti che si possono verificare in sede di test stressanti come prove di durata o semplicemente per valutare lo stato di usura del motore durante la sua intera vita al banco. La funzione predittiva è fondamentale per evitare rotture disastrose che possono compromettere anche l'efficacia della diagnosi del problema che ha portato al guasto: questo aspetto è di primaria importanza quando i componenti sottoposti a test sono di origine prototipale o destinate all'impego in gara, situazione tipica all'interno di una sala prove.

In collaborazione con l'azienda Alma Automotive s.r.l. è stato quindi lanciato lo sviluppo di una personalità software dedicata al condition monitoring, basata sulla piattaforma hardware Miracle2 (M2), prodotta dall'azienda e su cui attualmente è implementato il sistema indicating OBI-M2 (On-board Indicating), in grado di calcolare e rendere disponibili i principali indici di combustione ricavabili dai segnali di pressione cilindro, sia in applicazioni on-board che al banco prova. Il cuore della centralina, di dimensioni molto contenute, è costituito da un modulo National Instruments che racchiude un processore real-time ed un FPGA (Field Programmable Gate Array) e consente, attraverso la programmazione in linguaggio Labview, l'implementazione di sistemi per numerose e diverse applicazioni, capaci di operare appunto in tempo reale con il funzionamento del motore. Tra i sistemi di condition monitoring per motori a combustione interna quello prodotto dall'azienda Reilhofer KG ("Delta Analyzer") è stato preso come riferimento, sia perché in uso presso Ducati, sia perché particolarmente diffuso sul mercato tedesco.

Il sistema è basato principalmente sul rilevamento di vibrazioni e lo sviluppo del progetto parte dalla selezione degli accelerometri e dalla validazione della catena di misura. Il confronto prevede test sia su motore che su banco vibrante di sensori di

diversa tipologia, come quelli da laboratorio e quelli destinati ad applicazioni di massa. A valle di un'attività propedeutica per l'apprendimento dei principali strumenti di programmazione in Labview e di presa di confidenza con l'hardware (accelerometri e centralina M2) è stata implementata una struttura di base del software che consentisse l'acquisizione tramite FPGA, lo streaming ed il processamento real-time dei dati ed in parallelo il salvataggio.

È stata realizzata una campagna sperimentale durante la quale monitorare tramite accelerometri l'intera vita del motore, dal rodaggio fino allo smontaggio (eventualmente causato dalla rottura), in modo da poter disporre di dati per l'analisi e la calibrazione offline del sistema. Il metodo di diagnosi è incentrato principalmente sul confronto fra un riferimento dello stato vibrazionale del motore e lo stato attuale al variare del punto di funzionamento basato su dati come ad esempio carico, rpm, temperature e posizione all'interno del ciclo di prova previsto. La valutazione dello stato di salute del motore è effettuata con semplici indici derivati principalmente dallo spettro delle vibrazioni, confrontato con la condizione vibrazionale baseline, registrata a motore nuovo. Il linguaggio utilizzato, unitamente all'hardware Miracle2, fornisce molte potenzialità in tal senso. A partire dai dati rilevati sull'intera vita del motore, l'obiettivo è stato quello di poter identificare indici significativi che consentano il riconoscimento di un trend di degrado e l'impostazione di valori di soglia.

È stata inoltre svolta un'analisi di rilevamento della detonazione tramite accelerometri, effettuando la taratura sugli indici MAPO registrati da OBI. Sempre relativamente alle strategie di condition monitorning applicate alla sala prova, è stata prevista l'effettuazione di valutazioni sullo stato vibrazionale della macchina elettrica, facendo riferimento ai metodi di rilevamento ed alle soglie riportate nella normativa ISO 20816-1. Inoltre, attraverso dati provenienti dagli encoder del motore elettrico e dalla ruota fonica del motore termico, è stata svolta un'analisi rotazionale per determinare strategie volte al monitoraggio del giunto elastico, elemento molto stressato.

Capitolo 1
CONDITION MONITORING

1.1 INTRODUZIONE ED OBIETTIVI DEL CONDITION MONITORING

Il “condition monitoring” di una macchina consiste nella misura di vari parametri relativi alla condizione di funzionamento (come le vibrazioni, la temperatura dei cuscinetti, pressione dell’olio, detriti nell’olio e prestazioni), che rendono possibile determinare se la macchina è in buone o cattive condizioni meccaniche. Nel caso del rilevamento di una condizione ritenuta critica, allora è spesso possibile risalire anche alla causa del problema grazie ad analisi più mirate.

Le tecniche di condition monitoring sono usate unitamente alla manutenzione predittiva (“predictive maintenance”), manutenzione basata sull’indicazione che un guasto stia per verificarsi. La tendenza e l’obiettivo di questo approccio è soppiantare la manutenzione a rottura (“run-to-breakdown maintenance”) e la manutenzione preventiva con i conseguenti vantaggi:

- Evitare guasti catastrofici con i conseguenti elevati costi e rischi.
- Ridurre al minimo gli interventi di revisione sulle macchine, riducendo i costi.
- Eliminare gli interventi non necessari ed il conseguente rischio di introdurre problemi su macchinari funzionanti correttamente.
- Consentire di avere le parti di ricambio ordinate in tempo e ridurre i costi di magazzino.
- Ridurre il tempo di intervento grazie alla conoscenza in anticipo del guasto.
- Consentire una migliore diagnosi sulle cause di un determinato problema, che diversamente sarebbero di difficile individuazione in caso di rottura catastrofica.

I sistemi per il condition monitoring si dividono in due tipi:

Periodic monitoring systems (off-line condition monitoring systems):
I parametri della macchina sono misurati o registrati ad intervalli regolari e possono essere analizzati anche in un secondo momento. Questi sistemi sono adatti a macchinari che lavorano ininterrottamente per lunghi periodi ed in cui il danneggiamento può essere individuato con largo anticipo, come ad esempio su impianti per la generazione di energia, impianti di produzione o macchinari che lavorano sul campo. Le misurazioni che vengono eseguite possono essere in numerose posizioni diverse della stessa macchina e le analisi possono essere complesse e mirate ad individuare i singoli guasti.

Permanent monitoring systems (on-line condition monitoring systems):
Le misurazioni (delle vibrazioni in particolare) sono eseguite continuamente ed i valori attuali sono confrontati continuamente con i livelli di riferimento per valutarne l'accettabilità. Lo scopo principale di sistemi di questo tipo è proteggere la macchina dalla rottura provvedendo a generare un allarme o al tempestivo spegnimento in caso di superamento di valori di soglia prefissati. I sensori sono montati in modo permanente ed il sistema di acquisizione ed analisi deve essere automatizzato per funzionare in modo continuo. Il monitoraggio permanente è richiesto laddove sia presente un'applicazione critica oppure dove il guasto può presentarsi in maniera piuttosto repentina.

Tipicamente lo strumento maggiormente utilizzato e con maggior profitto sono le **vibrazioni strutturali** rilevate sulla macchina (accelerazione, velocità e spostamento), ma come accennato in precedenza, esistono casi specifici che possono essere basati sul monitoraggio di altri parametri. Avere a disposizione una panoramica delle frequenze eccitate dai vari organi della macchina è importante per condurre un'analisi dei guasti che possa individuare i problemi basandosi sulle frequenze che appaiono sullo spettro misurato durante il funzionamento. Di seguito sono riportate alcune caratteristiche che possono essere necessarie per l'analisi di un guasto:

- Velocità di rotazione degli alberi, frequenze corrispondenti a difetti nei cuscinetti, numero di denti nelle ruote dentate, numero delle pale o dei vani nelle pompe, numero dei poli di un motore elettrico.
- Forze generate da disallineamenti e masse non bilanciate.
- Influenza di fattori come temperatura e pressione sui fenomeni vibratori.
- Sensibilità al carico ed alle condizioni operative.

Importante è la scelta del trasduttore e della grandezza da utilizzare: trasduttori di posizione, ad esempio a correnti parassite, per misurare in una macchina rotante lo spostamento relativo fra albero e alloggiamento, oppure dei più leggeri e robusti accelerometri, più adatti alla diagnosi di fenomeni ad alta frequenza come danneggiamento dei cuscinetti o ruote dentate. In linea di massima è possibile utilizzare come trasduttore un accelerometro ed integrarlo elettronicamente all'occorrenza per ottenere velocità o spostamento. Il parametro più appropriato da utilizzare può essere considerato quello che garantisce la misurazione con lo spettro più "piatto". Con riferimento ad una singola armonica, qualunque sia il parametro considerato, rimangono inalterati forma e periodo della vibrazione, mentre si verifica una variazione di ampiezza e di fase. Quando si eseguono misure mediate nel tempo, si trascura la fase e le relazioni tra i tre parametri vengono stabilite unicamente dalla pulsazione $\omega=2\pi f$.

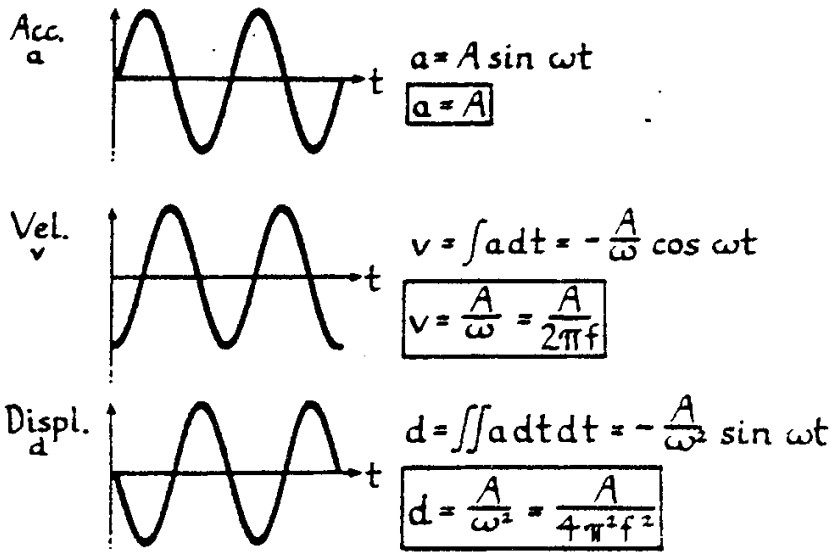


Figura 1: Integrazione da accelerazione a spostamento

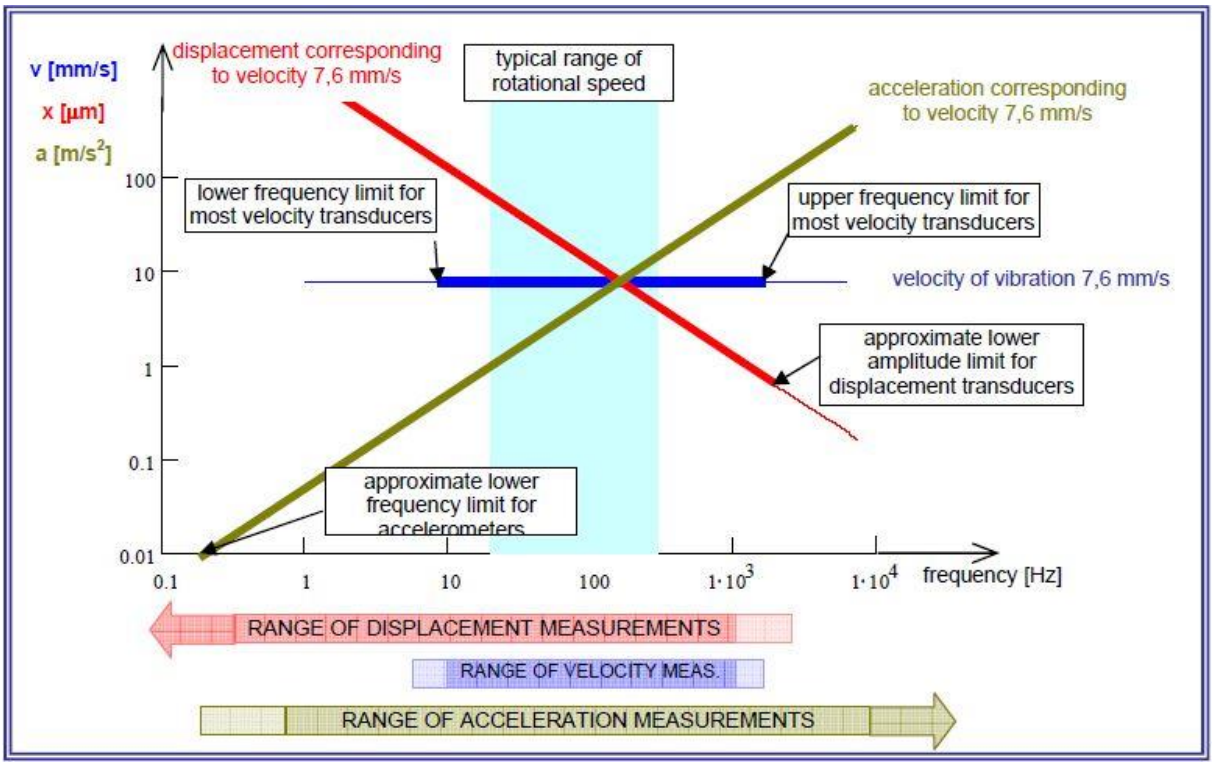


Figura 2: Confronto in frequenza dell'utilizzo di accelerazione, velocità e spostamento. Nell'esempio viene mostrato uno spettro costante di velocità pari a 7,6 mm/s.

In letteratura sono presenti numerosi lavori inerenti all'ambito produzione energia [1] [2], aeronautico [3] e ovunque si renda utile o strettamente necessario intervenire in modo predittivo sulle rotture (motori navali [4] e sistemi di trasmissione [5] ad esempio). Tipicamente si applicano valutazioni basate sull'analisi degli ordini di vibrazione e su indici ad essi correlati in relazione con lo stato vibrazionale del componente nuovo o indicato sulle specifiche normative di riferimento [6], oppure nel caso di turbine a gas è possibile utilizzare come parametri di valutazione del trend di salute pressioni e temperature caratteristiche durante il ciclo del fluido elaborato. Tipico è l'utilizzo di strategie di condition monitoring (analisi del cepstrum e dell'inviluppo del segnale ad esempio [7]) per l'individuazione di difetti legati ai cuscinetti ad esempio di motori elettrici e macchinari industriali. Con particolare riferimento ai motori a combustione interna esistono casi di studio che riguardano soprattutto il riconoscimento di difetti ed errato montaggio [8]. Esiste anche un'ampia letteratura che mostra l'utilizzo dell'analisi chimico-fisica dell'olio lubrificante per determinare lo stato di usura del motore [9], [10].

1.2 TECNICHE DI ANALISI BASATE SU VIBRAZIONI

1.2.1 *Metodi statistici basati sul segnale base tempo*

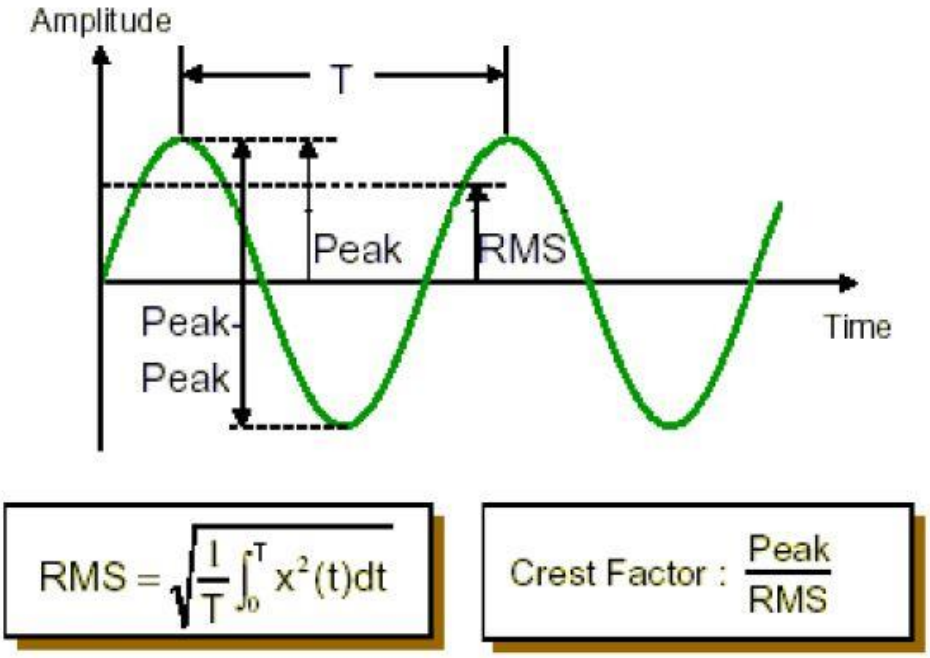


Figura 3:RMS, Peak-Peak, Peak e CF per un segnale sinusoidale

RMS

Il valore RMS di un segnale su un certo intervallo di tempo corrisponde al valore continuo che provvede allo stesso contenuto energetico del segnale in esame:

$$RMS = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x^2(i)}$$

Dal punto di vista diagnostico l'RMS non presenta elevata sensibilità perché media il contributo di eventuali componenti impulsive. È un parametro che può essere calcolato su spostamento, velocità, accelerazione a seconda del campo di frequenze d'interesse.

Peak-Peak

Il valore picco-picco di un segnale riporta l'escursione massima del segnale ed è sensibile ad eventi impulsivi, ma può mascherare l'aumento del livello medio di vibrazione.

Crest Factor

Il crest factor (CF) è definito come rapporto fra il valore di picco ed il valore efficace.

$$CF = \frac{Peak}{RMS}$$

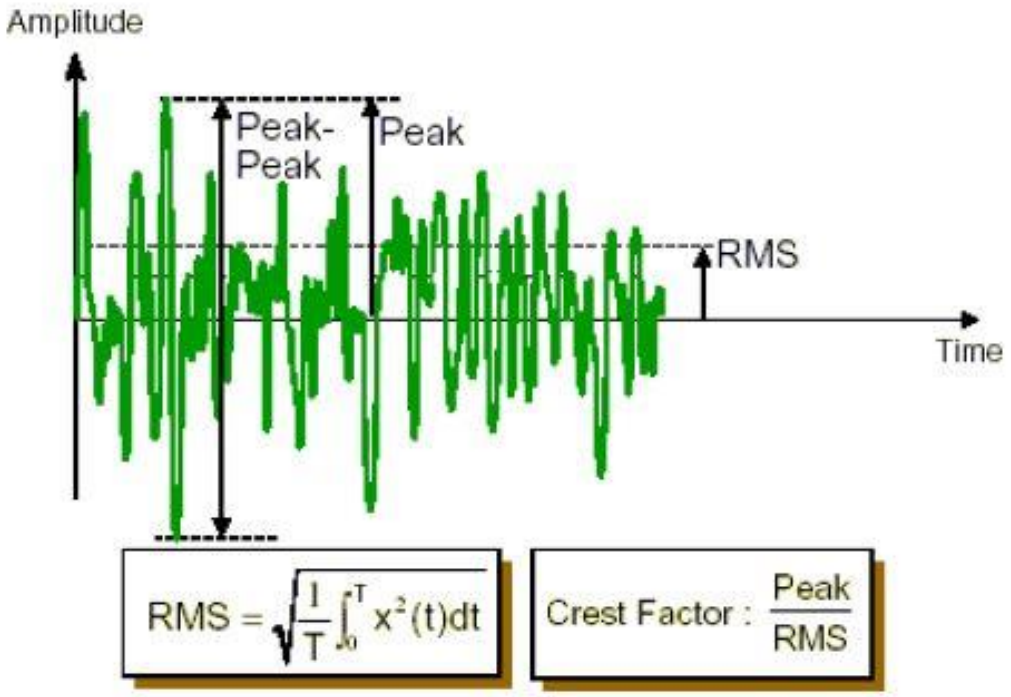


Figura 4: RMS, Peak-Peak, Peak e CF per un segnale random

Kurtosis

È un indice statistico, frutto di un algoritmo di calcolo basato su un determinato range di frequenze del segnale temporale, che è utile ai fini della definizione del carattere impulsivo del segnale. È adimensionale ed è definito come il quarto momento statistico della distribuzione dei dati. Confronta l'attuale distribuzione dei dati con una distribuzione Gaussiana, per la quale il kurtosis è 3, mentre può raggiungere valori di oltre 50 in presenza di fenomeni impulsivi.

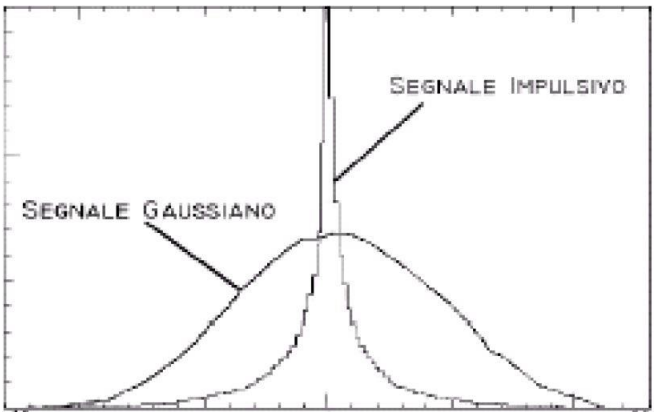


Figura 5: Confronto della distribuzione fra segnale impulsivo e gaussiano

$$Kurtosis = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{x(i)-\mu}{\sigma}\right)^4$$

Questo parametro, molto sensibile agli impulsi, non è adatto come parametro diagnostico per macchine che generano impulsi periodici, mentre è vantaggioso per rilevare difetti su macchine rotanti insieme ad esempio all'RMS. Su una trasmissione a ruote dentate ad esempio, in presenza di forti danneggiamenti il kurtosis può tendere a diminuire, mentre l'RMS continua ad aumentare costantemente.

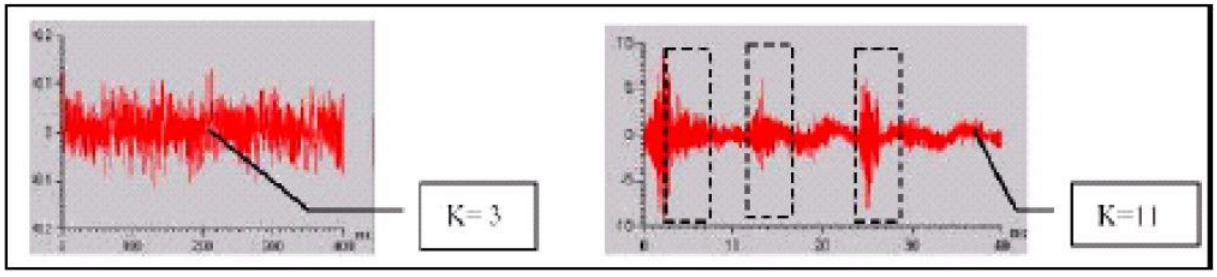


Figura 6: Rappresentazione e valore tipico del kurtosis

1.2.2 ***Analisi spettrale***

Ai fini del condition monitoring è importante avere la capacità di rilevare tutti i tipi di difetto a partire da quelli a bassa frequenza (disalineamenti, sbilanciamenti, cinghie difettate ecc.) fino a quelli ad alta frequenza (difetti sugli ingranamenti e sui cuscinetti, frequenze legate alle pale di organi rotanti ecc.). L'utilizzo delle informazioni contenute nello spettro consente una maggiore capacità analitica delle difettosità, ma gli strumenti generali presentati di seguito vanno applicati in modo specifico e dedicato alle singole applicazioni per ottenere buoni risultati in termini di mancate diagnosi e

falsi allarmi. In generale si può affermare che i metodi di condition monitoring che sfruttano l'analisi spettrale e derivati si basano sempre sulla comparazione dello stato vibrazionale attuale della macchina con uno spettro di riferimento rappresentativo di una buona condizione della macchina.

FFT (Fast Fourier Transform)

È la procedura che consente il calcolo rapido di dello spettro di ampiezza e fase di un segnale non periodico, finito e discreto. Questa tecnica introdotta negli anni '60 ha permesso di ridurre il numero di operazioni necessario al calcolo della DFT (Discrete Fourier Transform) da N^2 a $N*\log_2(N)$.

La trasformata discreta di Fourier, eseguita quindi su un segnale campionato, è definita come:

$$X(k\Delta f) = \Delta t \sum_{n=0}^{N-1} x_n e^{\frac{-i2\pi kn}{N}}$$

-Δf è la risoluzione dello spettro
-x_n è l'n-esimo campione di x(t), segnale continuo nel tempo
-N è il numero di campioni
-k è l'ordine dell'armonica dello spettro che va da 0 a (N-1)/2;

Supponendo di aver eseguito la trasformata su un periodo temporale T con frequenza di campionamento Fs (N campioni acquisiti), la risoluzione spettrale ottenuta sarà:

$$\Delta f = \frac{1}{T} = \frac{Fs}{N}$$

Mentre si avranno informazioni in frequenza fino alla frequenza di Nyquist:

$$f_{max} = Fs/2$$

Se si vuole avere una risoluzione spettrale alta quindi è necessario che il buffer di analisi sia temporalmente lungo, mentre se si vogliono ottenere informazioni ad alta frequenza la Fs dovrà essere alta. Ovviamente utilizzare un numero di campioni molto alto per l'analisi, corrispondente ad un tempo d'acquisizione ampio, significa esporsi al rischio che le condizioni di funzionamento varino significativamente all'interno del buffer campionato, rendendo poco utile l'analisi in caso di bruschi transitori di velocità.

È importante che durante il campionamento sia rispettato il criterio di Nyquist per non avere aliasing sullo spettro misurato: $fmax_{segnale} < 1/2\ Fs$.

Ciò che si ottiene dalla trasformata di Fourier è una funzione complessa, da cui si può risalire ai due spettri di **ampiezza** e **fase** del segnale.

$$|X(f)| = \sqrt{Re\big(X(f)\big)^2 + Im\big(X(f)\big)^2}$$

$$\Phi = \tan\left(\frac{Im(X(f))}{Re(X(f))}\right)$$

Riferendosi allo spettro d'ampiezza ottenuto esso è doppio, cioè contiene informazioni pari alla metà dell'energia complessiva del segnale oltre la frequenza si Nyquist. A partire dall'output della FFT per ottenere l'ampiezza occorre eliminare il semispettro negativo, moltiplicare per due il risultato e dividerlo per il numero di campioni:

$$Spettro\ d'ampiezza\ (peak\_value) = 2\,\frac{|X(k\Delta f)|}{N}$$

con k=[1:N/2]

Questo appena ottenuto è lo spettro d'ampiezza positivo. Considerando come riferimento di ingresso alla FFT una sinusoide di ampiezza 3V a frequenza 100Hz si ottiene in uscita una banda spettrale a 100Hz di ampiezza pari a 3V, cioè il valore di picco della componente a 100Hz. Alcuni linguaggi di programmazione forniscono invece come uscita della FFT il valore efficace corrispondente:

$$Spettro\ d'ampiezza\ (RMS) = \sqrt{2}\frac{|X(k\Delta f)|}{N}$$

con k=[1:N/2]

In realtà se stiamo eseguendo il calcolo su un periodo T, tale per cui 100Hz non è un multiplo della risoluzione spettrale 1/T, troveremo l'ampiezza distribuita sulle bande spettrali laterali e non come unica linea spettrale. Per diminuire questo fenomeno detto leakage in genere si usano delle finestre che diminuiscono la dispersione. Ovviamente essendo il segnale moltiplicato per dei coefficienti moltiplicativi compresi tra 0 e 1, vanno apposti dei coefficienti correttivi per evitare grossi errori sull'ampiezza delle bande spettrali misurate. La finestra più comunemente utilizzata è quella di Hanning.

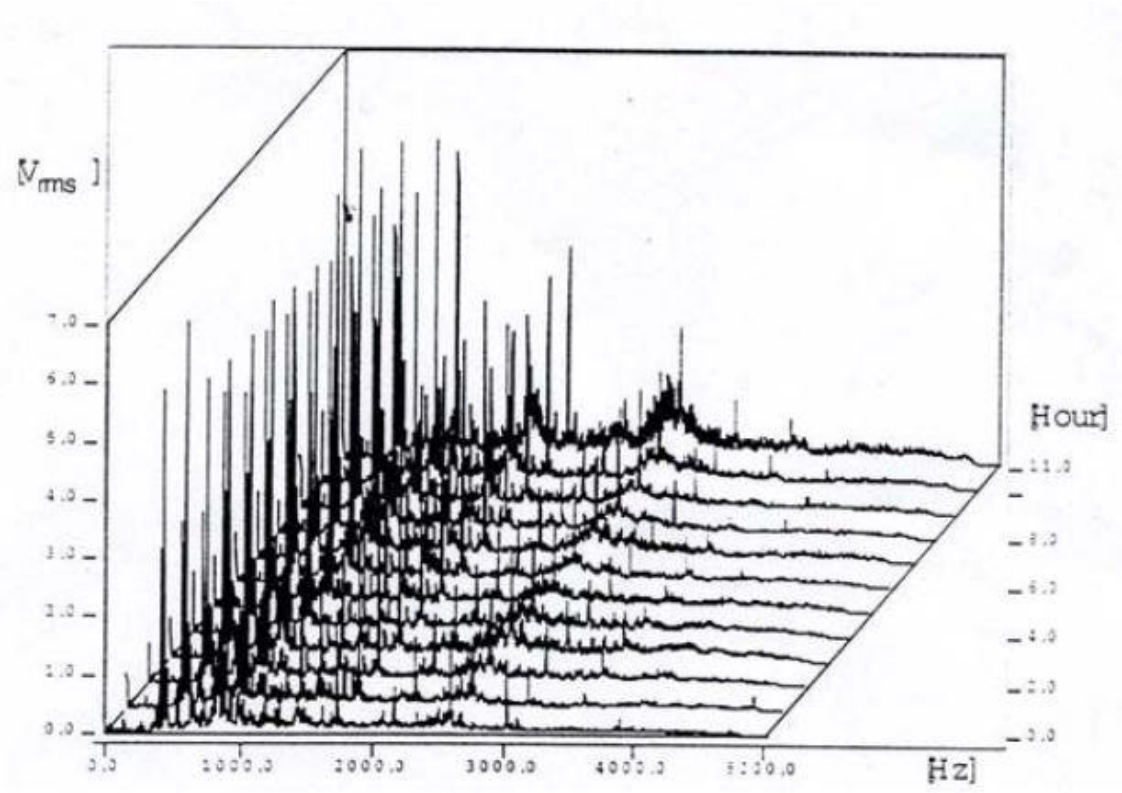


Figura 7: Spettro d'ampiezza in un plot "waterfall" che ne mostra l'andamento nel tempo

PSD (Power Spectral Density)

La trasformata di Fourier della funzione di autocorrelazione R_{xx} di un segnale x(t) è detta autospettro o densità di potenza spettrale. Definita la funzione di autocorrelazione, che indica quanto una funzione è correlata con se stessa, come:

$$R_{xx}(\tau) = lim_{T\to\infty} \frac{1}{T}\int_{-T/2}^{T/2} x(t)x(t+\tau)dt$$

La PSD è definita come:

$$S_{xx}(\omega) = F\{R_{xx}(\tau)\}$$

Che è legata alla trasformata di Fourier di x(t) dalla relazione:

$$S_{xx}(\omega) = |X(\omega)|^2$$

Di fatto è il corrispettivo dello spettro d'ampiezza (non possiede informazioni sulla fase) elevato al quadrato nell'ampiezza, e rappresenta il contenuto in potenza del segnale alle diverse frequenze.

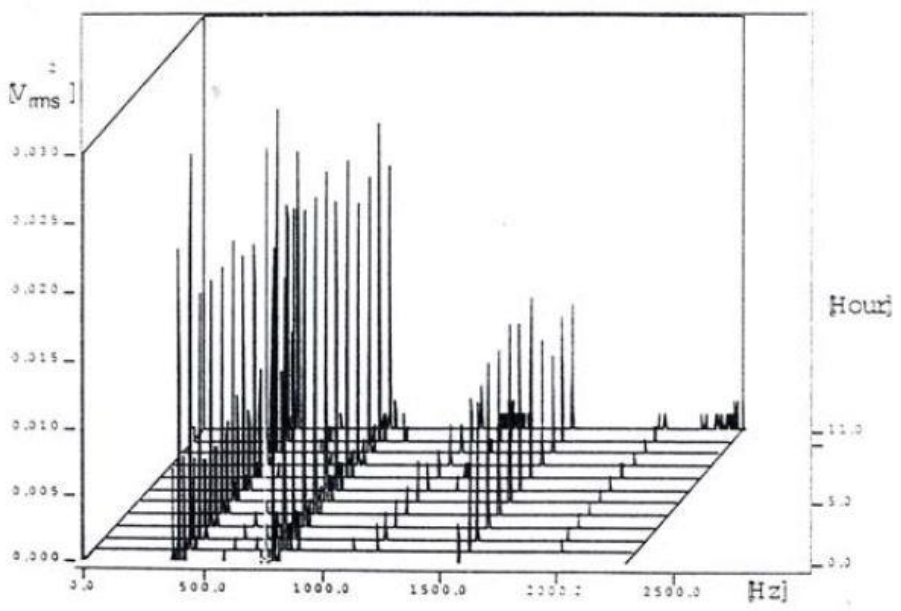


Figura 8: PSD corrispondente alla figura7. Fornisce una rappresentazione molto più pulita delle componenti.

Order Spectrum

Normalizzando lo spettro sulla velocità di rotazione principale della macchina si può ottenere lo spettro degli ordini. Consente di poter valutare fenomeni che avvengono in modo sincrono con la velocità di rotazione e suoi multipli.

Figura 9: Il passaggio dalle frequenze agli ordini avviene semplicemente normalizzando l'asse delle frequenze sulla velocità di rotazione

Se il campionamento anziché avvenire su base tempo avviene su base angolo, ciò che si ottiene dalla FFT del segnale campionato ad angolo costante è esattamente uno spettro degli ordini la cui risoluzione è data da: $\frac{campioni/giro}{n^o\ di\ campioni}$.

Eseguendo l'STFT (Short Time Fourier Trasform), cioè ripetendo l'FFT su piccoli buffer del segnale e disponendo dei tempi e delle velocità di rotazione corrispondenti è possibile costruire i seguenti grafici, utili ai fini diagnostici.

Tempo-frequenza

Permette di visualizzare sinteticamente l'andamento delle ampiezze corrispondenti alle varie frequenze. È possibile seguire nel tempo la frequenza principale di rotazione della macchina ed i suoi principali ordini. Rappresenta una cascata di spettri calcolati ad intervalli costanti. La rappresentazione può essere anche del tipo "waterfall" e spesso ci si riferisce a questo tipo di analisi come spettrogramma.

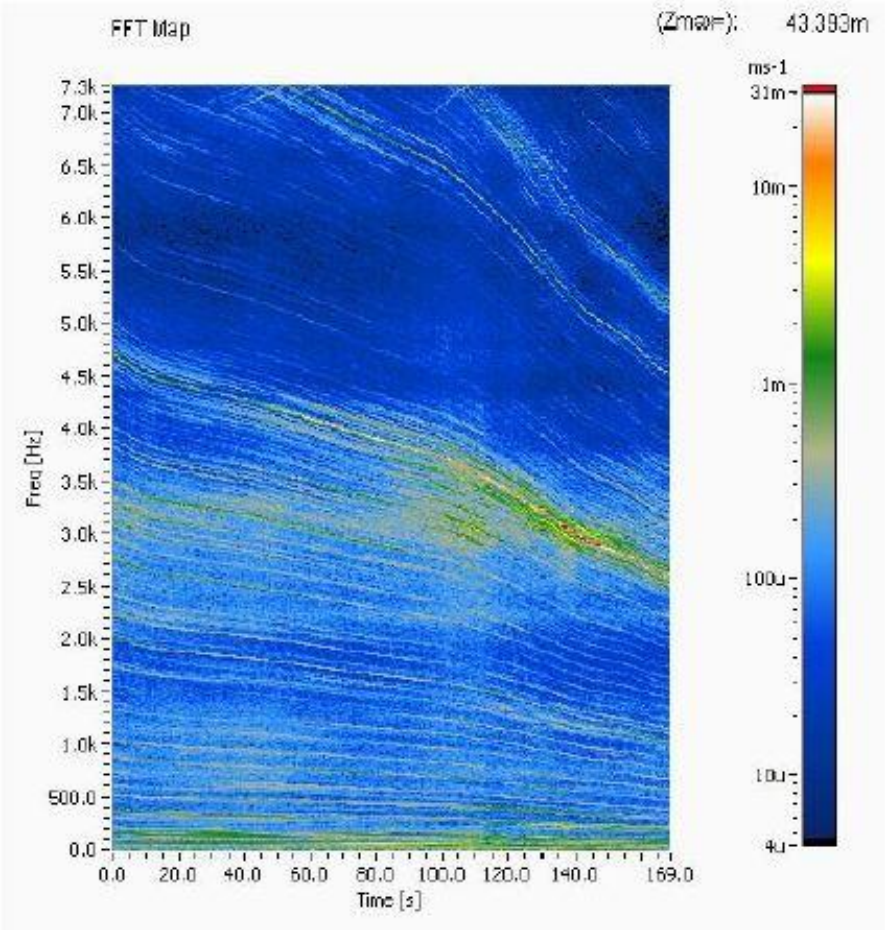


Figura 10: Colormap Tempo-Frequenza

Diagramma di Campbell

Se al posto del tempo sull'asse delle ascisse si mettono le velocità di rotazione corrispondenti all'esecuzione delle FFT, quello che si ottiene è il diagramma di Campbell. Consente facilmente di identificare le frequenze naturali eccitate dal processo vibratorio. Ad esempio può essere applicato all'analisi di un transitorio di salita o di discesa per identificare le risonanze di un giunto di trasmissione.

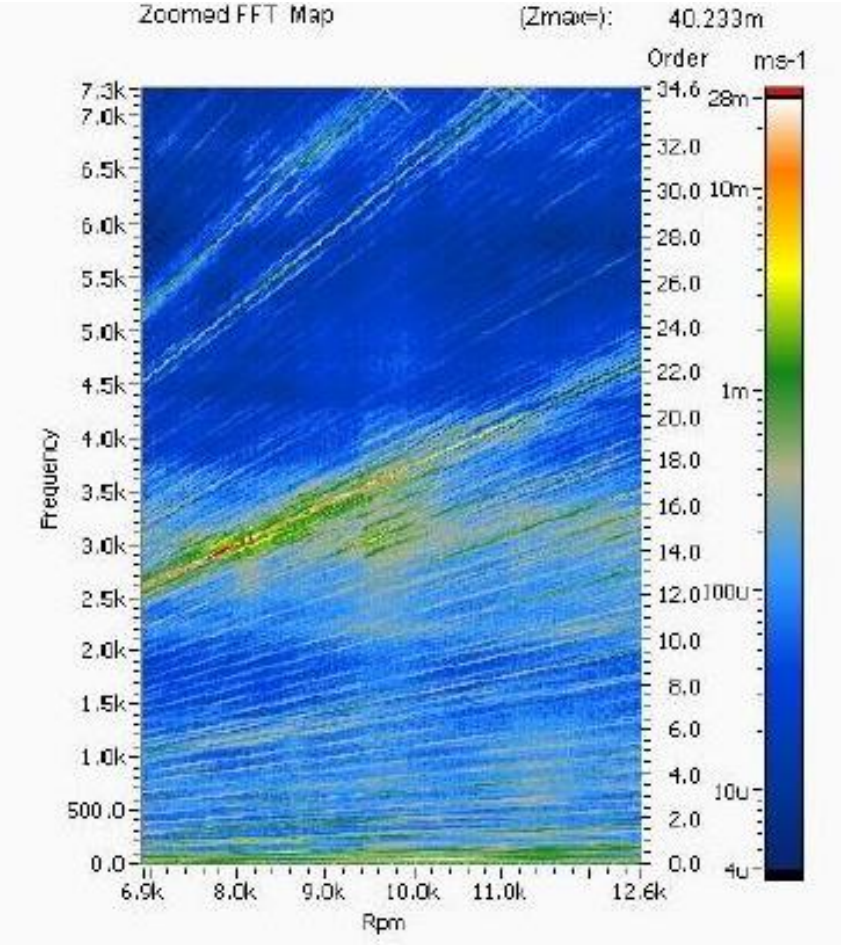


Figura 11: diagramma di Campbell velocità-frequenza

Order Colormap / Waterfall

Normalizzando l'analisi Tempo-Frequenza sulle velocità di rotazione si ottiene una cascata di spettri degli ordini nel tempo. Questo è particolarmente utile per seguire il progredire di difetti su componenti che ruotano vincolati alla macchina e che presentano vibrazioni a determinati ordini.

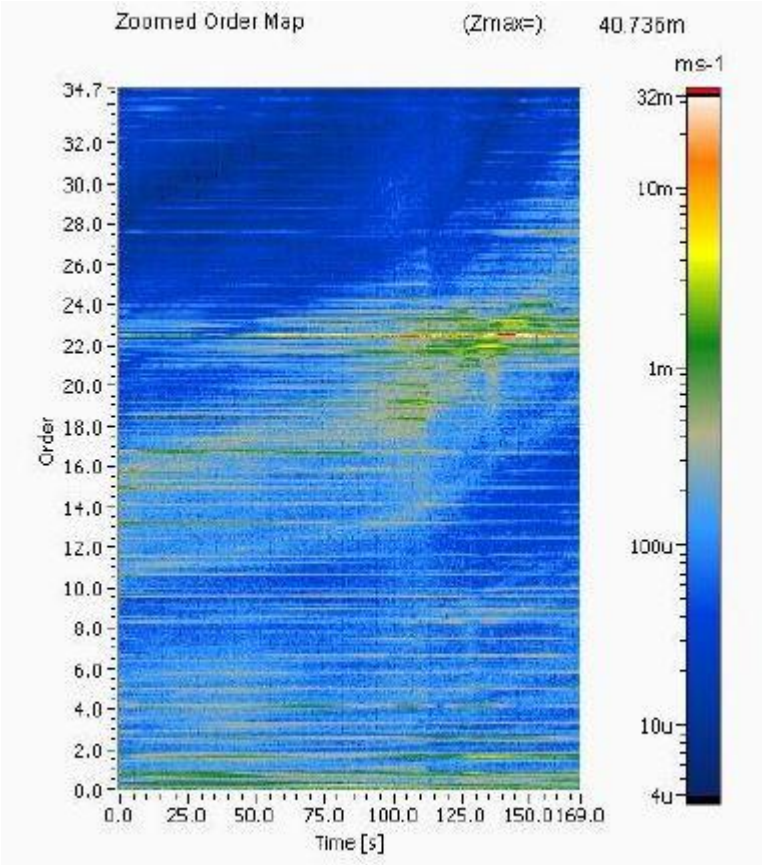


Figura 12: Order colormap

1.3 OBIETTIVI E ESIGENZE DEL CASO DI STUDIO PARTICOLARE: PROPULSORE AL BANCO PROVA / TEST SU VEICOLO

L'attività svolta in questa tesi è nata dalla volontà dell'azienda Alma-Automotive di sviluppare un sistema di condition monitoring da destinare all'impiego nella propria sala prove. Attualmente al banco viene testato un motore motociclistico bicilindrico derivato dalla serie, che può venire sottoposto a stressanti prove di durata in condizioni estremamente dinamiche. Lo scopo che si vuole raggiungere con il sistema di monitoraggio è quello di poter disporre di alcuni indici sintetici rappresentativi della condizione del propulsore, in modo da poter intervenire in tempo in caso di guasti incipienti o di programmare le prove e gli interventi da eseguire sull'esemplare di propulsore in uso. Il vantaggio di un sistema che fornisca un output di questo tipo, come visto in precedenza, è fondamentalmente quello di evitare rotture catastrofiche, che spesso impediscono anche la corretta diagnosi delle cause del guasto durante le analisi eseguite allo smontaggio del motore. Inoltre disporre di un indice relativo allo

stato di salute del propulsore, con i dovuti riferimenti può rendere possibile valutare la vita utile residua.

La principale peculiarità di questa applicazione rispetto alle più tipiche applicazioni del condition monitoring è l'estrema dinamicità delle prove subite dal motore in prova. Infatti la sala è allestita con una macchina elettrica, che si interfaccia al motore termico attraverso un giunto elastico, in grado sia di frenare che di trascinare il propulsore, rendendo possibile la simulazione di profili di velocità di rotazione e di carico del tutto simili a quelli che si avrebbero sul veicolo. La presenza di simulazioni di cambi marcia e brusche frenate costituisce un vincolo piuttosto stringente per eseguire analisi in frequenza, che devono necessariamente essere eseguite su buffer temporali ridotti per evitare di essere eccessivamente condizionate dai trend introdotti dai transitori. A priori non si possono nemmeno trascurare gli effetti sullo spettro vibrazionale introdotti dai contraccolpi dovuti alle brusche decelerazioni ed accelerazioni.

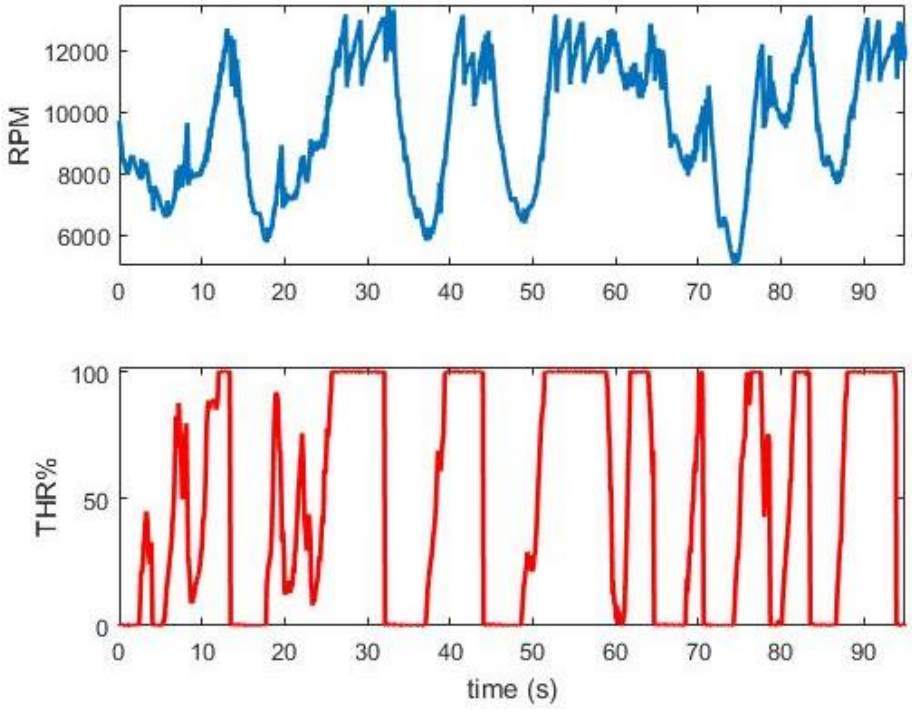


Figura 13: esempio di un profilo RPM-carico% durante una prova di durata

Ovviamente non è possibile utilizzare gli stessi metodi utilizzati per monitorare un motore elettrico che funziona ininterrottamente a regime costante od un grande motore industriale destinato alla produzione energia. Al sistema di condition monitoring è quindi richiesta la capacità di seguire l'estrema dinamicità delle prove e fornire output

sensati con le condizioni di funzionamento “istantanee” del motore. Questo può essere possibile rendendo minimi i buffer di dati su cui eseguire i calcoli e sfruttando la comunicazione con sensori, banco prova e centralina motore per avere in tempo reale il riferimento dello stato di funzionamento del motore. Insieme all’azienda Alma-Automotive è stato deciso di avviare lo sviluppo di un software a tale scopo basandosi sulla già esistente piattaforma hardware Miracle^2 (descritta nel capitolo 2), capace di provvedere alla potenza di calcolo necessaria, agli input-output ed ai protocolli di comunicazione maggiormente diffusi in ambito automotive.

In conclusione, ciò che ci si è prefissati come obiettivo di questa tesi è un sistema software di base, che consenta di eseguire analisi in real-time valutando l’applicazione e la robustezza (falsi allarmi e mancate diagnosi) di alcuni criteri di diagnosi basati su indici sintetici da confrontare con una baseline a motore nuovo. L’andamento del trend di usura deve poter essere visualizzabile e eventualmente utilizzabile per stoppare l’esecuzione della prova in corso. L’ambito operativo previsto in questa tesi è quello della sala prove, ma, così come già avviene per il software OBI di Alma-Automotive, in futuro è prevista la possibilità di applicazioni on-board vista la compattezza e l’apparato di comunicazione del microcontrollore Miracle^2.

1.4 REILHOFER DELTA ANALYSER

Volgendo lo sguardo a ciò che offre oggi il mercato per il condition monitoring in ambito automotive spicca il prodotto delta-Analyser dell’azienda tedesca Reilhofer KG. Si tratta di un sistema a rack, da affiancare al banco prova, che basa i propri strumenti di diagnosi sul rilevamento delle vibrazioni strutturali del powertrain tramite accelerometri, acquisendo in parallelo altre informazioni fondamentali a stabilire il punto di funzionamento del motore come rpm, carico e temperature. È in grado di operare continuativamente ed in condizioni estremamente dinamiche su motori alternativi a combustione interna, trasmissioni, motori elettrici, turbine, compressori, macchine idrauliche e alberi rotanti. Visto l’elevato ingombro e la massa di circa 14 kg, questo strumento tuttavia risulta poco adatto per applicazioni on-board, su auto e moto oppure in applicazioni che richiedono strumentazione compatta e versatile.

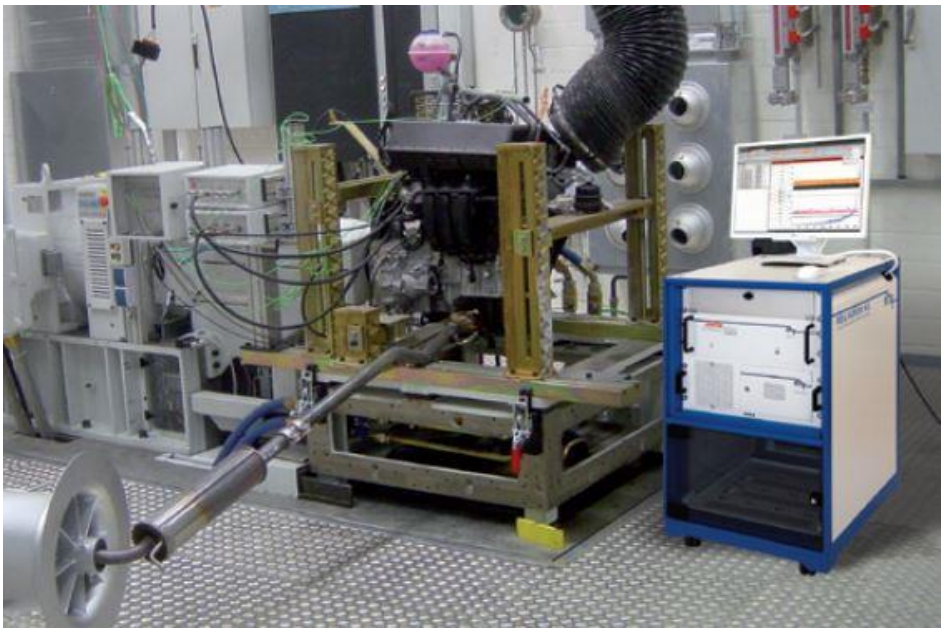


Figura 14: Reilhofer delta-Analyser installato in un banco prova

La filosofia di base del sistema di diagnosi si basa sull'apprendimento del comportamento vibrazionale di base del motopropulsore in condizioni ottimali, restituendo in tempo reale indici rappresentativi dello stato del motore, confrontati continuamente con delle soglie di allarme sulla base della condizione di funzionamento istantanea. Sono implementate diverse strategie di monitoraggio basate sia sull'analisi degli ordini, per rilevare danneggiamenti meccanici e variazioni sulla combustione, che sul segnale nel dominio del tempo, come la presenza di forti picchi o di danneggiamenti a bassa energia. Un comportamento vibrazionale differente da quello atteso viene segnalato come allarme e può essere utilizzato per bloccare l'esecuzione del test. L'analisi dei dati è agevolata da un software dedicato che permette diverse visualizzazioni come spettrogrammi, waterfall diagram e quella dei trend-index, oltre ad essere provvisto di uno strumento denominato Order-Calculator, predisposto al calcolo della previsione degli ordini eccitati a partire dalla struttura meccanica di motore, trasmissione e ausiliari.

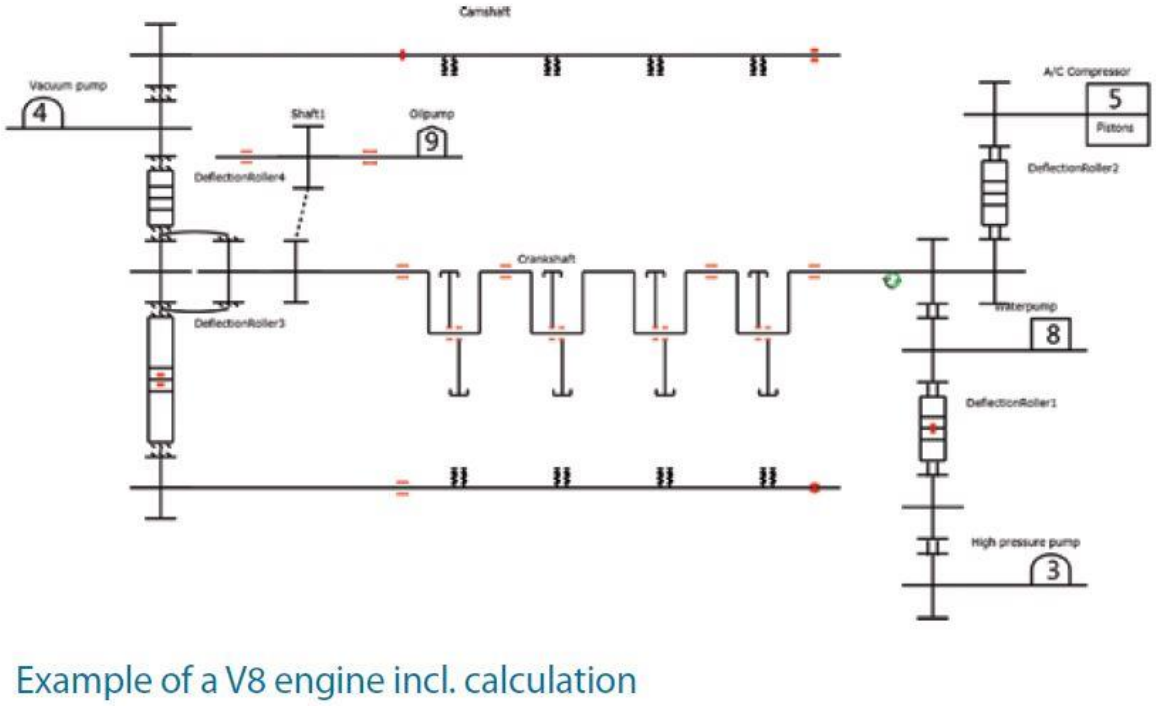


Figura 15: schema costruito attraverso l'Order-Calculator

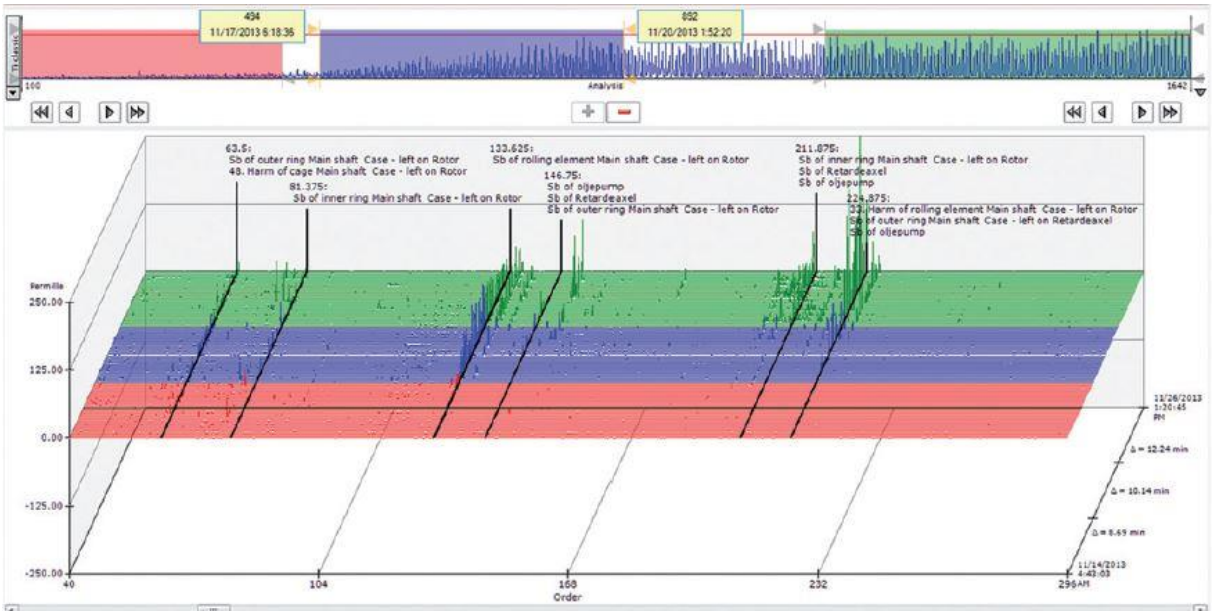


Figura 16: waterfall plot dell'analisi degli ordini. Sono evidenziate da bande nere gli ordini corrispondenti a componenti specifici del motore, ottenuti mediante lo strumento Order-Calculator

Nel software è inclusa anche la costruzione di un database per i guasti (Global Failure Database), utile per tenere traccia degli episodi verificatisi e poter svolgere analisi comparative o statistiche sui guasti e sui componenti da essi interessati.

1.4.1 ***Canali e segnali acquisiti***

Per permettere la diagnosi in tempo reale sono necessari fondamentalmente i segnali relativi a:

-stato vibrazionale
-velocità di rotazione
-carico
-temperatura olio/acqua ed altre informazioni che possono essere utili per mappare la condizione di funzionamento

In figura 7 si può osservare un esempio di set up per il monitoraggio di un gruppo motore/trasmissione attraverso l'acquisizione dei segnali accelerometrici sulle carcasse, delle velocità di rotazione e da altri dati che possono essere acquisiti via analogica, digitale o via CAN comunicando con il banco.

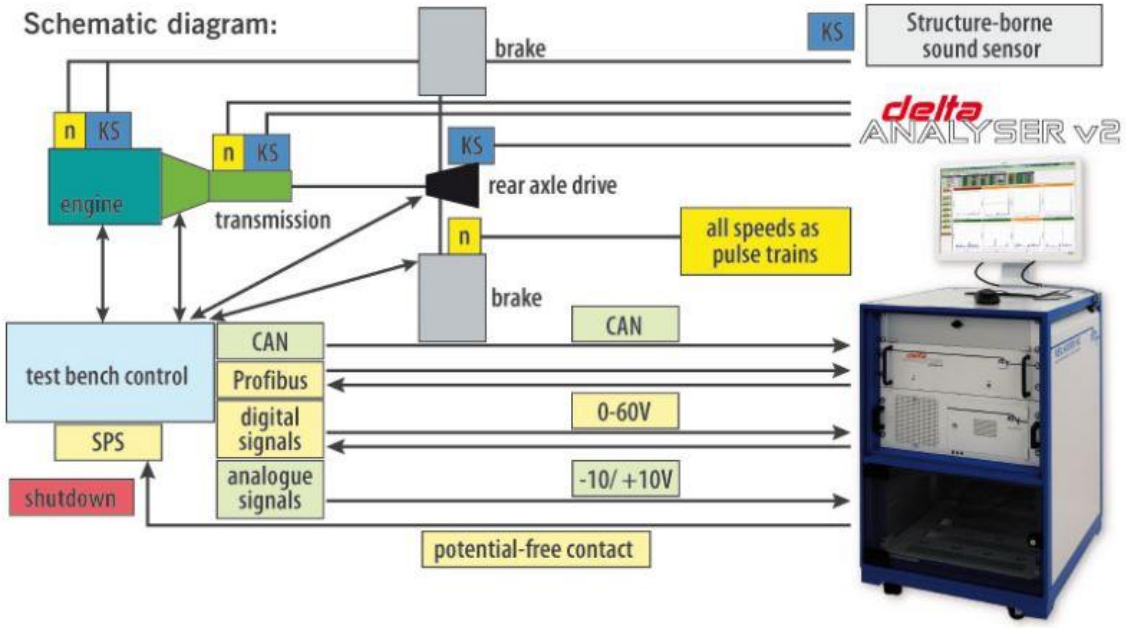


Figura 17: delta-Analyser set up

Il sistema di acquisizione/output è composto da due elementi fondamentali: il modulo delta-Analyser propriamente detto, dedicato ad ingressi ed uscite analogico/digitali e lo speedBOX, predisposto per l'acquisizione dei segnali di ruota fonica o encoder utilizzati per risalire alle velocità di rotazione.

Figura 18: delta-Analyser e speedBOX

delta-Analyser:

È il cuore del sistema di acquisizione ed ha ingressi dedicati agli accelerometri. In un set up standard la maggior parte degli ingressi utilizzati per stabilire il punto di funzionamento sono di tipo analog-input.

input	analog	• 4 ch accelerometer signal • 2 - 8 ch generic	Samplig rate max: 50 kHz Input voltage: ± 11.25 V A/D converter: 16 bit Adjustable low-pass filter Adjustable high-pass filter
	digital	• 16 ch	Input voltage : ± 32V Sampling rate max: 25 kHz
output	analog	• 8 ch	Output voltage: ± 10.92 V
	digital	• 16 ch	Output voltage: 5.5 - 28 V
	Ethernet	100 / 1000 Mbit / s	
	Serial	RS 232	
interface	CAN	2.0B, extended frames, basic frames	

Tabella 1: input/output delta-Analyser

I canali dedicati agli accelerometri possono gestire diversi tipi di accelerometri tra cui quelli con elettronica di pre-condizionamento integrata (ICP, IEPE) e consentono di eseguire sul segnale un filtraggio configurabile. Inoltre sono quelli su cui viene spesa la maggior parte della potenza di calcolo per eseguire l'analisi spettrale attraverso l'FFT.

speedBOX:

È il modulo dedicato all'acquisizione dei segnali provenienti da ruote foniche VRS, ad effetto hall o encoder. La velocità di rotazione del motore può essere acquisita singolarmente o abbinata ad un segnale per la fasatura, come quello proveniente da una ruota posta sull'albero a camme o prelevando il segnale di fase dagli impulsi di corrente sulle bobine.

input	analog	• 4 x rotational speed	Samplig rate max: 40 MHz Input voltage: up to 100 V A/D converter: 10 bit Connectible low-pass filter 100 kHz
	digital	• 4 x rotational speed (TTL)	Input voltage: 0 / 5 V (TTL) Frequency: 0 Hz to 1 MHz
output	digital	• 2 x 5 rotational speed (TTL)	Output voltage: 0 / 5 V (TTL) Max frequency: 2.5 MHz
interface	USB	For confguration of the speedBox	

Tabella 2: input/output speedBOX

1.4.2 *Tecniche di diagnosi: Trend_Index*

Differentemente da quello che viene comunemente eseguito sulla maggior parte dei sistemi di monitoraggio industriale e che viene indicato in normativa ISO 20816, i metodi di diagnosi non si basano esclusivamente sull'analisi dell'RMS del segnale, in quanto adatto solamente all'identificazione di danni che presentano una notevole

intensità di rumore strutturale. Nella maggioranza dei casi questo significa accorgersi del guasto in modo tardivo e senza aver la possibilità di osservare il progresso del danneggiamento né la sua origine. Solamente dei criteri diagnostici basati sull'analisi spettrale possono essere d'aiuto per questo scopo. Anche un componente in perfetto stato produce rumore e sfortunatamente spesso non differisce troppo da quello di uno danneggiato se osservato nel dominio del tempo. Passando al dominio in frequenza o degli ordini, ogni componente spettrale viene evidenziata, aumentando la sensibilità ai danneggiamenti incipienti. Dato che diversi tipi di danneggiamento producono pattern diversi sul segnale accelerometrico, è necessario avere diversi metodi di diagnosi per ottenere i migliori risultati.

Test method	To find what?
Order analysis – absolute	Rotationally symmetric faults
Order analysis – relative	Rotation dependent scraping and squeaking
Frequency analysis – absolute, relative	Static tests without rotational speed
Time signal classification	Run-out detection
Spectral line monitoring – distinct lines	Separate monitoring of individual components
Spectral line monitoring – special range	Monitoring of special resonances
Resonance analysis	Loudness-independent faults
720° Cylinder synchronous measurement	Disturbances of energy distribution over crank angle
CrashPreventer	Fast break shutdown
Noise shaping analysis	Mixed friction detection
Monitoring Windows	Angle-dependent event monitoring
Curtosis analysis	impact detection

Tabella 3: tecniche di diagnosi implementate ed eventi a cui sono dedicate

Per i guasti più rumorosi, come la rottura di una biella o di una ruota dentata, è possibile attuare un semplice controllo sul valore efficace del segnale, forzando lo stop del test al verificarsi del superamento di una soglia preimpostata. Un intervento di questo tipo può comunque salvare la macchina da ulteriori danni che potrebbero compromettere la diagnosi delle cause della rottura.

- Questa funzione è implementata attraverso il **Crash-Preventer**, dedicato al riconoscimento delle rotture più rapide e rumorose.

Se è importante non solo fermare la prova all'occorrenza di un guasto, ma seguirne anche lo sviluppo ed il deterioramento l'utilizzo dell'RMS non è sufficiente. L'analisi degli ordini è il miglior metodo per seguire i test dinamici, osservando lo spettro delle vibrazioni in modo indipendente dalla velocità di rotazione. Il delta-Analyzer utilizza due differenti metodi:

- **Analisi degli ordini assoluta**, pensata per osservare l'usura meccanica ed i cambiamenti nella combustione.
- **Analisi degli ordini relativa**, dedicata al rilevamento di strisciamenti, raschiamenti ecc.

- **Analisi in frequenza,** non necessita dell'informazione sugli rpm ed è dedicata ai test statici.

Alcuni danneggiamenti tuttavia possono essere avvertibili ad orecchio da un operatore umano, ma risultare non evidenti nell'analisi spettrale. Questi difetti emettono poca energia e sono visibili solamente nel dominio del tempo.

- La **Time-Signal-Classification** è pensata per diagnosticare danneggiamenti a bassa energia.

- La **Curtosis analysis** è finalizzata al rilevamento di danneggiamento che causano impatti ripetuti.

Tuttavia in alcuni test è richiesto solamente di seguire un singolo componente e fermare la prova al minimo cambiamento. Per questa applicazione è stata sviluppata l'analisi della singola linea spettrale, basata su due metodi differenti: sull'energia di vibrazione e su limiti di tolleranza.

- La **Spectral line monitoring** è dedicata al monitoraggio di singoli componenti.

Tutti I metodi descritti precedentemente richiedono una certa intensità per essere efficaci. Esistono casi in cui il rumore prodotto dal difetto è del tutto trascurabile rispetto al rumore strutturale prodotto dal motore.

- La **Relative sector analysis** è pensata per diagnosticare i danneggiamenti in modo indipendente dall'intensità della vibrazione.

E' possibile anche monitorare il segnale accelerometrico su ogni singolo cilindro durante i 720 gradi del ciclo. Si possono ad esempio ottenere informazioni riguardo ad irregolarità nell'accensione e tenendo conto dell'ordine di combustione si riesce a risalire al cilindro interessato.

- La **720 degree cylinder synchronous measurement** consente di osservare la distribuzione dell'energia durante un ciclo motore.
- Attraverso la funzionalità **monitoring window** è possibile tenere traccia di eventi che si verificano a cadenza angolare fissa, come ad esempio la chiusura delle valvole.

Queste analisi tuttavia, se valutate in senso assoluto, non producono alcuna informazione relativa al degrado meccanico del sistema in esame. A tal fine viene eseguita una fase d'apprendimento, generalmente a motore nuovo o alla sostituzione di alcuni componenti, finalizzata alla costruzione di un riferimento del rumore strutturale prodotto durante le condizioni "normali" di funzionamento del motore. Il comportamento vibratorio di base durante questa fase viene mappato sulle condizioni di funzionamento motore, acquisite dal sistema delta-Analyser per mezzo di ingressi predisposti all'acquisizione della velocità di rotazione, del carico ed eventualmente di altre informazioni come le temperature dei fluidi. La "baseline" così costruita è composta da spettri degli ordini ottenuti a partire dall'analisi in frequenza del segnale accelerometrico congiuntamente con la velocità di rotazione corrispondente alla misurazione ed è utilizzata come riferimento per un indice di degrado chiamato **trendindex**.

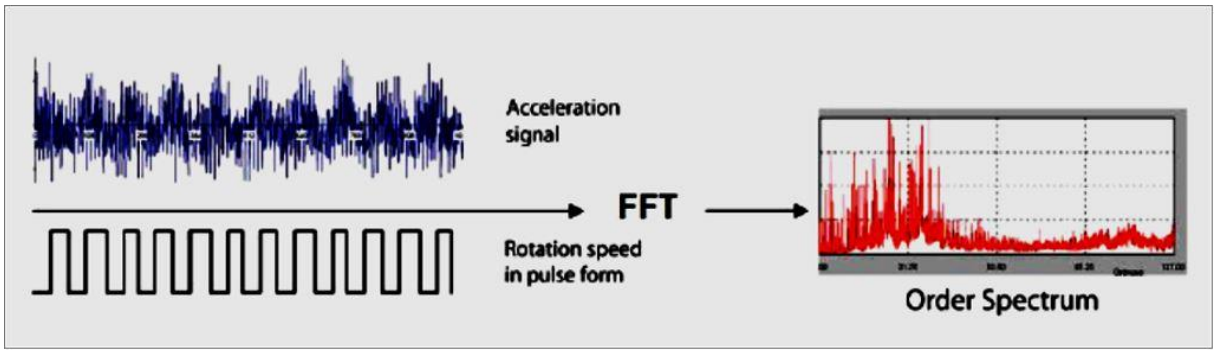


Figura 19: Costruzione dello spettro degli ordini a partire dal segnale accelerometrico e della ruota fonica / encoder

Durante questa fase vengono anche calcolate attraverso metodi statistici le tolleranze superiori ed inferiori corrispondenti allo spettro registrato, questo per tenere in considerazione l'inevitabile deviazione del reale funzionamento dal riferimento costruito durante l'apprendimento. Una volta in possesso di questi dati è possibile passare alla fase di monitoraggio vera e propria che prevede la costruzione di un indice di trend del danno nel tempo (trendindex), ottenuto confrontando continuamente lo spettro attuale del segnale vibratorio con quello della baseline. In definitiva sono previste 3 fasi del funzionamento del sistema delta-Analyser:

- **Fase di apprendimento** in cui è costruita la mappa di riferimento
- **Fase mista** in cui si inizia a calcolare il trendindex e si fissano le tolleranze
- **Fase di monitoraggio** vero e proprio

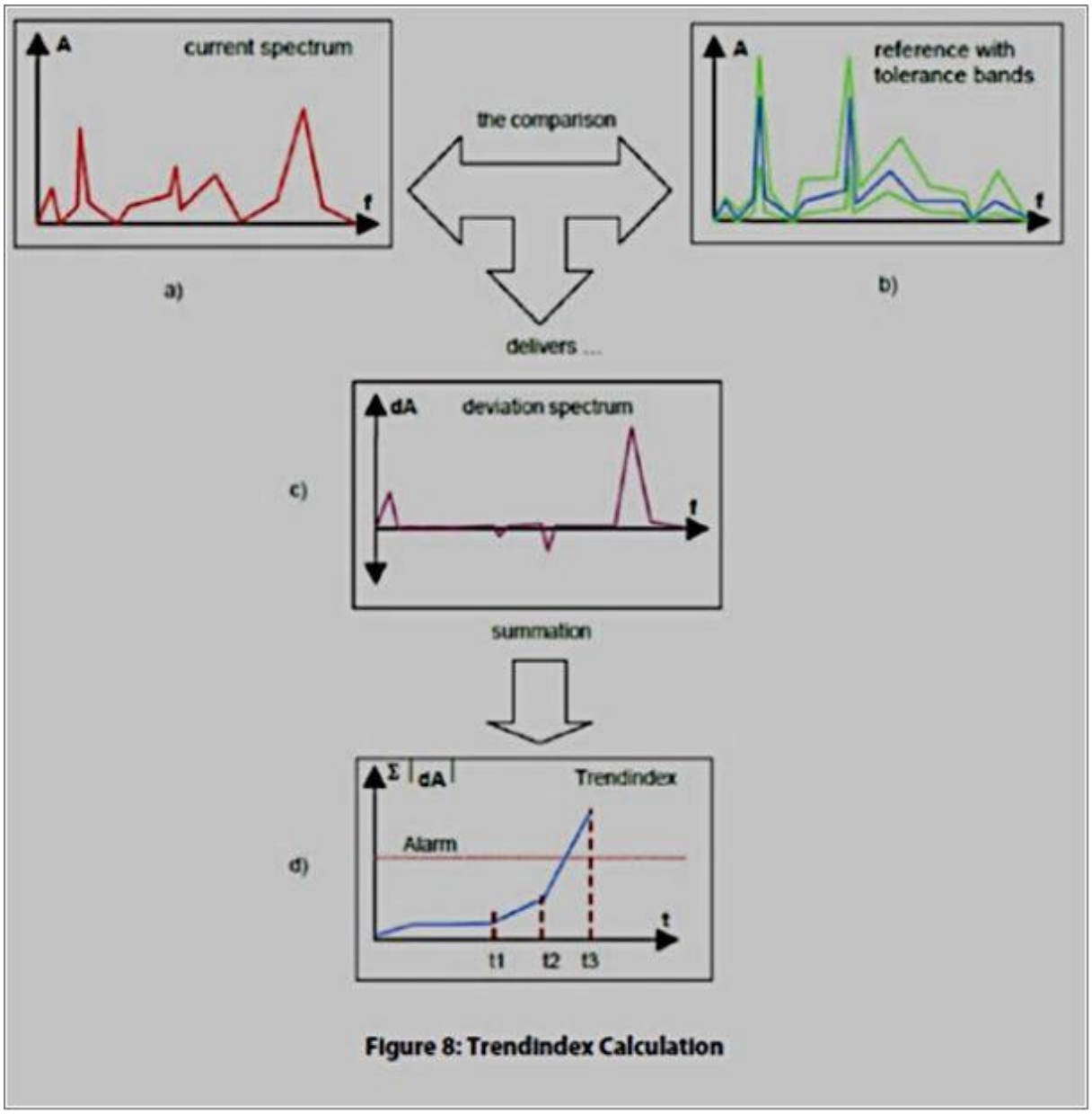


Figura 20: Costruzione del trendindex a partire dallo spettro di riferimento e dallo spettro attuale

Sebbene la procedura di calcolo dettagliata del trendindex sia una informazione riservata del produttore del sistema, esso è sostanzialmente calcolato come la sommatoria delle differenze fra lo spettro attuale e quello di riferimento che eccedono le tolleranze. Impostando una soglia su tale indice è possibile di disporre di un allarme o, se desiderato, dell'invio di un segnale che guidi lo stop del banco prova. In caso di rilevamento di un guasto è possibile analizzare offline l'andamento dello spettro nel tempo.

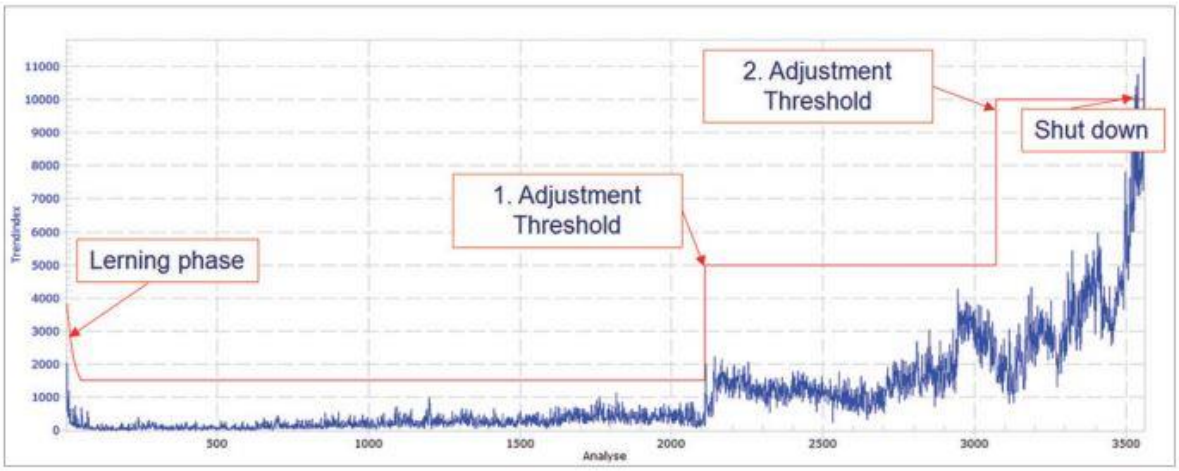


Figura 21: andamento del trendindex e della soglia nel tempo

Tutti questi metodi di diagnosi possono essere eseguiti contemporaneamente dal delta-Analyser. Ovviamente i dettagli delle tecniche di diagnosi sono di proprietà del produttore e non sono completamente disponibili.

Riferimenti

[11] [12] [5] [13] [14]

Capitolo 2
MIRACLE²

Il bisogno di una piattaforma hardware che consenta di far fronte con versatilità ad un panorama automotive in rapido mutamento può trovare una valida risposta in un'architettura di tipo real-time/FPGA, che consente di implementare soluzioni software affidabili e deterministiche grazie all'unione di un modulo FPGA (Field Programmable Gate Array), adatto alle interazioni ed ai calcoli di più basso livello, garantendo la velocità, il determinismo e l'isolamento fra le funzioni tipico di una soluzione hardware [15], unitamente alla programmabilità software, e di un processore operante con un sistema operativo real-time dedicato, che garantisce potenza e precisione nelle tempistiche per i calcoli. Ovviamente, per rendere le applicazioni compatibili con l'attività on-board è richiesto che l'hardware sia di piccole dimensioni, leggero, robusto e che sia in grado di lavorare anche in modalità stand-alone.

Attualmente l'azienda Alma Automotive rende disponibile il dispositivo Miracle^2 dotato di processore real-time ed unità FPGA, impiegato prevalentemente con software dedicato all'analisi di combustione (OBI-M2 "On-Board Indicating"), orientato sia ad applicazioni su veicolo che al banco prova. Miracle^2 si presenta come una piattaforma estremamente compatta utilizzabile per applicazioni come l'acquisizione ed il processamento Real-Time di dati o per il Control Prototyping. Notevoli potenzialità sono garantite dall'hardware che si caratterizza come una serie di input/output con potenza di calcolo disponibile, cosa che unitamente alla sua completa programmabilità in linguaggio Labview (anche Simulink, Matlab e Stateflow) garantisce grande versatilità software. La presenza di un modulo wifi e l'implementazione di numerosi protocolli di comunicazione standard come CAN, ethernet, XCP, CCP, RS232 ne consente inoltre la connessione e lo scambio di informazioni con i sistemi di bordo e di banco.

Figura 22: Miracle2, potenziali applicazioni

2.1 SISTEMA E SPECIFICHE TECNICHE

Il cuore della piattaforma Miracle^2 è costituito dalla SOM National Instruments che per la parte di calcolo si configura come l'unione un processore Dual Core ARM da 667MHz ed un modulo FPGA Artix-7. L'utilizzo di hardware e software National Instruments consente di sfruttare appieno le potenzialità e la versatilità del hardware per realizzare sistemi stand-alone embedded attraverso linguaggio grafico Labview.

Figura 23: SOM National Instruments

Il modulo FPGA (Field Programmable Gate Array) è un chip di silicio riprogrammabile che consente la stessa flessibilità garantita da un software che giro su un normale processore, con il vantaggio di non essere limitato nei processi dal numero di core. Il codice su un FPGA viene eseguito realmente in parallelo garantendo il determinismo e l'isolamento fra funzioni tipico delle soluzioni hardware. Queste caratteristiche consentono un'esecuzione del codice molto veloce, rendendo l'FPGA adatto a gestire soprattutto le funzioni di basso livello come l'input/output ed il pre-processamento dei segnali o calcoli di base da eseguire con estrema rapidità (clock 40MHz). Ovviamente il numero di gate logici è limitato, quindi durante la compilazione automatica del software per l'implementazione su FPGA viene verificata l'effettiva realizzabilità. I vantaggi che comporta l'implementazione in questo ambiente rendono importante adottare metodi di stesura del software atti a risparmiare il più possibile spazio da rendere disponibile per altre funzioni. La toolchain Labview prevede un ambiente separato pe la stesura del codice destinato all'esecuzione su FPGA, con funzionalità dedicate e ridotte per venire in contro all'eseguibilità su hardware, ad esempio è limitato l'impiego dei numeri a virgola mobile e delle divisioni. L'utilizzo del linguaggio grafico Labview rappresenta efficacemente l'esecuzione in parallelo del codice e consente di tradurre automaticamente tramite compilatore il software in codice HDL (Hardware Description Language).

Miracle^2 oltre al processore è dotato di convertitore A/D e D/A con 24 canali differenziali a 16 bit dedicati all'input e 16 canali dedicati all'output, oltre a possedere 16 input e 16 output digitali.

- La frequenza di campionamento sugli analog input è pari a 400 kHz con filtro anti-aliasing hardware posto a 100kHz.

Miracle^2 è completato dall'apparato comunicativo costituito da modulo wifi, porta ethernet, 2 porte CAN, una porta seriale, GPS (su RS232), piattaforma inerziale a 9 assi e microfono. Per ulteriori dettagli di seguito è riportata la scheda tecnica.

Technical data	
Dimensions	105x85x30mm
Weight	400 g
Temperature range	-40°C +85°C
Power supply	6-26 VDC
Power consumption	6 W typ.
Vibration	20-2500Hz 10g sine sweep - 20-2500Hz 6g random profile
Hardware	
Real time processor	667 MHz Dual-Core ARM Cortex A9
FPGA	Artix-7
Storage	512 MB Onboard + 32 GB Flash
RAM	512 MB
I/O Capabilities	
Analog Input	24 channels (2 high voltage channels ±40V), 16bit, ±10V differential input range, up to 400ksps, simultaneous sampling, with antialiasing filter (100kHz) ±0.1% accuracy
Analog Output	16 channels, 16bit, ±10V output range, up to 100ksps, calibrated to ±0.1% accuracy
Digital Input (accept up to 25 V)	8 @ 5V, 500 kHz
Digital Input (protected up to 25 V)	8 @ 5V, 10MHz
Digital Output (protected up to 25 V)	16 @ 5V, 10MHz
Connectivity	
Gigabit Ethernet, WiFi 150N, 2 x Can (1 Mbit/s), RS232, USB, Optional GPS, Motorsport High Density Connectors	
Auxiliary sensors	
Accelerometer, Magnetometer, Gyroscope (9 axis total), audio microphone	

Il software è sviluppabile quindi su tre ambienti distinti che possono comunicare fra di loro: FPGA, Real-Time e Host. Dal primo all'ultimo si allungano i tempi di esecuzione delle routine che vengono eseguite su buffer di dati sempre più grandi ed in modo sempre meno deterministico, ma che possono presentare calcoli sempre più complessi e di alto livello.

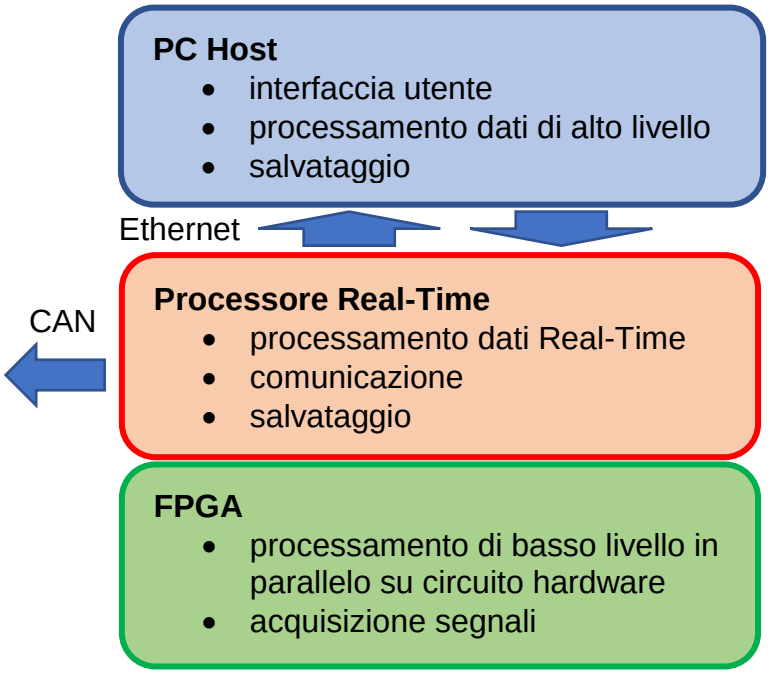


Figura 24: funzionalità host, real-time, FPGA

2.2 **OBI-M2 (ON BOARD INDICATING) E APPLICAZIONI ESISTENTI**

Attualmente la piattaforma Miracle^2 trova la sua principale applicazione per l'analisi di combustione attraverso il software OBI (On Board Indicating), sviluppato per consentire la calibrazione ed il controllo motore on-board o al banco prova. Il sistema è in grado di analizzare fino a 12 cilindri e di comunicare in Real-Time i parametri calcolati via CAN o XCP su ethernet ed è in grado di funzionare sia in modalità Stand-Alone sia sotto il controllo da parte di un'interfaccia eseguita su PC host, attraverso il quale è possibile visualizzare i dati e inviare comandi.

OBI è in grado di acquisire il segnale di giri dalla ruota fonica/sensore ad effetto Hall e fasarsi sui cicli, permettendo la conversione da tempo ad angolo dei campioni e il conseguente calcolo real-time ciclo per ciclo dei parametri di combustione a partire dal segnale di pressione cilindro:

- IMEP, IMEPH
- Pmax,APmax
- Cumulative Heat Release
- MFBxx (xx definito dall'utente, ad esempio MFB10, MFB50, MFB90)
- Indici di knock (MAPO e integrale) calcolati su finestra angolare impostabile

Questi dati possono essere inviati via CAN ciclo per ciclo e ad esempio utilizzati per il controllo in closed-loop della combustione da parte della ECU del motore. Oppure semplicemente possono essere inviati via ethernet in tempo reale per essere monitorati e salvati su PC host. Di fatto il software ed il trattamento dei dati si sviluppa su tre livelli:

- **FPGA:** gestisce l'acquisizione degli ingressi analogici attraverso il convertitore A/D ed il filtraggio da applicare ai fini della visualizzazione e del calcolo (low-pass e high-pass). Inoltre è incaricato del processamento del segnale SMOT (velocità motore) e di fase per effettuare la conversione tempo-angolo e per stabilire la velocità angolare istantanea del motore.

- **Processore Real-Time:** riceve gli indici sintetici ciclo per ciclo e lo streaming degli ingressi analogici dell'FPGA e si occupa del salvataggio e della comunicazione dati via CAN e ethernet verso altri dispositivi come la ECU motore o eventualmente verso PC host. Il salvataggio dei dati può avvenire sulla memoria interna da 32 Gb in modalità stand-alone oppure sul computer host per mezzo dell'invio dei dati via ethernet.

- **PC host:** è dedicato all'interfaccia attraverso la quale l'utente può impostare i parametri di calcolo, gestire i salvataggi e visualizzare i dati grezzi o sintetici ed effettuare alcune valutazioni di tipo statistico sui dati ciclo-ciclo. Per il funzionamento del software OBI non è richiesto necessariamente il collegamento all'host.

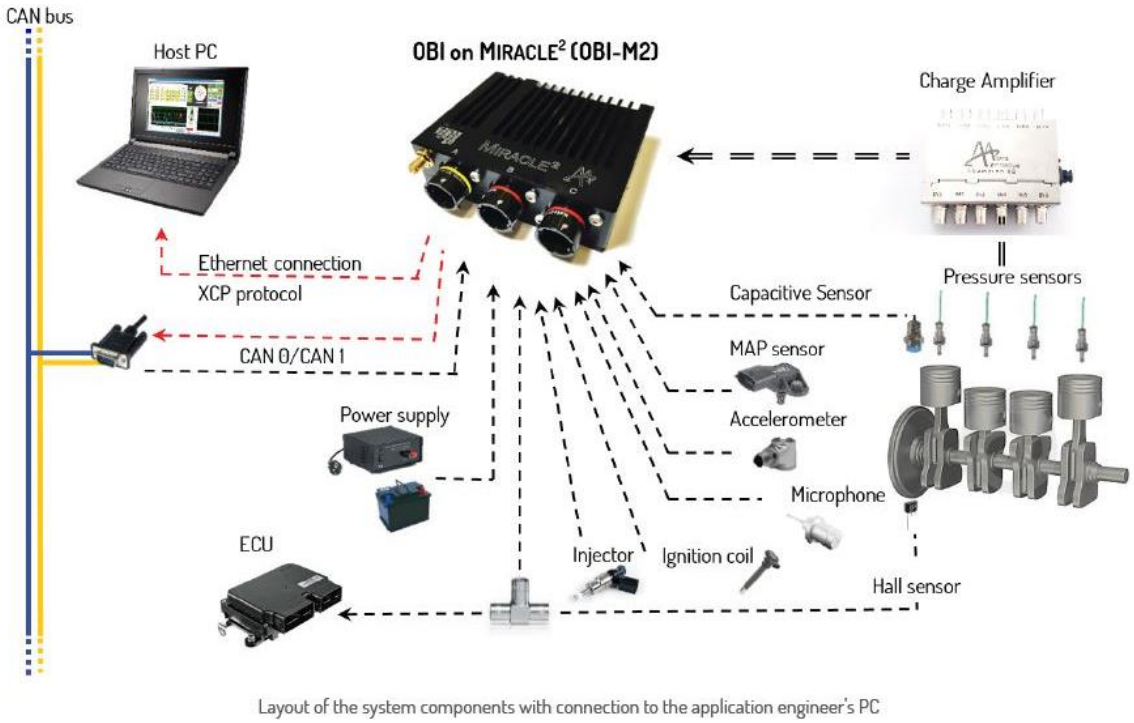


Figura 25: Layout del sistmema OBI su Miracle2 con PC host

La centralina Miracle^2, a dimostrazione della sua versatilità, è stata inoltre utilizzata per rispondere alle seguenti esigenze di applicazione:

- Monitoraggio e diagnosi dello stato degli iniettori tramite accelerometri

- accensione multi-spark per la riduzione inquinanti in un motore GDI ad alte prestazioni
- implementazione di un sistema in grado di controllare un'iniezione multipla a partire da un pattern ad iniezione singola controllato dalla ECU standard
- controllo multi-turbo/electric-turbo
- utilizzo come micro-HIL ("Hardware in the Loop") per il test e lo sviluppo di ECU
- rapid control prototyping ECU in sostituzione dell'ECU di serie

Riferimenti:

[16] [17]

Capitolo 3
SOFTWARE CO-MO

3.1 INTRODUZIONE E STRUTTURA GENERALE DEL SOFTWARE

Il software finalizzato al condition monitoring è stato implementato in linguaggio Labview per poter essere eseguito in modo agevole su Miracle^2, sfruttando l'esperienza aziendale e ed alcune soluzioni realizzate in precedenza. In questo elaborato di tesi, per semplicità e significatività della trattazione non verrà fatta menzione di tutti gli stadi intermedi del codice, sviluppato nell'arco di sei mesi, né verrà mostrato tutto il codice di contorno necessario al funzionamento del sistema, ma solamente la parte strettamente sviluppata per il condition monitoring. Successivamente alla familiarizzazione con l'hardware ed il software di sviluppo, infatti l'applicazione Co-Mo è stata inizialmente costruita da un foglio bianco, implementando mano a mano le funzioni di base, mentre nella versione finale il codice relativo al condition monitoring è stato integrato all'interno del progetto OBI, opportunamente ripulito ed alleggerito per liberare risorse, sfruttandone la maturità e la presenza di funzionalità utili al sistema Co-Mo. Un approccio di questo tipo ha consentito di accorciare i tempi di sviluppo, evitando lunghe sessioni di debug, e predispone il software per lo sviluppo futuro di funzionalità basate sul know-how aziendale. Inoltre, l'utilizzo di OBI come base di partenza ha consentito l'approfondimento delle linee guida di programmazione aziendale e consentirà una più facile leggibilità ed integrazione del codice in caso di future modifiche. È importante però distinguere la differente applicazione dei software OBI e Co-Mo per comprendere le diverse problematiche di realizzazione del codice.

OBI è finalizzato all'analisi di combustione basata su sensori di pressione, ma consente anche il calcolo di indici di detonazione accelerometrici e l'acquisizione di segnali generici analogici. Il software si sviluppa sui 3 ambienti FPGA, real-time, host che svolgono le seguenti funzioni principali:

- **FPGA:**
  - Conversione A/D e gestione fino a 24 canali analogici
  - Fasatura con i cicli motore e acquisizione della velocità di rotazione attraverso ruota fonica / sensore ad effetto Hall / encoder
  - Conversione dal dominio del tempo a quello angolare dei segnali acquisiti
  - Filtraggio passa-basso e calcolo delle grandezze indicating di ciclo
  - Filtraggio passa-alto e calcolo degli indici di detonazione del ciclo
  - Invio dei dati sintetici e grezzi al real-time attraverso protocollo di comunicazione specifico

- **Real-time:**
  - Impostazione ed invio dei parametri di calcolo all'FPGA
  - Ricezione dei dati sintetici e grezzi dall'FPGA
  - Comunicazione via CAN ciclo per ciclo dei dati sintetici
  - Gestione generale della comunicazione e delle connettività
  - Salvataggio dei dati grezzi e sintetici su scheda SD interna
  - Invio dei dati al PC host

- **Host:**
  - Spacchettamento dei dati inviati dal real-time
  - Interfaccia utente per la visualizzazione dati
  - Interfaccia utente per l'impostazione dei parametri di calcolo
  - Salvataggio su disco

Ne risulta che il processore real-time non è gravato da calcoli onerosi per il processamento dati, né è impegnato nella gestione dell'interfaccia e del post processamento.

Il software Co-Mo per semplicità al momento si sviluppa solamente su FPGA e real-time, e sebbene sia basato sulla struttura del codice OBI, al momento non ne sfrutta tutte le funzionalità. Al momento le funzioni implementate sono:

- **FPGA:**
  - Conversione A/D e gestione fino a 12 canali analogici
  - Fasatura con i cicli motore e acquisizione della velocità di rotazione attraverso ruota fonica / sensore ad effetto Hall / encoder
  - Conversione dal dominio del tempo a quello angolare dei segnali acquisiti
  - Filtro anti-aliasing e sotto-campionamento ad 80 kHz
  - Filtraggio passa-alto e calcolo degli indici di detonazione del ciclo
  - Filtro impostabile per l'invio al real-time
  - Invio dei dati sintetici e grezzi al real-time attraverso protocollo di comunicazione specifico

- **Real-time:**
  - Impostazione ed invio dei parametri di calcolo all'FPGA
  - Ricezione dei dati sintetici dall'FPGA
  - Comunicazione via CAN ciclo per ciclo dei dati sintetici
  - **Spacchettamento dati grezzi provenienti dall'FPGA**
  - **Analisi spettrale fino a 6ch**
  - **Calcolo del trendindex, RMS, Crest Factor, Peak Value**
  - **Costruzione e salvataggio della mappa del comportamento vibrazionale del motore**
  - Gestione generale della comunicazione e delle connettività
  - Salvataggio dei dati grezzi, degli indici sintetici e dell'analisi spettrale su scheda SD interna **in formato TDMS**
  - Interfaccia utente
  - Invio dei dati al PC host

Nell'attuale versione del codice il processore real-time è quindi gravato da una notevole mole di operazioni dall'elevato costo computazionale, in primis lo spacchettamento ed il processamento dei buffer di dati provenienti dall'FPGA ed in secondo luogo la costruzione della mappa oltre che la gestione dell'interfaccia utente. Le funzionalità di comunicazione e calcolo degli indici di detonazione sono

sostanzialmente rimaste immutate rispetto al software OBI, ma per il loro corretto funzionamento sono necessarie correzioni nel codice che gestisce i protocolli interni del flusso dei dati. Questa parte non è stata oggetto della tesi, ma è potenzialmente funzionante.

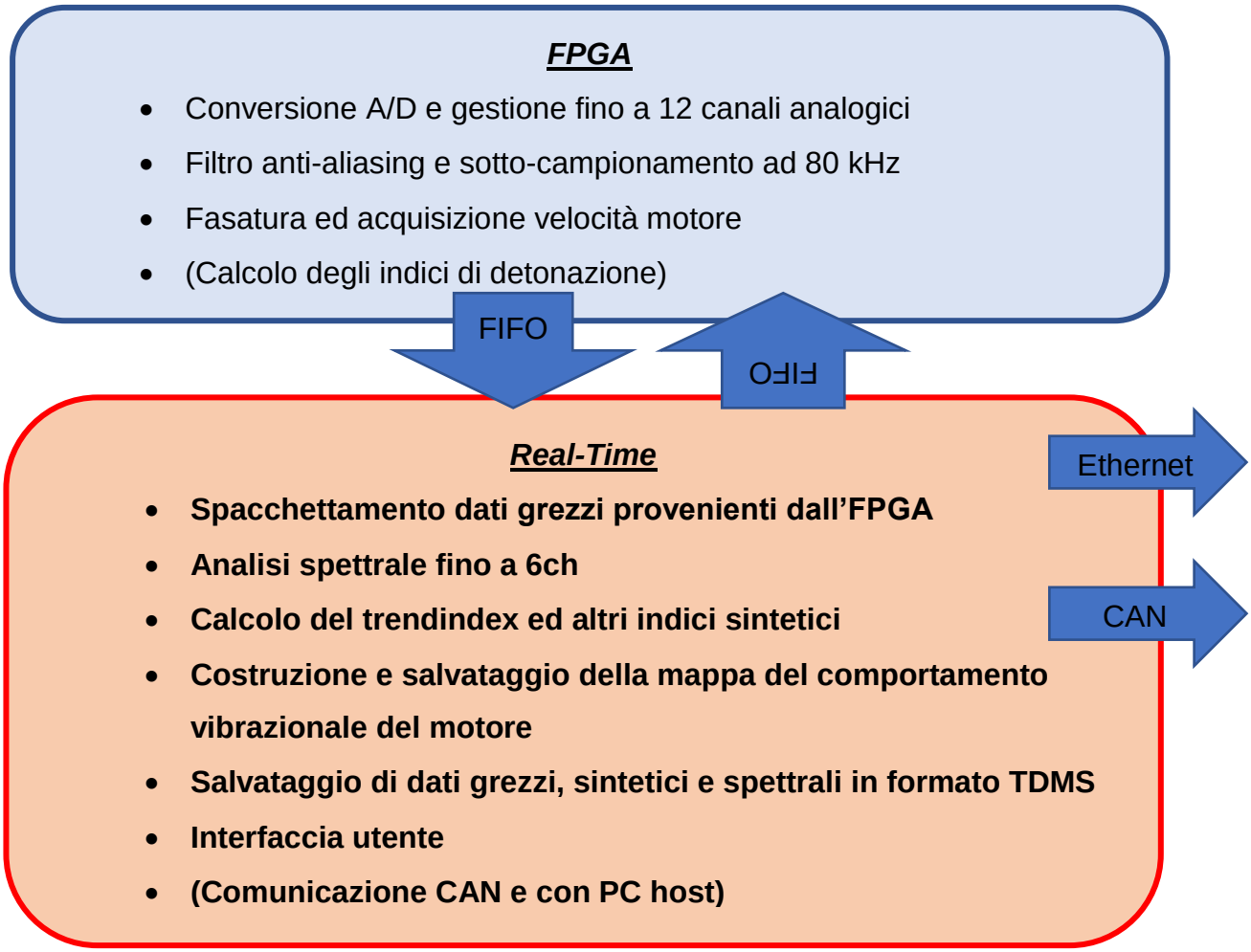


Figura 26: schema funzionale real-time / FPGA

3.2 **FPGA**

La parte di codice FPGA è stata alleggerita dal calcolo degli indici indicating come IMEP e ROHR per far spazio all'implementazione di nuove parti, ma al contempo è stata mantenuta la struttura di base del flusso dati che prevede 2 percorsi principali:

- Invio dei dati campionati al real-time
- Invio dei dati campionati e dell'angolo associato per il calcolo degli indici sintetici

Visto l'utilizzo di accelerometri come sensori principali del sistema e l'implementazione già realizzata è stata mantenuta tutta la parte relativa al calcolo degli indici MAPO e integrale per la rilevazione della detonazione. È stata altresì mantenuta tutta la struttura che tramite FIFO gestisce il passaggio dei parametri fra real-time e FPGA. Infatti la maggior parte delle impostazioni non vengono comunicate attraverso variabili poste sul front-panel dell'FPGA, ma attraverso delle FIFO in cui i dati vengono codificati assieme a dei numeri identificativi ed in seguito spacchettati. Questo approccio consente di risparmiare risorse.

Le principali modifiche codice sono state effettuate nel loop principale che gestisce l'acquisizione, il filtraggio e l'invio dei dati. Tutta l'implementazione è stata condizionata dalla necessità di rientrare entro il tempo di ciclo previsto.

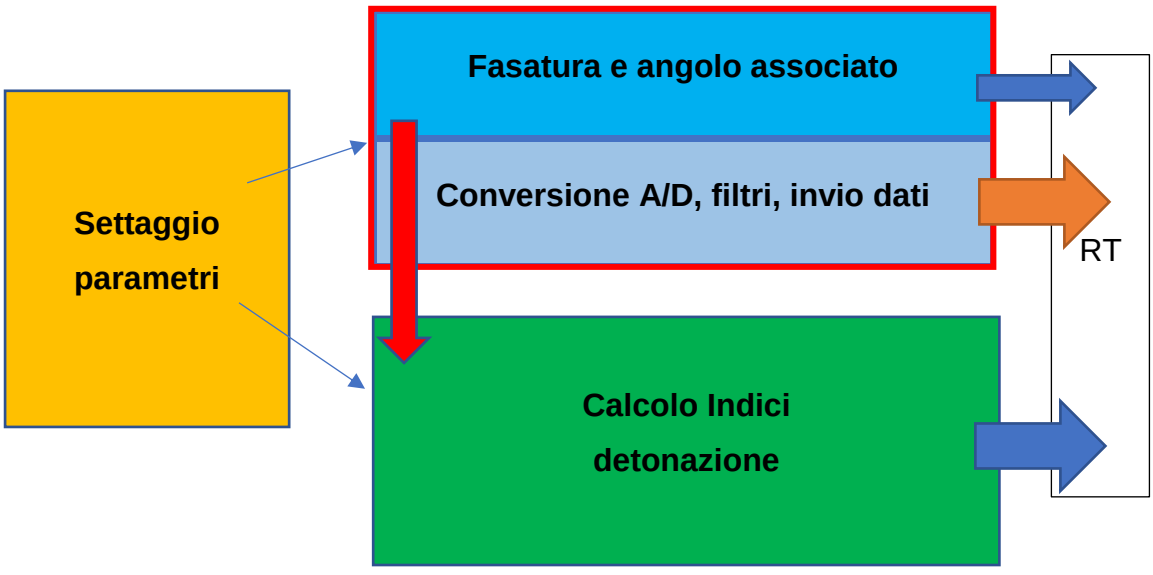


Figura 27: schema funzionale FPGA

In particolare, è stato ridotto il numero dei canali disponibili all'invio da 24 a 12 per diminuire il numero di ticks necessari all'esecuzione del codice: infatti, in seguito all'introduzione dello stadio aggiuntivo di filtraggio e sotto-campionamento, non era più possibile rimanere entro i 100 ticks per l'esecuzione del loop, che con il clock di 40MHz e la frequenza di campionamento, e quindi di loop, fissata a 400KHz corrispondono a 25 µs. È stato inoltre necessario introdurre diversi feedback node aggiuntivi lungo le pipeline per spezzarne l'esecuzione su più cicli e consentire di rimanere entro i 100 ticks. Va ricordato che il convertitore A/D è a 16 bit e che quindi i dati trattati sono i16.

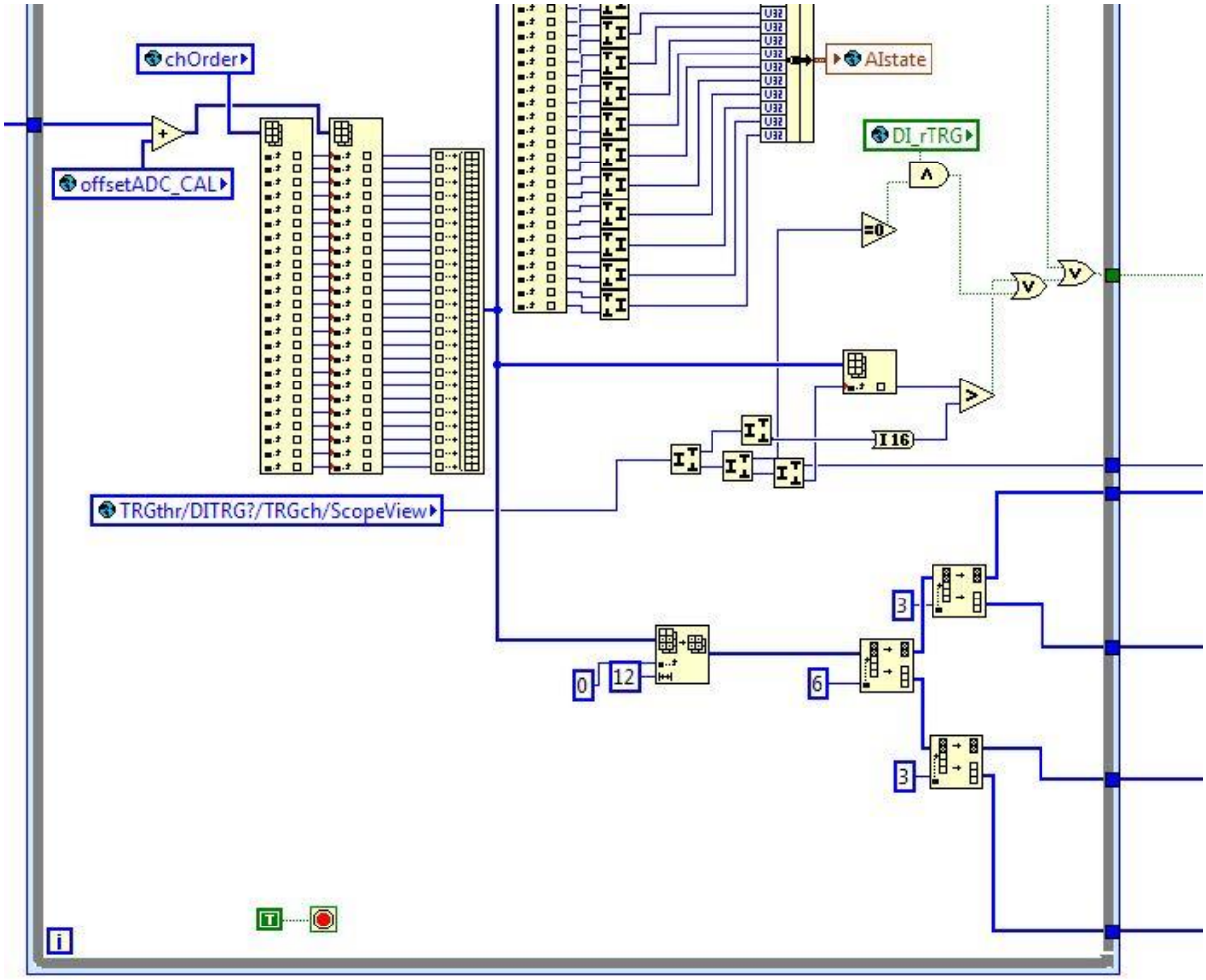


Figura 28: selezione di solo 12ch e successiva separazione in 4 gruppi da 3 canali

L'introduzione del sotto-campionamento e del relativo filtro anti-aliasing software software, nasce dall'esigenza di ridurre la mole di dati inviati e poi processati in real-time, anche perché sarebbe inutile avere dei dati a 400 kHz quando il contenuto informativo oltre le poche decine di kHz è praticamente nullo. La riduzione dei campioni

permette anche di limitare la memoria occupata dal salvataggio. Per consentire l'esecuzione nei tempi di loop il filtro anti-aliasing ed il ri-campionamento è stato implementato attraverso 4 blocchi "rational-resampler" che vengono eseguiti in parallelo, ognuno per 3 canali. Il ri-campionamento è di 80 kHz ed il filtro è impostato fisso per produrre un'attenuazione fuori banda di almeno 30 dB con una frequenza di taglio (-3dB) di 37.6 kHz.

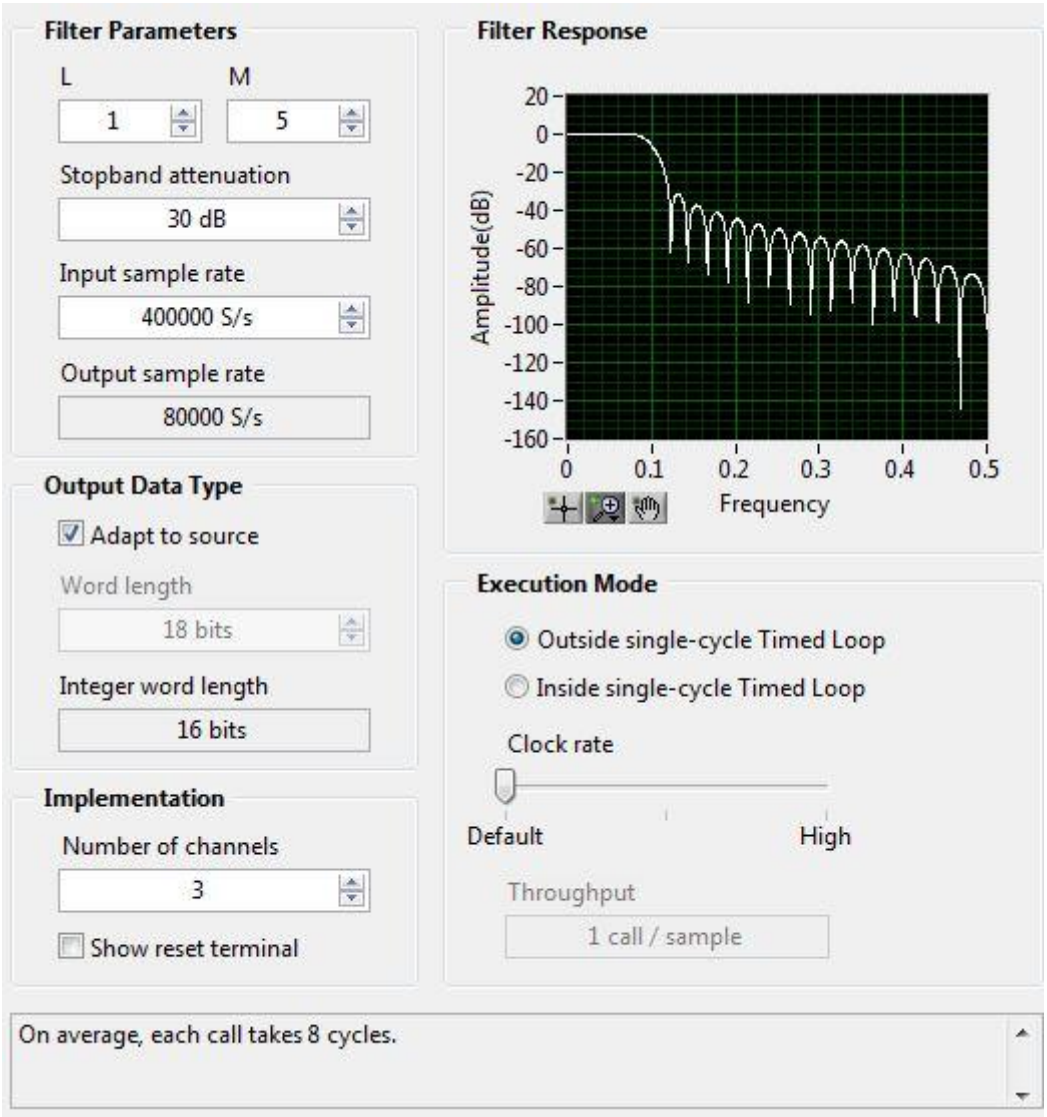


Figura 29: rational resampler settings

Passare da 400 a 80 kHz significa ridurre i campioni di un fattore 1/5, quindi il valore booleano posto in uscita dal for loop contenente i blocchi rational resampler diviene vero 1 volta ogni 5 cicli, permettendo l'invio in FIFO dei dati all'interno della case structure successiva.

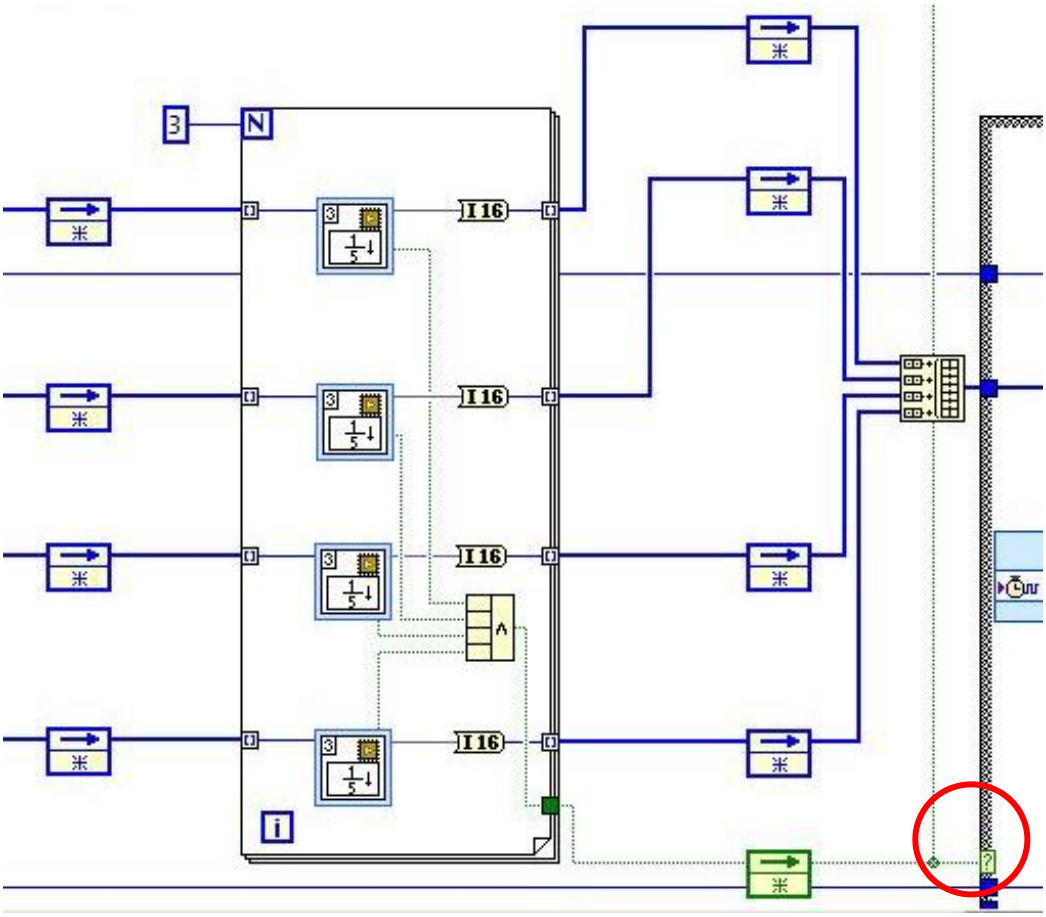


Figura 30: for loop che contiene i blocchi rational resampler per l'esecuzione del filtraggio e del booleano di ricampionamento che consente l'invio di 1 dato ogni 5 loop

La case structure successiva, comandata dal booleano, contiene il filtro impostabile, la costruzione del vettore di dati da inviare al real-time tramite FIFO ed il blocco di memoria utilizzato per passare i dati filtrati alla sezione di codice che calcola gli indici. Infatti, a valle del filtro configurabile dall'utente, il flusso dei dati si divide per essere inviato rispettivamente:

- Al calcolo degli indici attraverso il blocco di memoria P_LP_HP, che per ogni campione invia il dato filtrato passa alto e passa basso contemporaneamente come u32. Per il calcolo dei soli indici di detonazione verrà utilizzata solo la parte HP.
- Al real-time attraverso la **FIFO "Signal Processing"** sotto forma di i16.
- Al real-time attraverso un subVI che gestisce il protocollo dati per la visualizzazione ciclo per ciclo su host o per il salvataggio (funzionalità di OBI).

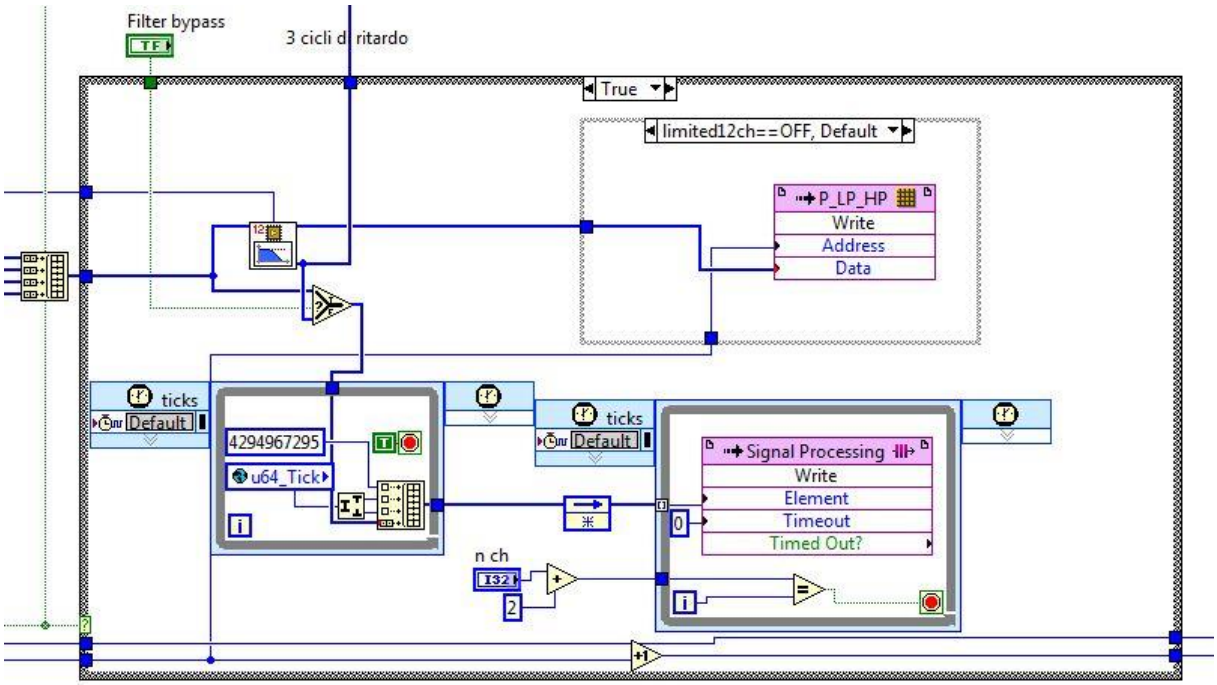


Figura 31: Case structure comandata dal booleano

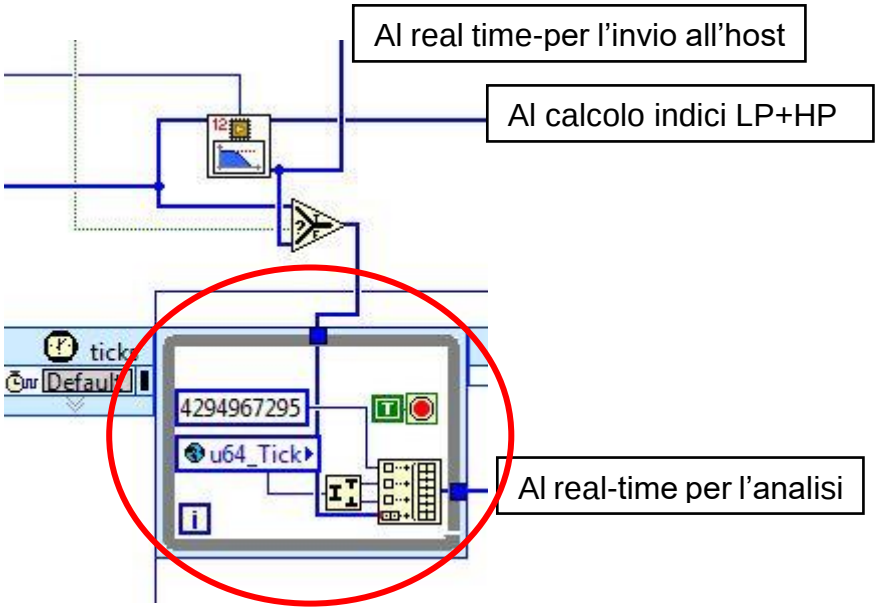


Figura 32: filtro settabile 12 ch e costruzione del vettore da inviare al real-time

La parte cerchiata in rosso compresa all'interno del single-cycle timed loop è quella che riguarda il trattamento dei dati finalizzato all'analisi spettrale. Sebbene su OBI fosse già presente una parte di codice dedicata all'invio dei dati al real-time, questo, a causa del complesso protocollo interno di comunicazione, non si prestava allo spacchettamento dei dati ed al successivo calcolo real-time. È stata quindi aggiunta la FIFO "Signal Processing", preceduta dall'impacchettamento dei dati secondo un

protocollo più semplice, che consentisse un utilizzo più immediato dei dati su base tempo in real-time. Il vettore inviato attraverso la FIFO, composta da elementi u32, è così assemblato ad ogni loop:

- Il numero 4294967295, corrispondente al fondo scala degli unsigned a 32 bit, utilizzato come **identifier** per riconoscere da real-time l'inizio di un nuovo set di campioni.
- Il tempo corrispondente ai campioni inviati, convertito da u64 in due u32
- Il vettore dei campioni corrispondenti ad ogni singolo canale convertiti da i16 ad u32

Con questo protocollo nel caso dell'invio di tutti e 12 i canali, si ritroveranno 15 nuovi elementi all'interno della FIFO ad ogni loop, che contengono le informazioni di 1 campione per canale ed il corrispondente tempo di acquisizione.

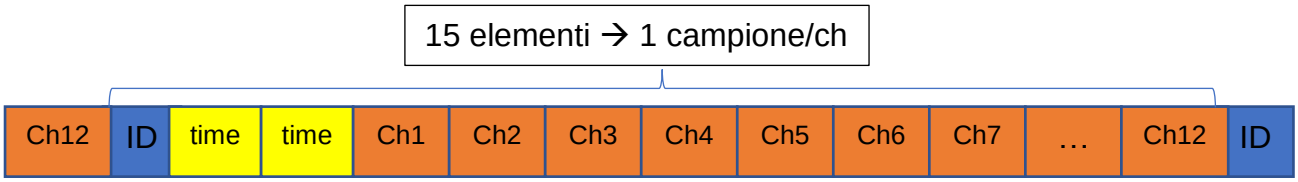


Figura 33: vettore conente i campioni inviato dall'FPGA al real-time

La FIFO viene riempita ad ogni loop con **n°canali+3 elementi,** che verranno poi identificati e spacchettati canale per canale ad intervalli regolari nel codice real-time per poi essere analizzati. L'identifier non può essere confuso con gli altri elementi. La compilazione di questa versione dell'FPGA mostra che ci sono ancora abbastanza risorse sfruttabili per eventuali modifiche future.

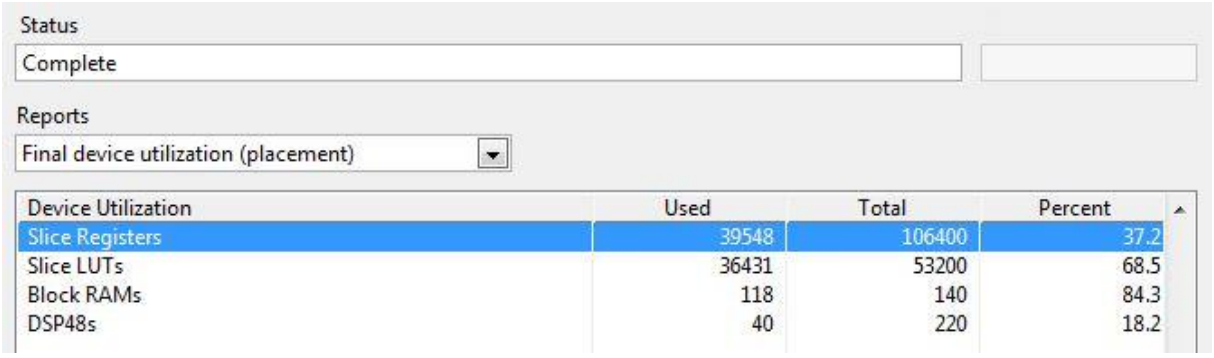


Figura 34: stato compilazione FPGA

3.3 REAL-TIME

Come spiegato in precedenza anche la parte di software real-time si appoggia sul progetto OBI, ma differentemente da quanto accaduto con l'FPGA la parte di codice dedicata al condition monitoring è quasi interamente separata da quello che è il codice preesistente. È stata infatti mantenuta tutta la parte di comunicazione via UDP e TCP per l'host e la comunicazione via CAN degl'indici sintetici, mentre sono state disabilitate alcune parti di codice come ad esempio il precedente metodo di salvataggio in binario con codifica interna. È importante precisare che sebbene in futuro il codice andrà ripulito e reso più specifico per l'applicazione Co-Mo, è stato scelto di integrare e sfruttare il più possibile i controlli e la struttura di configurazione di OBI. Rispetto alla stesura di tutto il software da zero questo ha comportato un maggiore impegno iniziale per la comprensione e l'adattamento del codice, ma consente di disporre di una base di partenza già collaudata, debuggata e integrabile nelle funzionalità. La configurazione dei parametri per l'FPGA e per tutto il funzionamento del software è di fatto lo stesso cluster di controlli di OBI con le opportune aggiunte e modifiche nei VI che operano l'inizializzazione, l'aggiornamento ed il passaggio delle variabili di configurazione. Con l'intento di rendere il più efficace possibile l'integrazione con il software precedente si è cercato il più possibile di utilizzare le variabili già esistenti anche nelle parti di codice nuovo, come ad esempio per la gestione dei canali in termini di ordinamento, nomi, guadagni e offset e per la configurazione dei filtri in FPGA.

La parte di codice dedicata al condition monitoring e sviluppata per l'ambiente real-time è quella più corposa ed articolata e la configurazione attuale è stata raggiunta per tentativi con l'obiettivo di ridurre il più possibile il carico computazionale del processore. La struttura è composta da 5 timed loop principali che assolvono le seguenti funzioni:

- Ricezione e spacchettamento dei dati provenienti dall'FPGA
- Analisi spettrale e calcolo degli indici
- Salvataggio
- Costruzione della mappa di vibrazione
- Visualizzazione dati

L’impostazione e l’aggiornamento dei principali parametri avviene invece in un altro loop, più lento dei precedenti, tramite un cluster di controlli a front panel che, in caso di modifica, scrive i valori desiderati su un insieme di variabili globali. Laddove non era necessario avere un controllo o un indicatore disponibile per l’utente o nel caso in cui non fosse necessario l’aggiornamento frequente ed improvviso da utente, sono state utilizzate delle variabili globali, che occupano meno risorse e consentono la scrittura e la lettura immediata.

L’utilizzo di timed loop separati consente di assegnare le risorse nel modo più efficiente e di gestire processi differenti su core del processore differenti (multi-threading). Infatti per ogni timed loop è possibile assegnare:

- core del processore (0 o 1 nel caso di Miracle)
- tempo di loop
- priorità del loop

La corretta gestione di questi parametri è fondamentale per l’esecuzione del codice in modo deterministico e quindi per l’appunto in real-time.

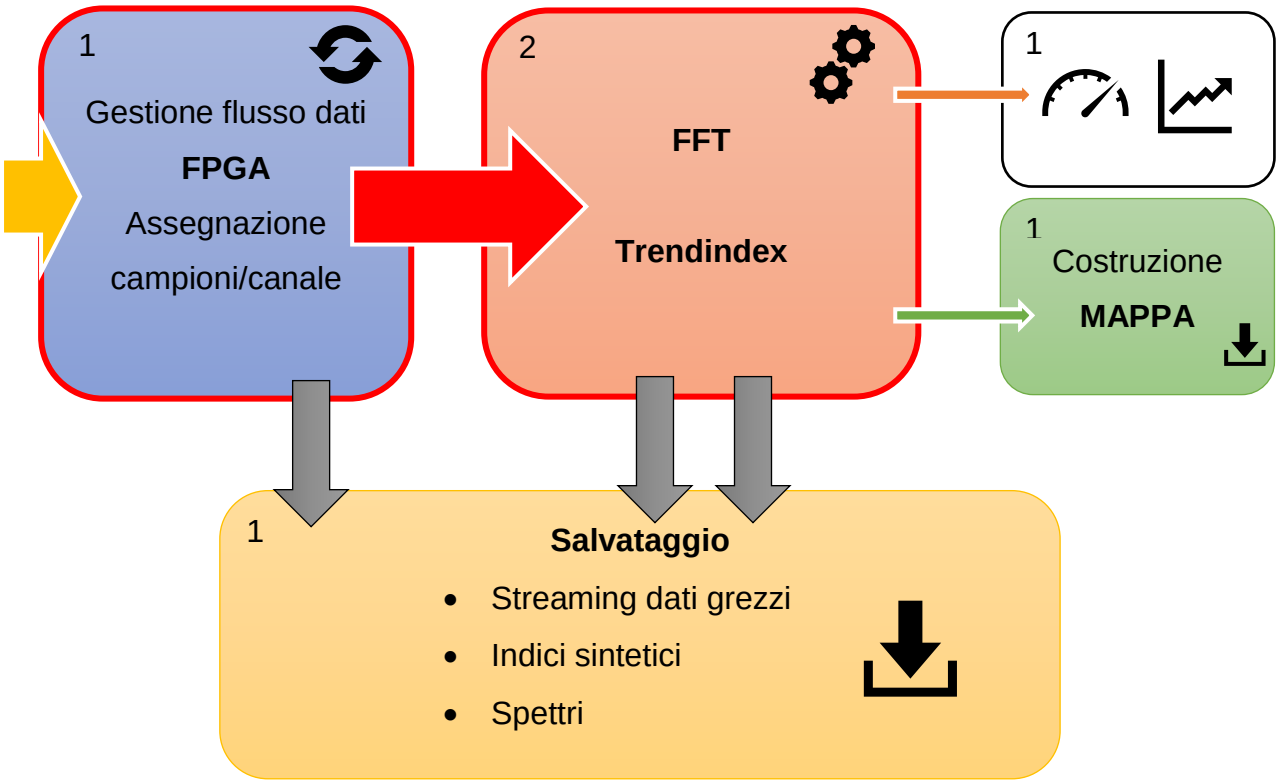


Figura 35: schema funzionale real-time

I timed loop che gestiscono il flusso dati dall'FPGA e l'FFT, contornati in rosso nello schema precedente, sono le strutture a maggior priorità da cui dipende la riuscita dell'analisi in tempo reale. Il numero presente in alto a sinistra indica il core del processore impiegato ed infatti per questi due loop sono stati scelti in modo separato per consentire l'esecuzione in parallelo dei processi. Questi due loop si scambiano dati continuamente e devono eseguire il codice con lo stesso tempo di ciclo, selezionabile dall'utente. Di seguito verrà sempre riportato come esempio un tempo di ciclo pari a 50ms.

Il loop che gestisce il flusso di dati dall'FPGA svuota la FIFO riempita come visto nel capitolo precedente e assegna i singoli campioni ai rispettivi canali. Come si vedrà in seguito è importante che nessun dato vada perso e che il numero di campioni resi disponibili per l'analisi sia fissato, nonostante l'inevitabile jitter sul tempo di ciclo e l'eventuale presenza di problemi. Ad ogni ciclo, il numero di campioni corrispondenti a 50ms verrà inviato attraverso una coda al loop di analisi, che a sua volta dovrà essere eseguito entro 50ms per evitare accumuli nella coda e permettere l'analisi in tempo reale dei dati. È fondamentale quindi che questi due loop siano eseguiti entro il tempo imposto di 50ms, altrimenti si generano accumuli sulla FIFO proveniente dall'FPGA o sulla coda diretta al loop d'analisi, compromettendo il funzionamento del sistema.

Gli accorgimenti e le soluzioni implementate per il corretto funzionamento del sistema verranno presentati nei successivi paragrafi ed hanno rappresentato gran parte del tempo di sviluppo del sistema, che deve essere in grado di funzionare per lungo tempo senza perdere dati durante il salvataggio e presentando i risultati dell'analisi spettrale con la massima tempestività. In ogni caso, trascurando eventuali accumuli, il tempo minimo fra l'acquisizione e la presentazione del trendindex è pari alla somma dei tempi dei due loop, cioè 50ms x 2 = 100 ms.

3.3.1 ***FIFO loop***

In questa sezione verrà presentato il codice relativo al loop che i occupa della ricezione e dell'organizzazione dei dati provenienti dall'FPGA ed essendo il primo step di calcolo che i dati incontrano sul real-time è fondamentale per la corretta esecuzione di tutto il codice a valle.

La prima operazione effettuata è lo svuotamento degli elementi dalla FIFO proveniente dall'FPGA. Il numero teorico degli elementi attesi è pari a:

$$T\_loop * (n° \; canali + 3) * Fs$$

Nel caso di un T_loop = 50 ms, 12 canali ed una Fs = 80 kHz si ottengono 60000 campioni per ogni ciclo. In realtà l'oscillazione per quanto limitata del tempo di loop (jitter) causa la presenza in FIFO di un numero non sempre uguale di campioni. Tralasciando l'accumulo iniziale dovuto all'avvio non sincrono dell'FPGA e del real-time, il numero di campioni resi disponibili nella FIFO oscillerà attorno ai 60000. Per evitare di creare accumuli nella fifo, e quindi un ritardo, e per impedire che non vi siano sufficienti campioni disponibili è stato creato a valle un buffer di compensazione del jitter. I dati nel buffer sono i più recenti del ciclo precedente, verrà mostrato in seguito come si ottiene, e vanno quindi aggiunti all'inizio del vettore di dati del ciclo attuale.

Va ricordato che l'obiettivo di questa parte di codice è la gestione dello streaming di dati **senza alcuna perdita di campioni** e **l'assegnazione dei campioni ai rispettivi canali in numero fisso** pari a T_loop*Fs, in modo da avere in uscita dall'FFT uno spettro, confrontabile con sé stesso, composto sempre dallo stesso preciso numero di elementi.

Con l'attuale configurazione non è sufficiente la presenza del buffer per garantire il corretto funzionamento, può infatti avvenire che il buffer si svuoti e non sia disponibile al ciclo successivo il numero minimo di campioni. All'interno del riquadro verde quindi sono state introdotte le seguenti condizioni:

- In condizioni normali ad ogni ciclo la FIFO viene svuotata interamente dei suoi elementi, che oscilleranno attorno a **Samples Exp.**

- Se il numero di elementi rimanenti nella FIFO è minore di **Samples Exp** (pari a $T\_loop * (n° \; canali + 3) * Fs$) e contemporaneamente **size buffer cut** (numero di elementi del buffer) è minore di **n+3** (numero minimo di elementi per essere certi di avere un identifier) allora il numero di elementi di cui svuotare la FIFO è

imposto a **Samples Exp** ed è concesso un timeout massimo di 5 ms per aspettare il raggiungimento del numero di elementi necessario.

- Se **size buffer cut** rimane maggiore di **Samples Exp** per due cicli di seguito, il buffer viene azzerato. Il verificarsi di questo evento significa che qualcosa non sta funzionando nel processamento dei dati e comporta la perdita dei campioni nel buffer, ma previene l'overflow dei dati.

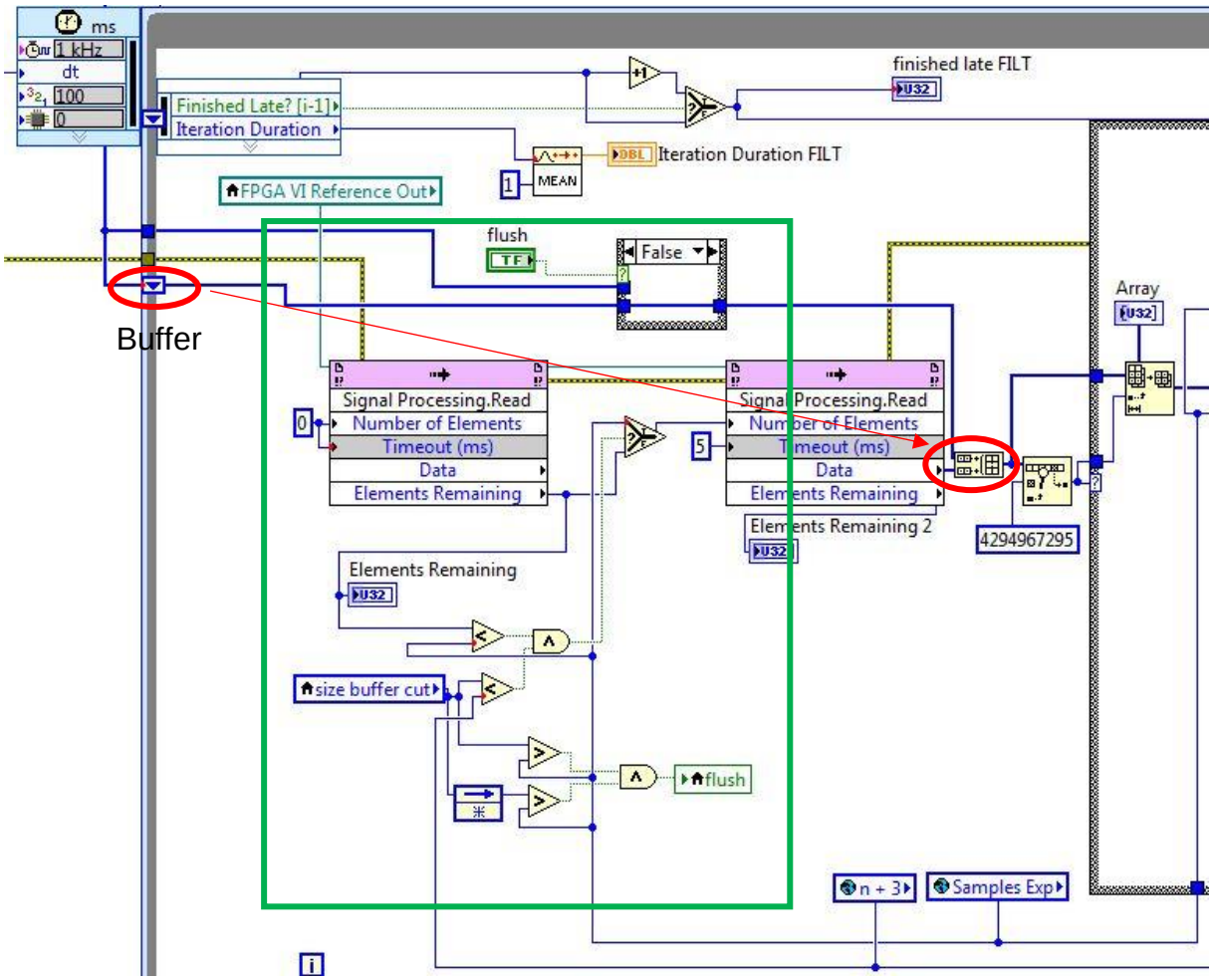


Figura 36: svuotamento FIFO, logiche di svuotamento e buffer.

Le condizioni precedenti impongono che il numero di elementi in ingresso alla case structure che gestisce lo spacchettamento dei dati, pari a quelli provenienti dalla FIFO più il buffer, consenta sempre di ottenere **Fs * T_loop** campioni per ogni canale a valle delle operazioni necessarie alla rimozione dell'identifier.

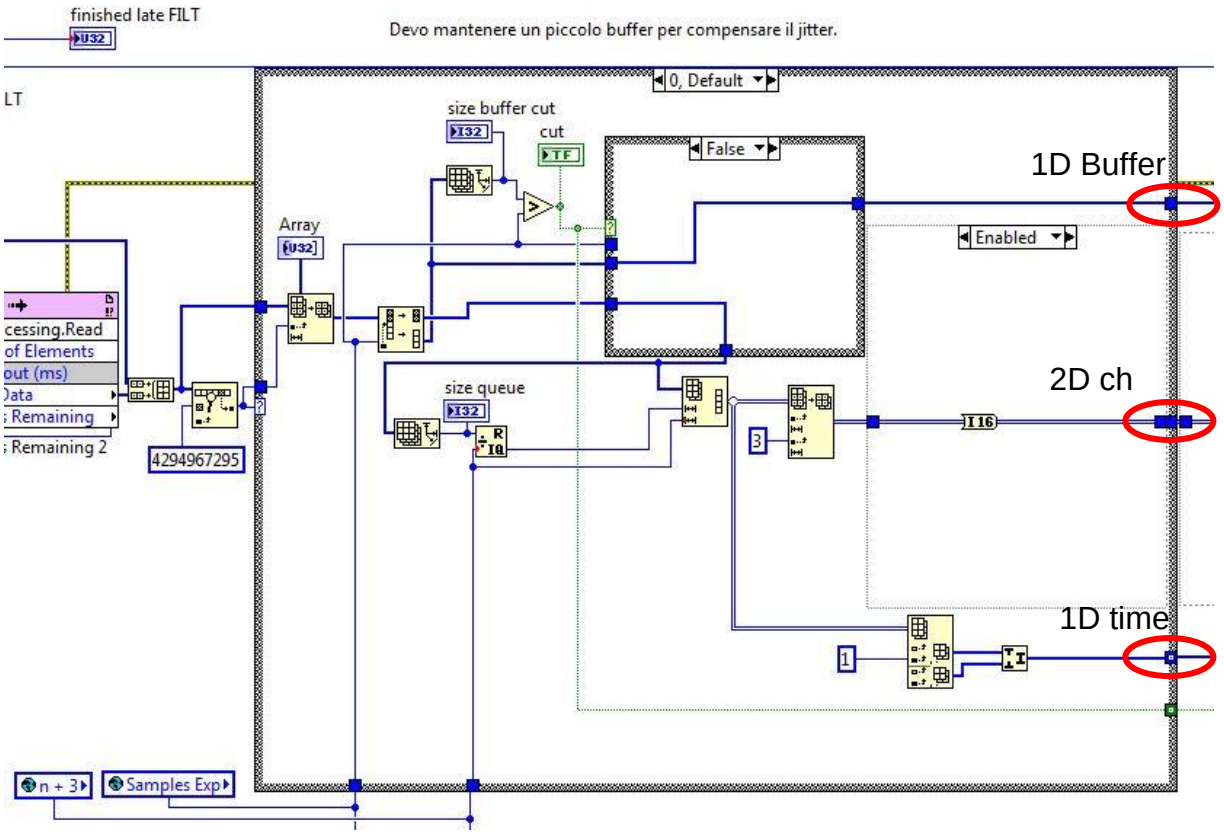


Figura 37: codice dedicato alla costruzione dell'array 2D dei canali

La case structure esegue il codice mostrato in figura 32 solamente se nel vettore in ingresso viene trovato un identifier. Questa condizione potrebbe non verificarsi al primo ciclo in caso di ricezione di un numero di elementi minore di n+3 dovuti all'avvio del codice non sincrono su FPGA e real-time. Il codice in figura consente di passare dal vettore 1D costruito in ingresso alla FIFO nell'FPGA ad un array 2D costituito dai campioni ordinati per canale ed un vettore 1D dei tempi corrispondenti ai campioni.

Figura 38: spacchettamento dati inviati dall'FPGA su real-time

I passaggi eseguiti per raggiungere questo risultato sono:

1. Troncamento dell'array in ingresso in corrispondenza del primo identifier (avviene solamente al primo ciclo perché successivamente il primo elemento è sempre un identifier).
2. Troncamento dell'array così ottenuto per ottenere un numero di elementi pari a **Samples Exp.** Gli elementi presi sono quelli meno recenti.
3. Reshape del vettore 1D del passo precedente in un array 2D contenente nelle prime colonne il tempo scomposto in 2 u32 e nelle successive i campioni corrispondenti ai canali.
4. Riconversione dei canali da u32 al formato originale i16
5. Ricomposizione del tempo in u64.

La parte del vettore troncata al punto 2 va a costituire il buffer, che in caso di superamento del numero di elementi pari a **Samples Exp** viene a sua volta troncato ed aggiunto al vettore in ingresso al punto 3. Così facendo, in caso di allungamento eccessivo del buffer (che significa avere un ritardo sui campioni analizzati), viene in sostanza inviato un vettore doppio di dati.

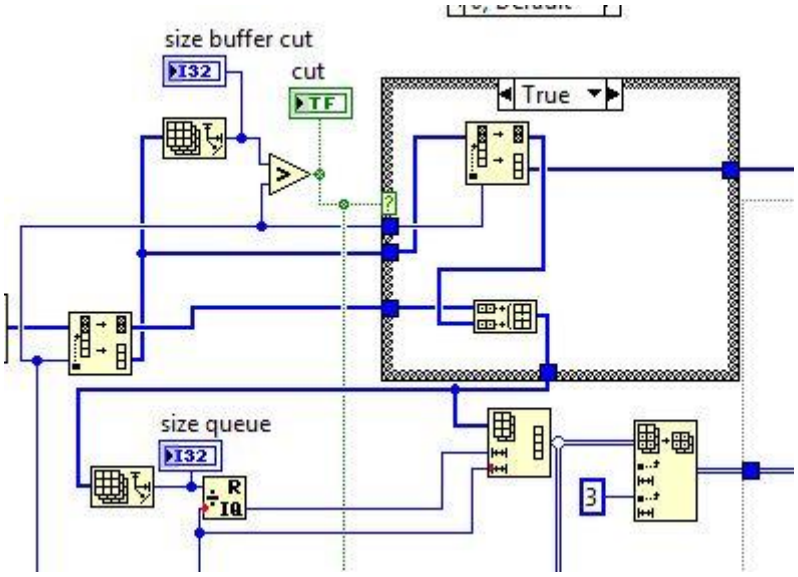
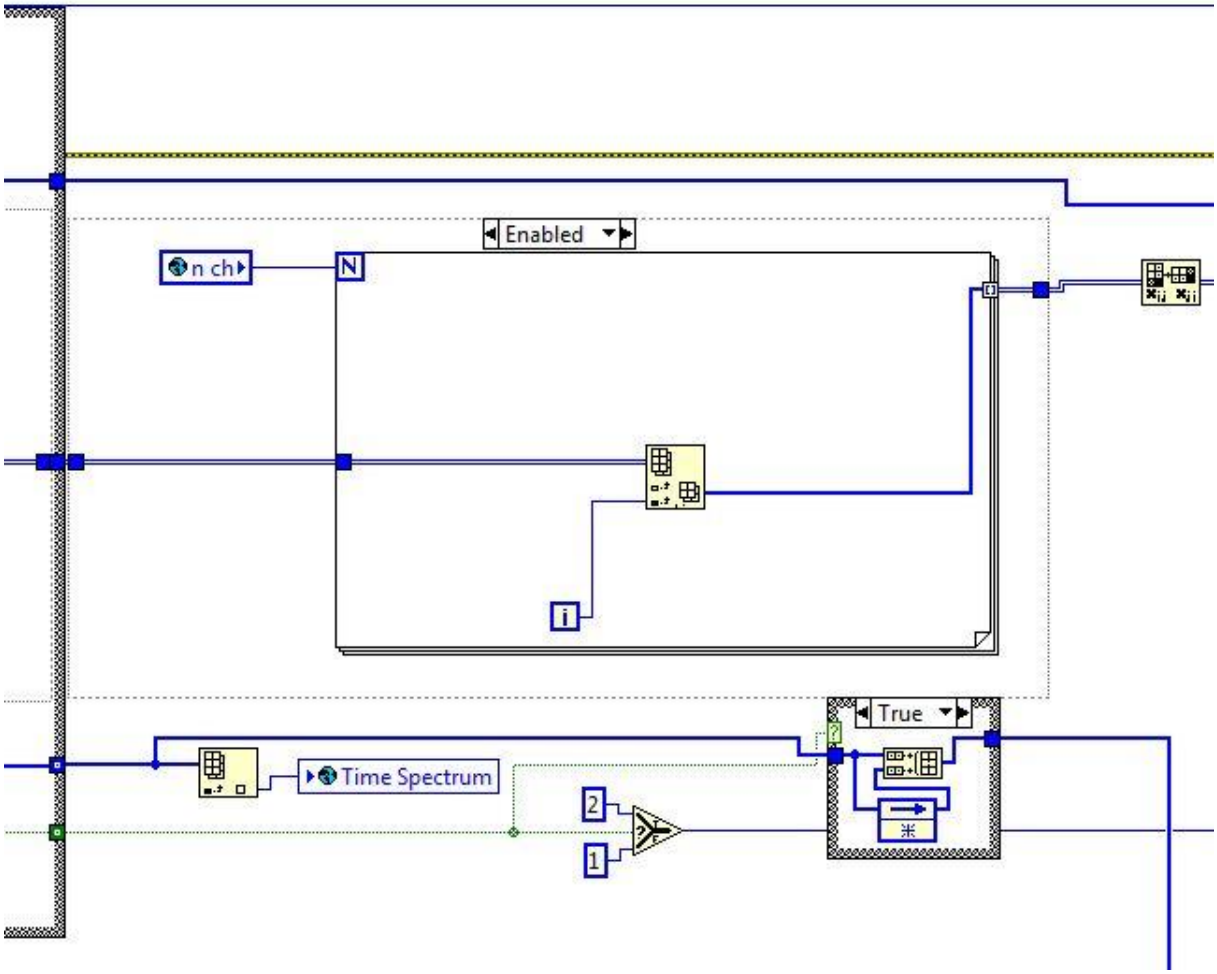


Figura 39: troncamento del buffer ed invio di un numero di elementi pari a 2*Samples Exp



È possibile selezionare un numero inferiore di canali rispetto a 12, inoltre in caso di troncamento ed aggiunta del buffer ai dati da analizzare anche il vettore tempo viene adeguato.

Successivamente avviene l'invio dei dati attraverso code alle 2 diverse destinazioni previste: il loop dell'FFT ed il loop di salvataggio.

Per quanto riguarda l'invio dei dati all'analisi vengono presi e mandati in coda sempre solamente gli ultimi campioni, in modo da analizzare sempre quelli più recenti anche in caso di doppio numero di elementi. In coda quindi si avranno sempre dei pacchetti di dati contenenti un array 2D di **n ch** colonne e **Samples ch** (Fs*T_loop) righe. La coda viene riempita solamente quando il booleano **Spectral Analysis** è posto a true.

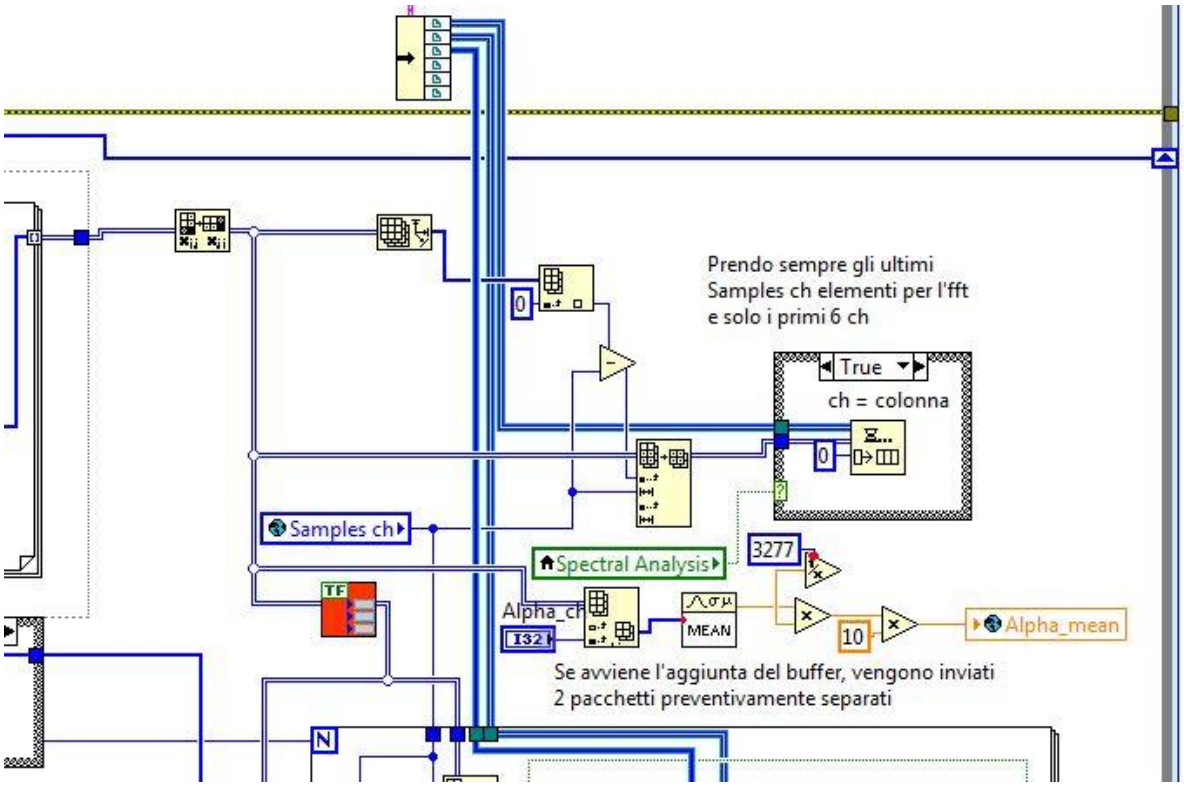


Figura 40: invio dei dati al loop di analisi attraverso un la coda. Costruzione del valore della farfalla sul tempo di loop

Invece alla coda di salvataggio vengono inviati tutti i dati ed in caso di taglio e aggiunta del buffer, quindi di un ciclo “doppio”, vengono spedititi due pacchetti distinti in coda, per permettere il conteggio degli elementi da salvare. È presente anche una coda di salvataggio separata per il tempo, che è rappresentato come u64 invece che come i16.

Nel codice sono implementate anche le seguenti funzionalità:

- Costruzione del valore medio sul tempo di ciclo della farfalla (o del valore rappresentativo del carico) come percentuale.
- Selezione dei canali da inviare al salvataggio
- Controllo preciso degli elementi inviati in coda, e quindi del tempo di salvataggio, attraverso le variabili **Loops to save**, **Save queue** ed un counter.
- Attivazione della registrazione attraverso un trigger selezionabile nel valore sui canali.

La gestione della coda di salvataggio avviene sotto il controllo del Save loop, e quindi verrà trattato più in dettaglio in seguito.

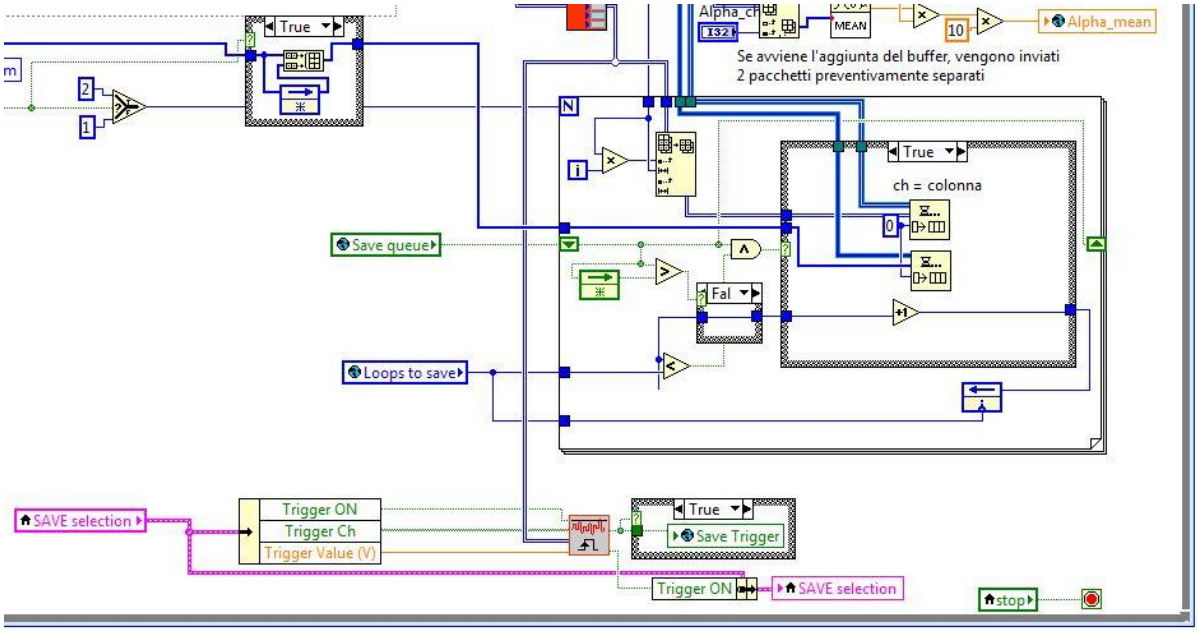


Figura 41: coda di salvataggio

Come accennato in precedenza all'interno di questo loop vengono anche gestiti i parametri che identificano la condizione media di funzionamento del motore sul set di dati da inviare all'analisi. Allo stato attuale del software vengono presi in considerazione solamente **Alpha_mean**, rappresentativo del carico motore, e **RPM_mean**, corrispondente al numero di giri medio.

Alpha_mean

Con la premessa di aver acquisito su un canale il valore in Volt corrispondete al carico motore (da un sensore TPS ad esempio) è possibile convertire il valore in bit proveniente dal convertitore A/D in una percentuale di carico e scriverlo su una variabile globale.

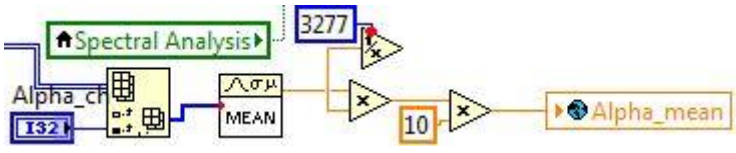


Figura 42: Calcolo Alpha_mean

RPM_mean

Dall'FPGA viene inviato su real-time il regime di rotazione ciclo per ciclo insieme agli indici sintetici così calcolati come su OBI. Sfruttando questa informazione è possibile assegnare alla variabile globale RPM_mean il regime di rotazione medio corrispondente al set di dati inviato all'analisi. All'interno del FIFO loop è posta la seguente variabile booleana che comanda il reset del calcolo degli RPM medi ad ogni nuovo ciclo:

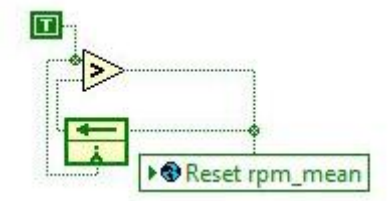


Figura 43: Reset rpm_mean

Mentre nel loop che gestisce i dati ciclo-ciclo, originario di OBI, viene sovrascritta su comando il nuovo valore di **RPM_mean.**

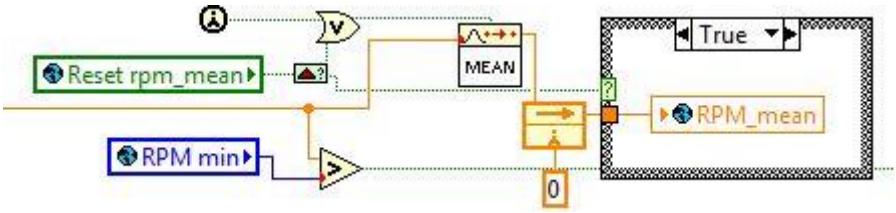


Figura 44: RPM_mean

3.3.2 *FFT loop*

Questo timed loop è quello che realizza il calcolo dell'FFT e del trendindex da essa derivato. Il suo tempo di ciclo viene impostato dall'utente ed è pari a quello del FIFO loop anche se è più critico per via della notevole mole di calcoli che il processore deve eseguire. La scelta del tempo di ciclo è determinante per l'analisi perché impone contemporaneamente:

- Il periodo con cui il trendindex viene aggiornato
- La risoluzione spettrale ottenibile

Infatti se T_loop viene preso pari a 50ms si avrà la pubblicazione di un nuovo indice ogni T_loop ms e la risoluzione spettrale ottenibile dalla FFT sarà pari ad 1/T_loop, cioè 20Hz. La scelta di un periodo di ciclo ampio migliora la qualità dello spettro prodotto, e potenzialmente la capacità di diagnosi, ma contemporaneamente espone l'analisi ad un'influenza sempre più marcata dei transitori. La scelta va effettuata tenendo conto del grado di dinamicità delle prove che il motore deve affrontare. È inoltre influenzato anche il ritardo stesso della pubblicazione dei risultati dell'analisi quantificabile, al netto dei ritardi trascurabili sull'FPGA, come:

$$T_{loop}FIFO + accumuli\ coda + T_{loop}FFT + buffer$$

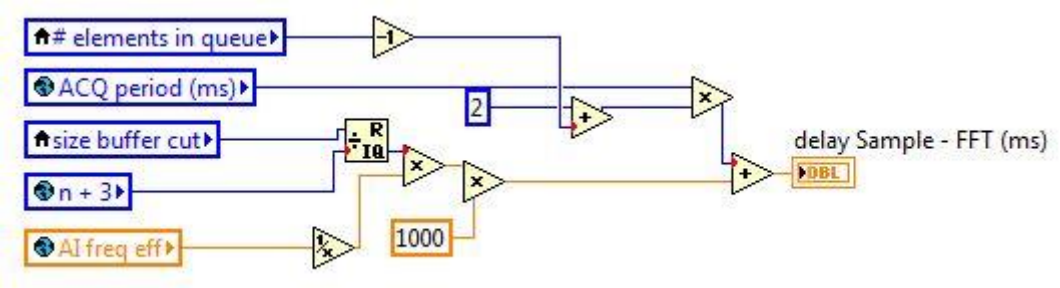


Figura 45: calcolo del ritardo della pubblicazione dell'analisi spettrale / indici

La coincidenza dei tempi di ciclo e le logiche di gestione del flusso dati impongono di fatto che non vi siano altri elementi in coda al di fuori di quello da analizzare e quindi l'unica variabile che influisce sul ritardo è la lunghezza del buffer, la cui costruzione è stata vista nel paragrafo precedente e che non può mai superare l'equivalente di T_loop. Il delay fra il campionamento e la pubblicazione oscillerà pertanto sempre fra 2 e 3 volte il tempo di ciclo.

All'interno del timed loop la prima operazione effettuata è lo svuotamento degli elementi presenti all'interno della coda inviata dal FIFO loop. L'esecuzione dell'intera analisi è comandata dal controllo booleano **Spectral Analysis,** che al suo avvio impone anche lo svuotamento della coda per eliminare eventuali accumuli verificatisi al precedente arresto. A questo punto viene verificato che il numero di elementi in coda (**# elements in queue**) sia maggiore o uguale a 1 e nel caso un booleano autorizza l'esecuzione della case structure che contiene il codice di analisi. Nel caso in cui si verifichi un accumulo di più di 1 elemento nella coda, dovuto ad esempio ad un ciclo

non ultimato in tempo, per evitare che l'analisi rimanga in ritardo e che la coda tenda ad aumentare è previsto che vengano svuotati tutti gli elementi e che venga preso solamente il più recente.

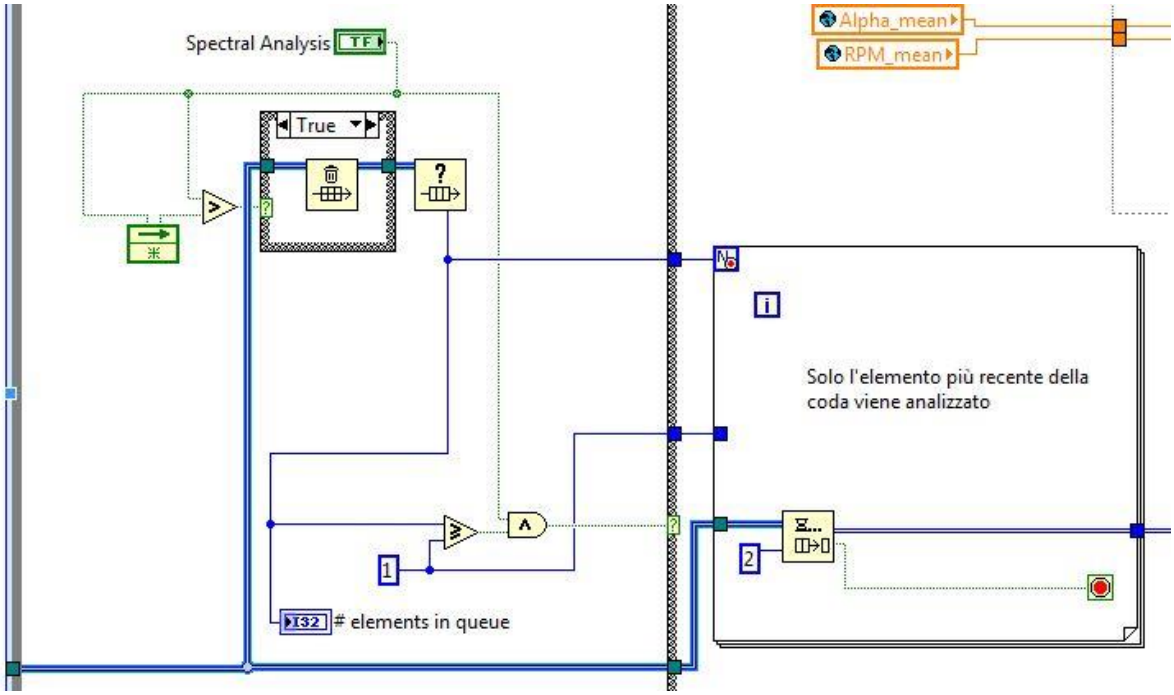


Figura 46: Svuotamento elementi della coda

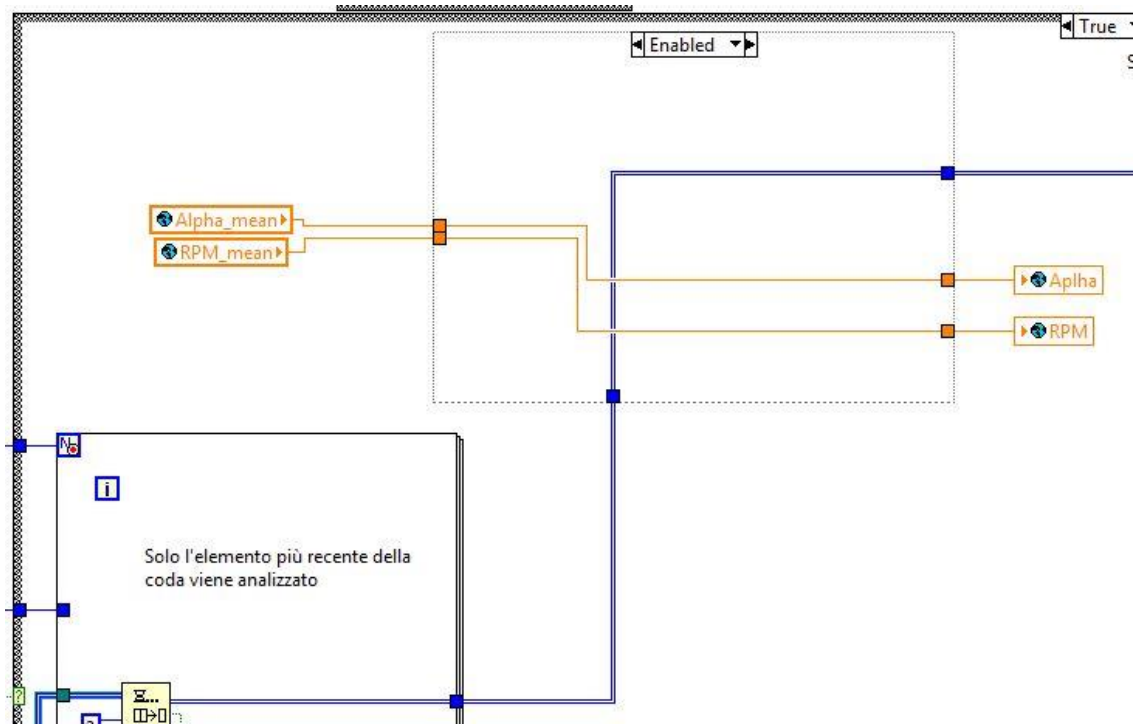


Figura 47: Invio dei dati all'analisi e assegnazione dei nuovi valori alle variabili Alpha e RPM

A questo punto i dati vengono inviati ai VI che eseguono l'analisi. Va ricordato che a i campioni sono ancora rappresentati come i16, così come escono dal convertitore A/D, per ridurre al minimo il loro peso sul processore. Vengono anche assegnati i valori di Alpha e RPM del presente set di dati per identificare il punto di funzionamento ed eseguire il calcolo del trendindex confrontandosi con la mappa vibrazionale. È stata prevista anche la possibilità di eseguire un overlap sul buffer di analisi perché se si utilizza una finestra di Hanning, con la quale si dà maggior risalto ai dati centrali del buffer, si deve necessariamente ricorrere ad un overlapping del 66%, al fine di non perdere dati. Al momento questa funzionalità è disabilitata poiché costituisce un aggravio dei costi computazionali (deve essere processato il triplo dei dati nello stesso tempo) e ne sarà valutata l'utilità sul calcolo del trendindex in futuro.

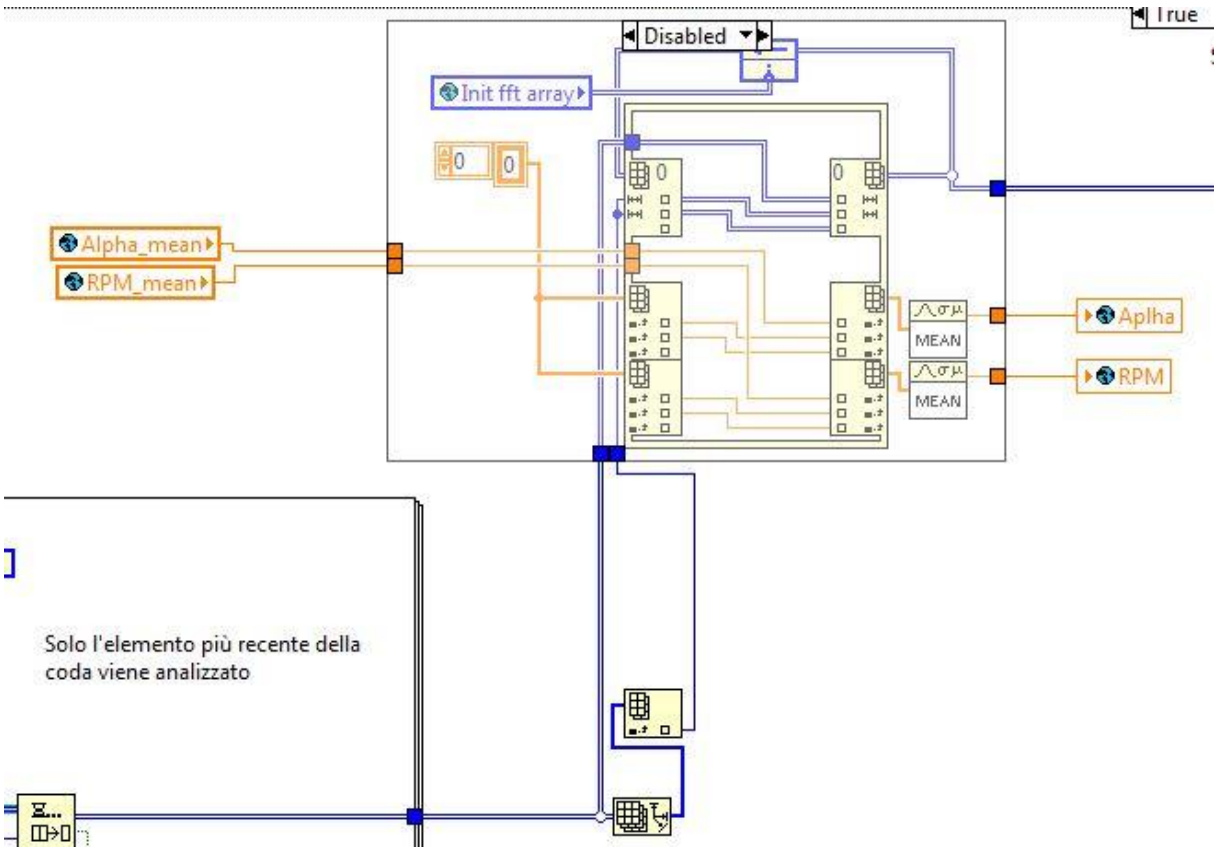


Figura 48: Overlap del buffer di analisi ottenuta con una in-place structure

I campioni e le informazioni relative alla condizione di funzionamento vengono poi inviate ai due subVI che eseguono rispettivamente l'analisi spettrale ed il calcolo del trendindex.

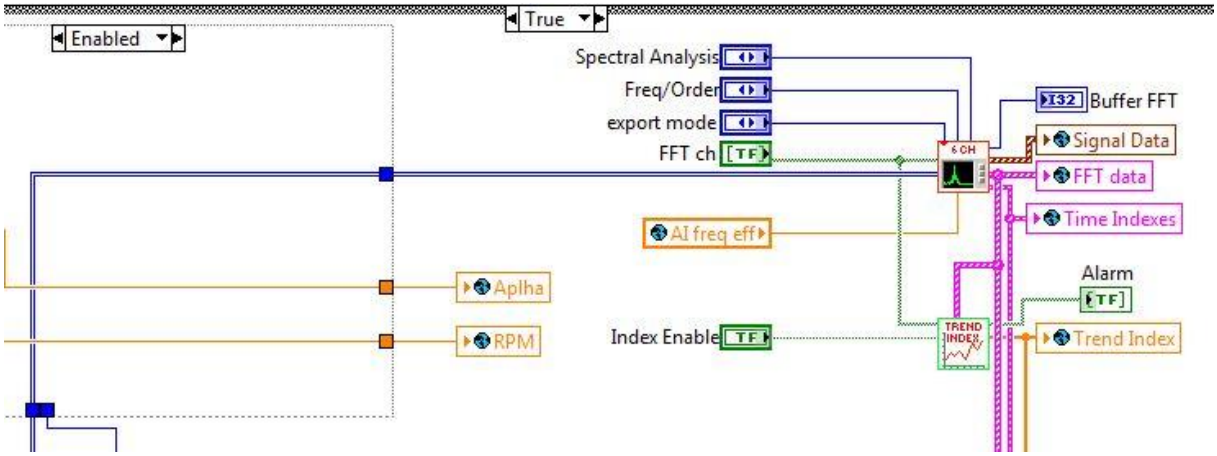


Figura 49: Spectral Analysis e trendindex VI

È consentito calcolare lo spettro al massimo sui primi 6 canali e l'utente da pannello frontale può impostare:

- **Spectral Analysis:** tipo di analisi (ampiezza o potenza)
- **Freq/Order:** visualizzazione come spettro in frequenza o degli ordini
- **FFT ch:** vettore di booleani che seleziona i canali su cui eseguire l'analisi

Come output del subVI dell'analisi spettrale troviamo:

- **FFT data:** l'array degli spettri
- **Signal Data:** un array di waveform contenenti i segnali su base tempo
- **Time indexes:** indici Peak, Crest Factor ed RMS calcolati sul buffer

Gli spettri dei canali attivi vengono poi confrontati con l valori di mappa all'interno del subVI TrendIndex che restituisce:

- **Trend Index:** array degli indici di degrado
- **Alarm:** array di boleani che indica il superamento della soglia massima imposta del trendindex.

Così come avveniva nel FIFO loop per i dati grezzi su base tempo, in questo loop gli indici sintetici e gli spettri calcolati possono essere inviati al SAVE loop per essere

salvati su file TDMS. La struttura delle code presenta lo stesso schema, con l'invio dei pacchetti di dati e del tempo corrispondente controllato da un contatore.

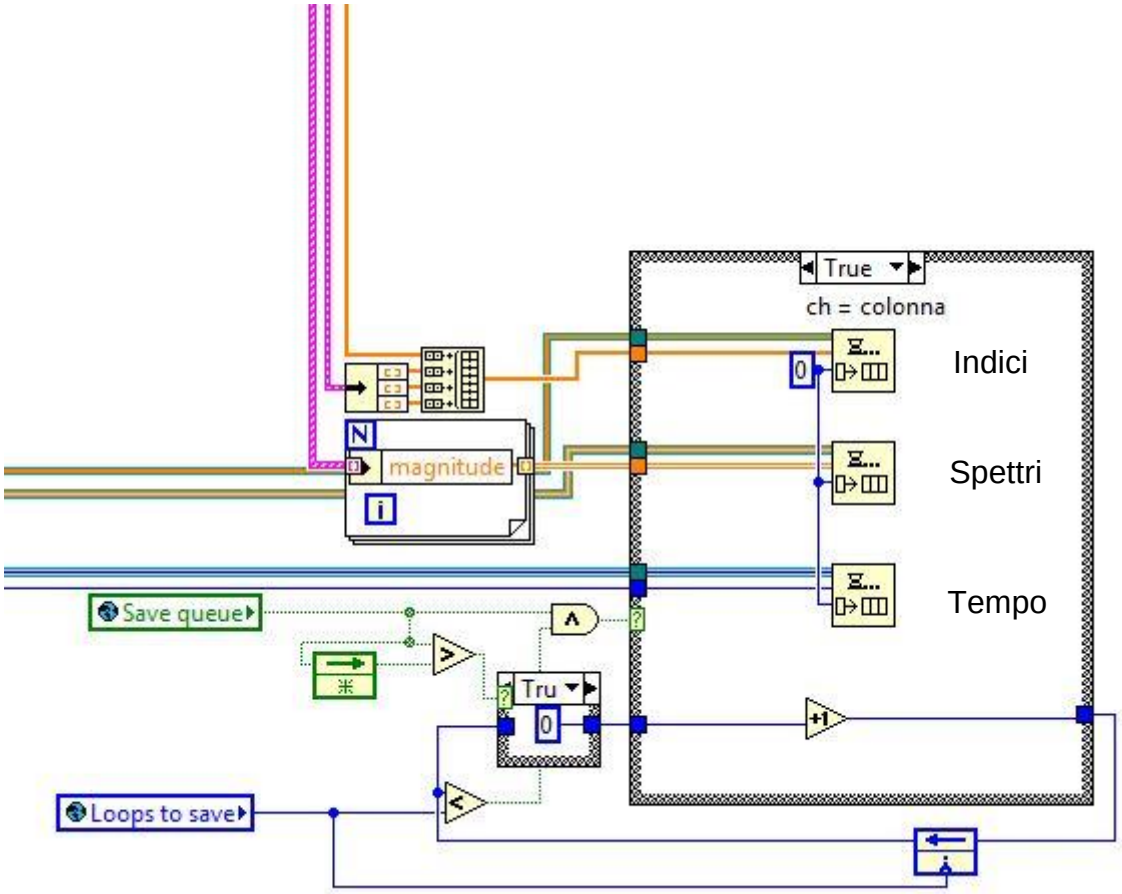


Figura 50: invio dei dati nella coda per il salvataggio

3.3.3 ***SAVE loop***

In questo timed loop viene eseguito il salvataggio dei dati ed a differenza dei loop visti precedentemente il tempo di esecuzione è più lento. Per rendere possibile la scrittura su file senza impegnare eccessivamente il processore, che contemporaneamente è impegnato ad eseguire anche tutte le altre parti di codice, è stato necessario desincronizzare l'invio dei dati ed il salvataggio vero e proprio degli stessi. Prendendo a riferimento il solito T_loop = 50ms per FIFO e FFT le tempistiche ed il flusso dati sono gestite come segue:

- Ad ogni ciclo di FIFO loop e FFT loop 1 pacchetto di dati viene inviato in coda

- Al raggiungimento di 20 elementi avviene lo svuotamento ed il salvataggio dei dati
- Il SAVE loop viene eseguito con un T_loop = 500ms, 10 volte superiore a quello degli altri, ma ogni volta vengono salvati 20 elementi. È possibile registrare dati relativi ad un multiplo temporale di 20*50ms = 1s.

Questo significa che il salvataggio viene eseguito 1 volta ogni 2 cicli, avendo così la certezza che non si formino accumuli nella coda. Un accorgimento di questo tipo è necessario perché è molto probabile che il ciclo possa impiegare più di quanto imposto, vista l'onerosità dell'operazione, e consente anche di mantenere il processore libero di svolgere le altre funzioni a più alta priorità.

Le modalità di salvataggio attualmente implementate sono 3 e consistono in:

- Salvataggio manuale abilitato e disabilitato con il pulsante **REC**
- Salvataggio con counter ad arresto automatico
- Salvataggio avviato con trigger di soglia sul canale

L'avvio della registrazione è comandato o dal booleano **Save Trigger** o dal pulsante **REC.** In caso di salvataggio manuale, al primo ciclo in cui il valore di **REC** è maggiore del ciclo precedente, viene inizializzata a 0 la variabile **Loops to save** che controlla il numero di pacchetti da salvare. Questa variabile globale è presente anche nei loop FIFO e FFT per il controllo del numero di elementi da inviare nelle code. Ad ogni iterazione **Loops to save** viene aumentata di 10, consentendo il salvataggio continuo fino al segnale di stop che blocca la somma. In presenza invece di salvataggio con contatore **Loops to save** viene semplicemente inizializzata con il valore corrispondente al tempo di registrazione desiderato (20 = 1s di registrazione).

La variabile **Loops to save** è fondamentale per il controllo preciso del numero di elementi da salvare e per avere lo stesso tempo di registrazione per ogni tipo di dato, che sia grezzo, indice o spettro. Infatti essa gestisce sia il termine dell'invio nelle code dei pacchetti di dati, sia lo stop vero e proprio della scrittura attraverso il confronto con i contatori di salvataggio **C.Save Analog** e **C.Save Indexes.** Quando il numero di

elementi desiderati provenienti dalle code è stato scritto su file **REC** viene automaticamente posto su falso e così come **Save queue**, che controlla le case structure in cui sono contenute le code. In sostanza la quantità di dati salvati non è direttamente controllata dall'input di salvataggio, ma indirettamente dalla variabile **Loops to save**.

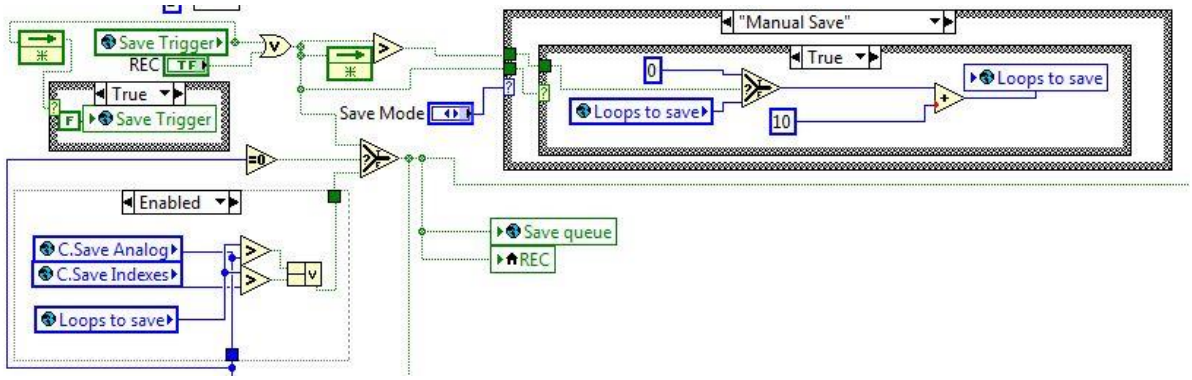


Figura 51: configurazione manual save

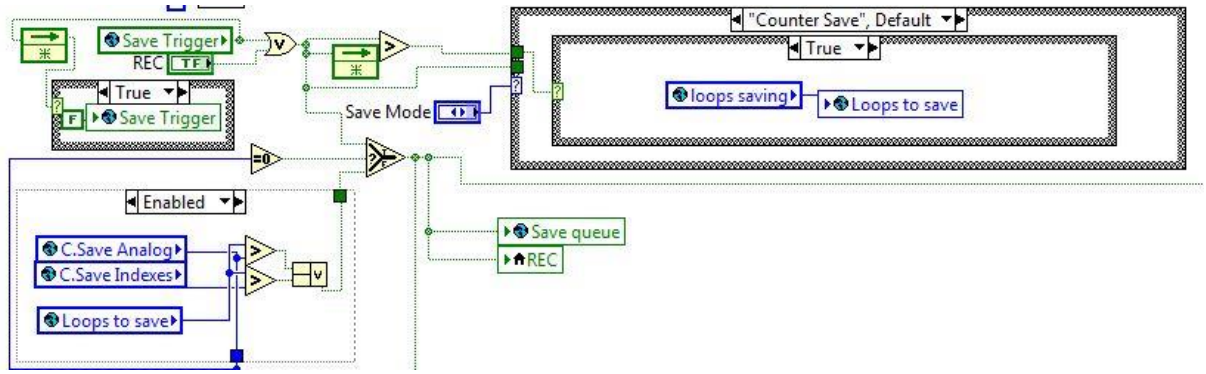


Figura 52: configurazione counter save

Dallo schema si può vedere che un booleano scorre verso il basso: il valore di questo elemento determina la catena di apertura, scrittura e chiusura del file, ed il suo stato (true o false) è visibile a front panel dal pulsante **REC,** a cui è collegato**.** Questa variabile controlla l'intera case structure di salvataggio, abilitando la registrazione per tutto il tempo in cui è posta a true più un ciclo. Il ciclo in più è necessario per operare la chiusura del file successivamente alla fine della scrittura.

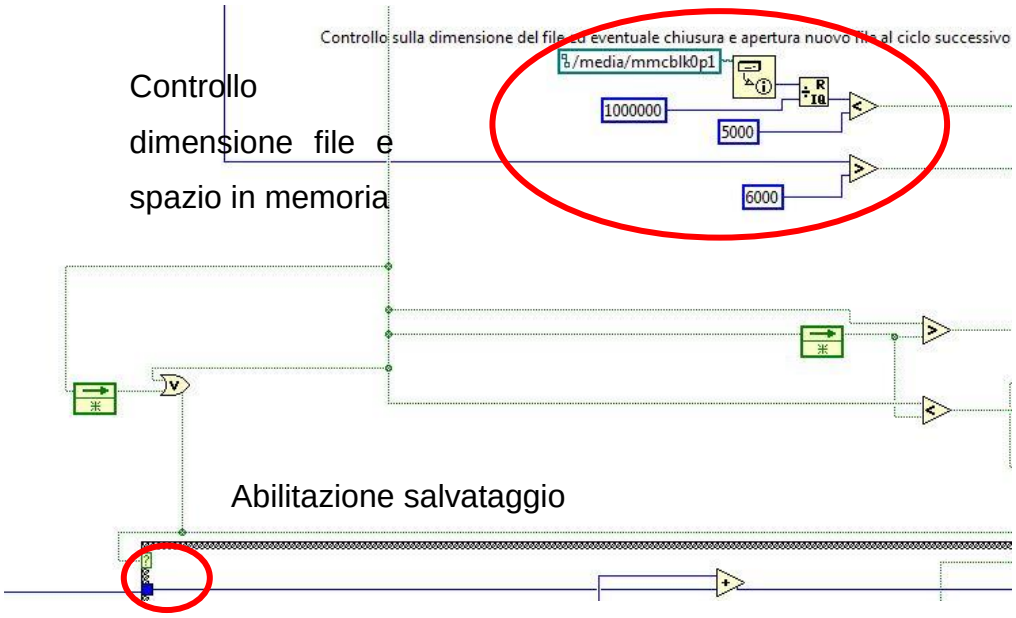


Figura 53: abilitazione del salvataggio e controllo

Ad ogni iterazione vengono effettuati dei controlli sulla dimensione del file e sullo spazio disponibile in memoria, al fine di evitare la produzione di file troppo grandi, impossibili da aprire, e per impedire il riempimento della memoria che causerebbe il blocco del sistema. Questi controlli al momento sono effettuati sul numero di iterazioni di salvataggio effettuate.

- Se lo spazio in memoria diventa minore di una certa soglia la registrazione viene interrotta.
- Se la dimensione del file eccede un certo valore, il file attuale viene chiuso e ne viene creato uno nuovo.

Quando la struttura che contiene il codice di registrazione viene attivata dal booleano di controllo, parallelamente avviene il settaggio dei booleani che gestiscono l'apertura, la scrittura e la chiusura del file. Il tipo di formato scelto per il salvataggio è il TDMS, un formato binario nativo di Labview, ma facilmente maneggiabile anche con altri software come Matlab, che contiene una parte header di informazioni e proprietà sul file e che per i dati viene strutturato in gruppi e canali. Essendo un formato binario, lo spazio occupato su disco, a meno delle informazioni accessorie, è quello minimo richiesto dalla rappresentazione dei dati.

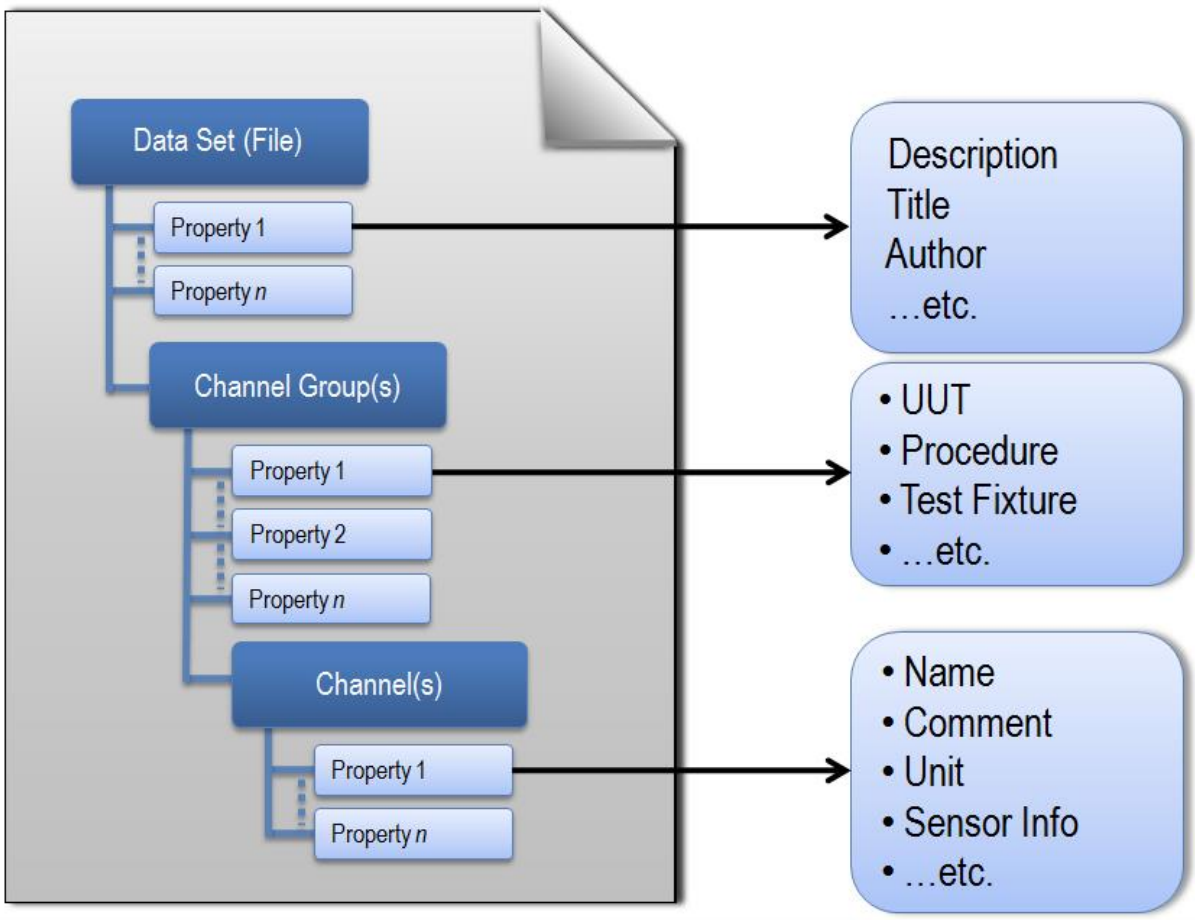


Figura 54: formato TDMS

Come detto precedentemente il file va aperto per poi procedere alla scrittura ed infine chiuso.

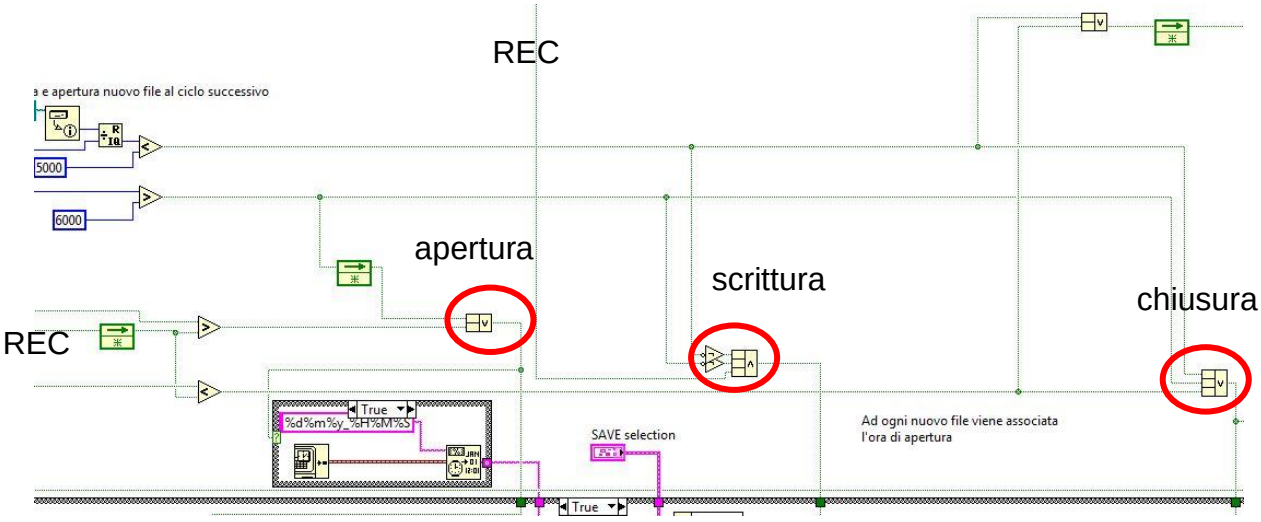


Figura 55: controllo di apertura, scrittura e chiusura

Apertura: è comandata quando l'attuale valore di **REC** è maggiore di quello precedente, o al ciclo successivo del superamento della dimensione massima del file.

Scrittura: è attiva per tutto il tempo in cui **REC** rimane true, a meno dell'intervento dei controlli sulla dimensione file e sullo spazio in memoria.

Chiusura: viene comandata quando **REC** passa da true a false, oppure quando intervengono i controlli sulla dimensione file e sullo spazio in memoria.

Ad ogni apertura di un nuovo file viene anche preso il timestamp ed associato al nome del file, in modo da tenere traccia dell'orario ed impedire la sovrascrittura accidentale dei file.

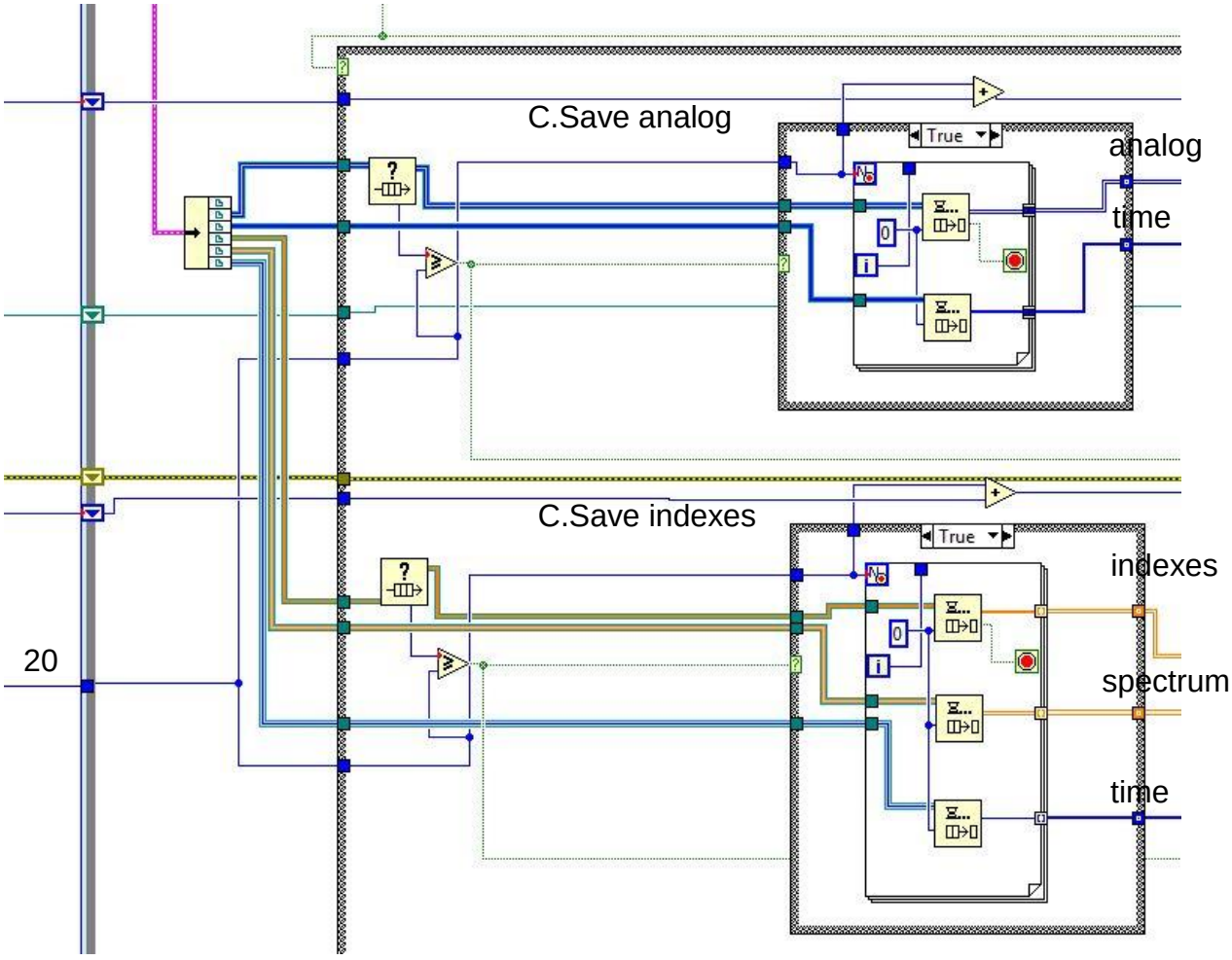


Figura 56: svuotamento code

Quando la struttura di salvataggio è abilitata dalla variabile **REC** e l'invio dei pacchetti di dati nelle code viene concesso da **Save queue**, ha inizio anche lo svuotamento delle stesse. Come visto in precedenza viene verificata la presenza di almeno 20 elementi in coda e poi ne si procede allo svuotamento. In parallelo i counter **C.Save analog** e **C.Save indexes** tengono traccia dei pacchetti salvati. La differente tipologia delle uscite dai for loop per i diversi dati (concatenating o indexing) è dovuta semplicemente alla differente organizzazione dei dati per consentire il salvataggio secondo la precisa struttura di gruppi e canali che verrà mostrata in seguito.

A valle della raccolta degli elementi dalle code è presente la case structure dedicata all'apertura del file ed al settaggio delle proprietà dei canali, come ad esempio la frequenza di campionamento, i guadagni e gli offset dei canali abilitati al salvataggio (**Save ch**).

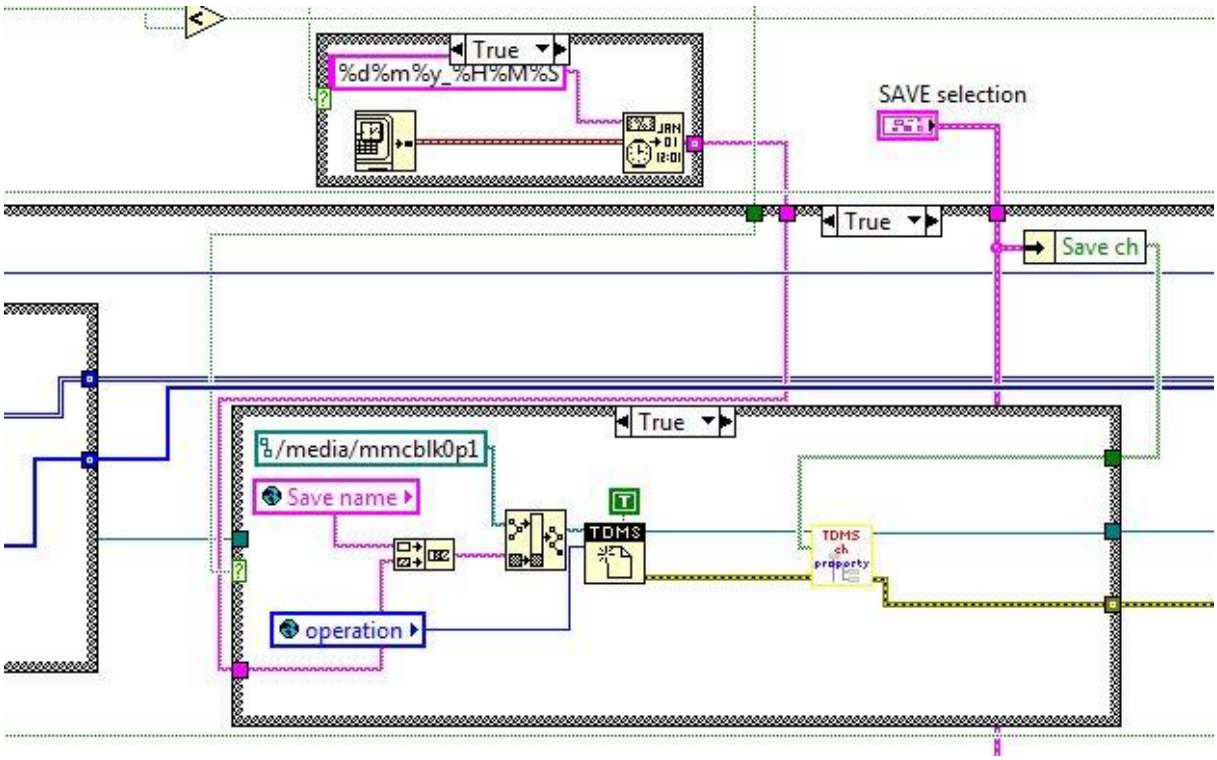


Figura 57: apertura del file e settaggio delle proprietà dei canali da salvare

Successivamente è possibile procedere alla scrittura dei dati, che avviene su di un unico file TDMS eseguendo più blocchi di scrittura in serie. La struttura del TDMS creato è la seguente:

- **Gruppo Analogici grezzi (i16):** i dati provenienti dai canali vengono salvati su altrettanti canali sul file. Per risparmiare risorse e memoria il salvataggio è in rappresentazione i16 accompagnata dalle informazioni di guadagno e offset per la conversione in unità ingegneristiche. Il tempo corrispondente ai campioni è salvato in u64.
- **Gruppo indici (double):** gli indici sintetici Trend Index, Peak, Crest Factor e RMS sono salvati per ogni canale in diversi "canali del file". Il tempo corrispondente è salvato in u64.
- **Gruppi analisi spettrale (double):** Per ogni canale su cui è attiva l'analisi spettrale viene creato un gruppo nel file, al cui interno sono presenti N/2 canali rappresentati le linee spettrali nel tempo (con N numero dei campioni d'analisi). Il tempo corrispondente è salvato in u64 in un gruppo a parte chiamato Spectrum Time.

L'impostazione dei nomi dei canali e la costruzione dei nomi di salvataggio avviene in un loop a più bassa priorità, sfruttando parte dei controlli di OBI.

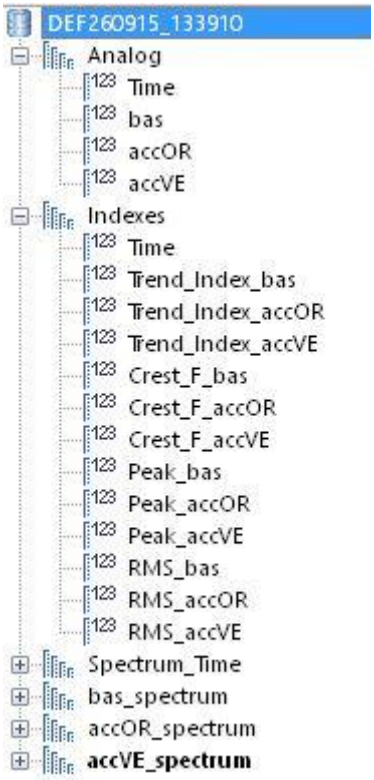


Figura 58: file in cui sono stati salvati 3 canali accelerometrici, includendo anche indici e spettri.

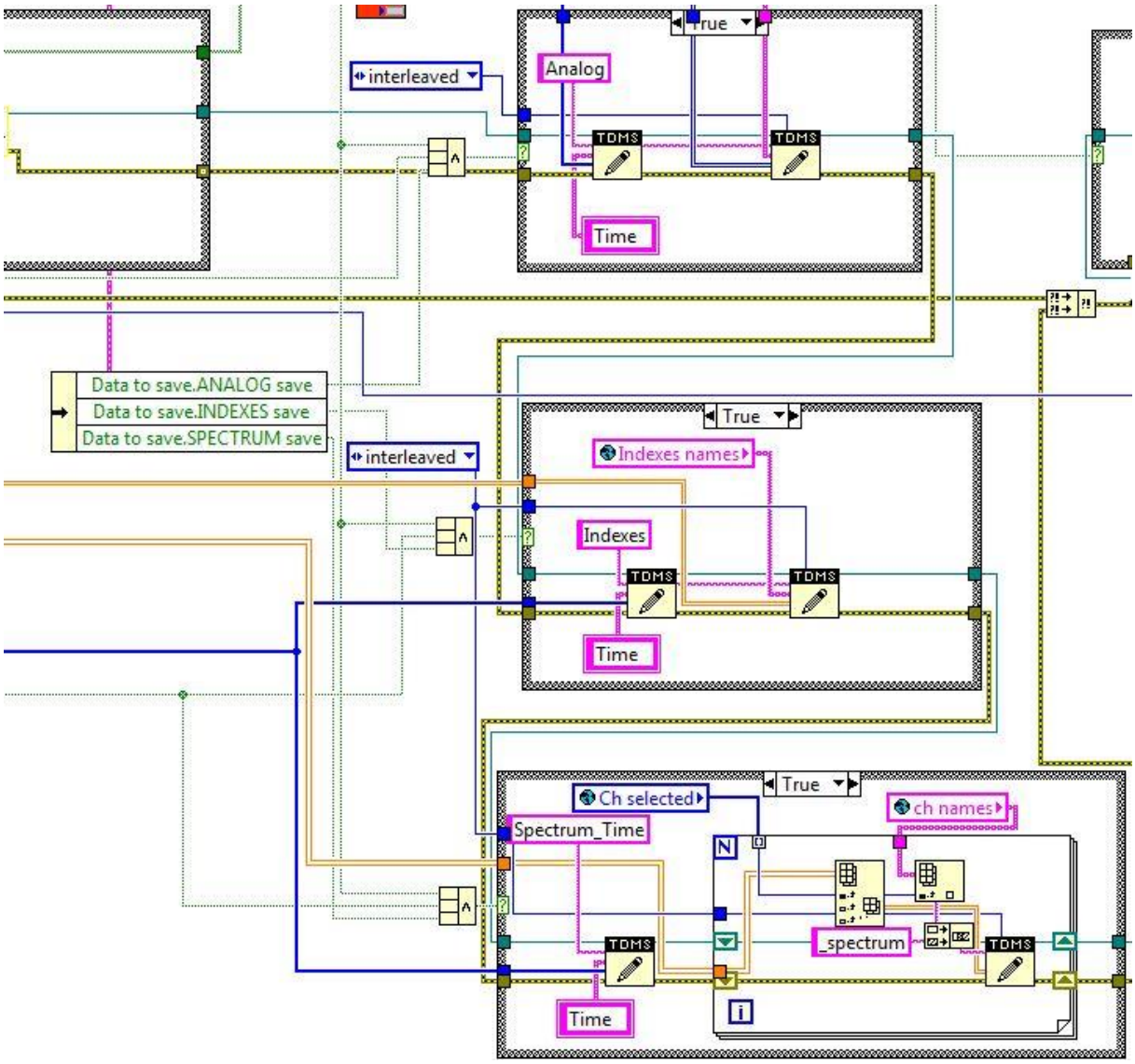


Figura 59: codice che gestisce la scrittura su file in gruppi e canali

Affinché la scrittura possa avvenire devono essere contemporaneamente posti a true i seguenti booleani:

- booleano che seleziona il tipo di dato da salvare
- booleano che indica la ricezione di 20 pacchetti di dati
- booleano che abilita la fase di scrittura

Superato il codice di scrittura, se si verificano le condizioni di chiusura, si termina il file e la variabile **Loops to save** viene posta a 0 e le code vengono “flushate”. In caso di interruzione dell’esecuzione dell’intero software Co-Mo attraverso il controllo **Stop,** è previsto che si attenda il termine del salvataggio prima del blocco delle iterazioni.

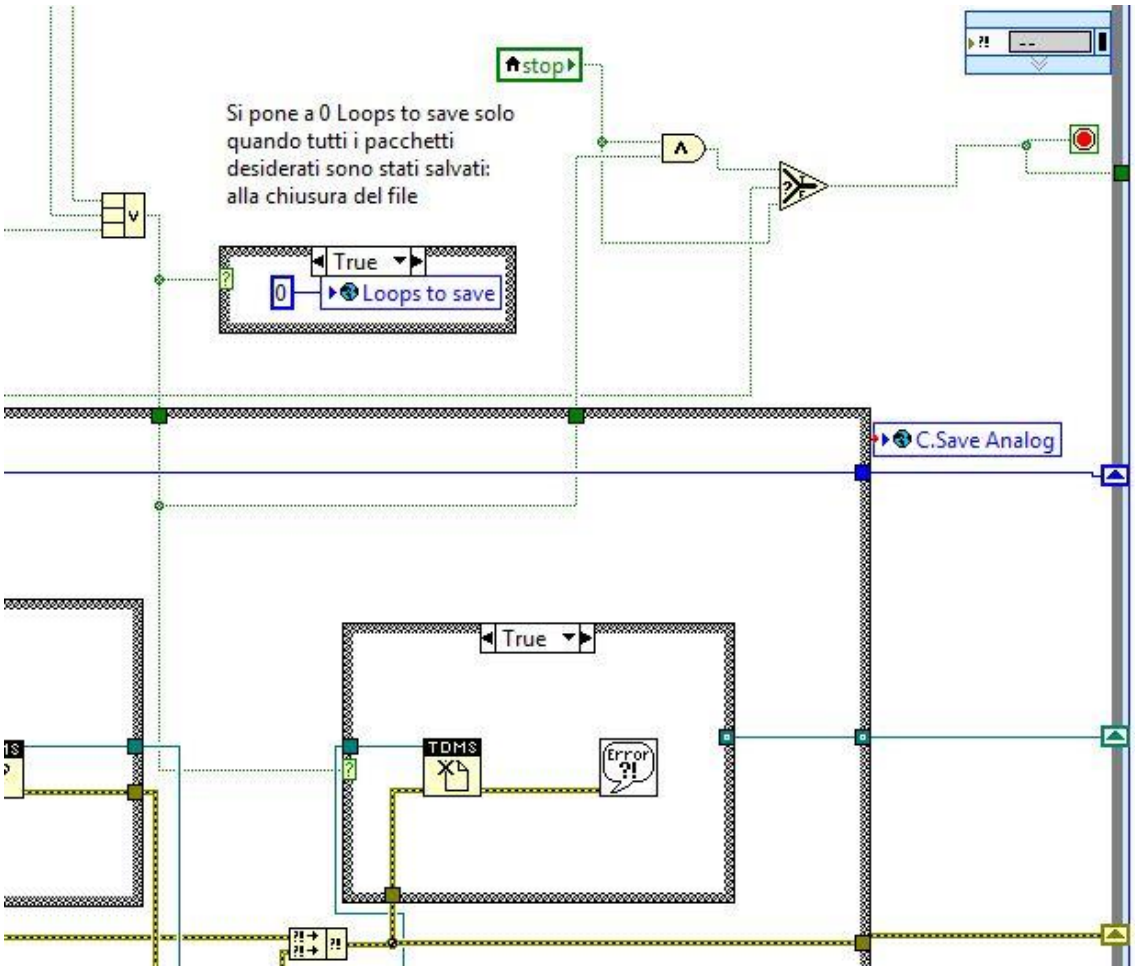


Figura 60: chiusura file

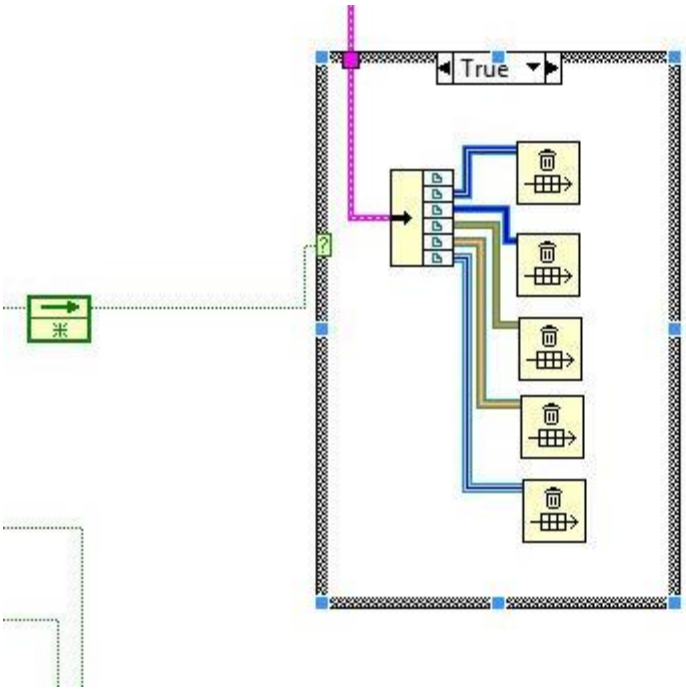


Figura 61: flush delle code per evitare eventuali accumuli alla chiusura

3.4 FUNZIONI DI ANALISI

In questo paragrafo saranno presentati i 2 subVI ed il loop dedicati alla gestione dell'analisi spettrale, della costruzione della mappa di vibrazione e del calcolo del trendindex. Sarà chiarito solamente il funzionamento del codice, mentre dettagli più approfonditi sui settaggi e sulle modalità di costruzione della mappa e dell'indice verranno affrontati nel capitolo 4.

3.4.1 *Spectral Analysis*

Il subVI spectral analysis esegue la FFT che converte il segnale su base tempo in uno spettro in frequenza. I canali su cui è possibile svolgere l'operazione sono al massimo i primi 6, selezionati attraverso l'array di booleani **FFT ch.**

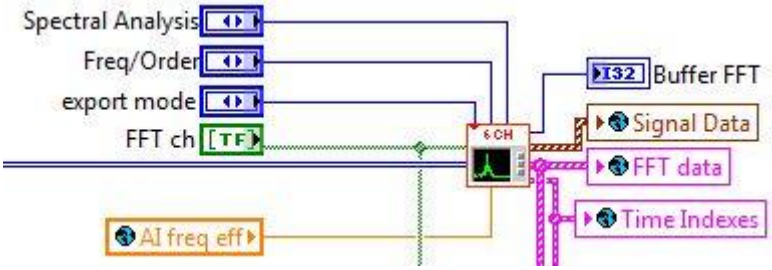


Figura 62: subVI spectral analysis

La prima operazione effettuata sull'array 2D in ingresso, contenente i campioni separati per canale, è la conversione da bit ad unità ingegneristica (g) grazie all'applicazione di **gain** ed **offset** opportunamente calcolati a partire dalle impostazioni utente.

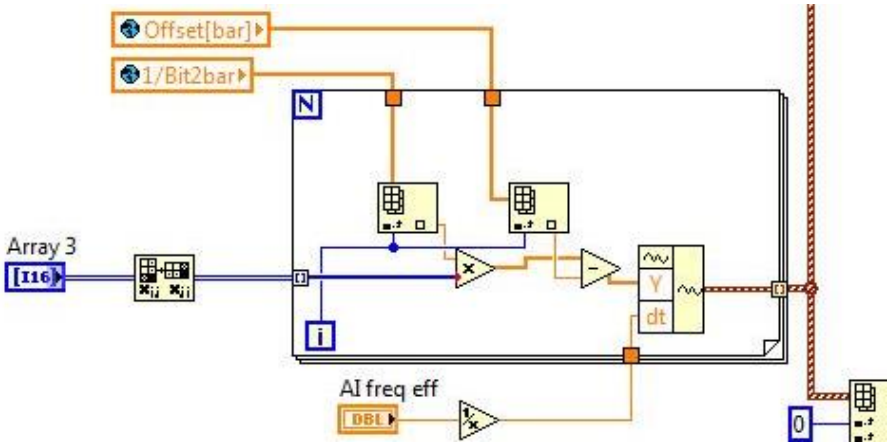


Figura 63: conversione da i16 a waveform

Successivamente i dati double così ottenuti sono uniti al periodo di campionamento per formare un array di waveform da utilizzare nei VI che calcolano lo spettro. Attraverso il controllo **Spectral Analysis** è possibile scegliere se eseguire l'FFT o la PSD/Power Spectrum del segnale, mentre il booleano **FFT ch** abilita solamente i canali selezionati. In uscita si trovano quindi la waveform stessa con il segnale su base tempo e l'array di cluster che contiene lo spettro appena calcolato

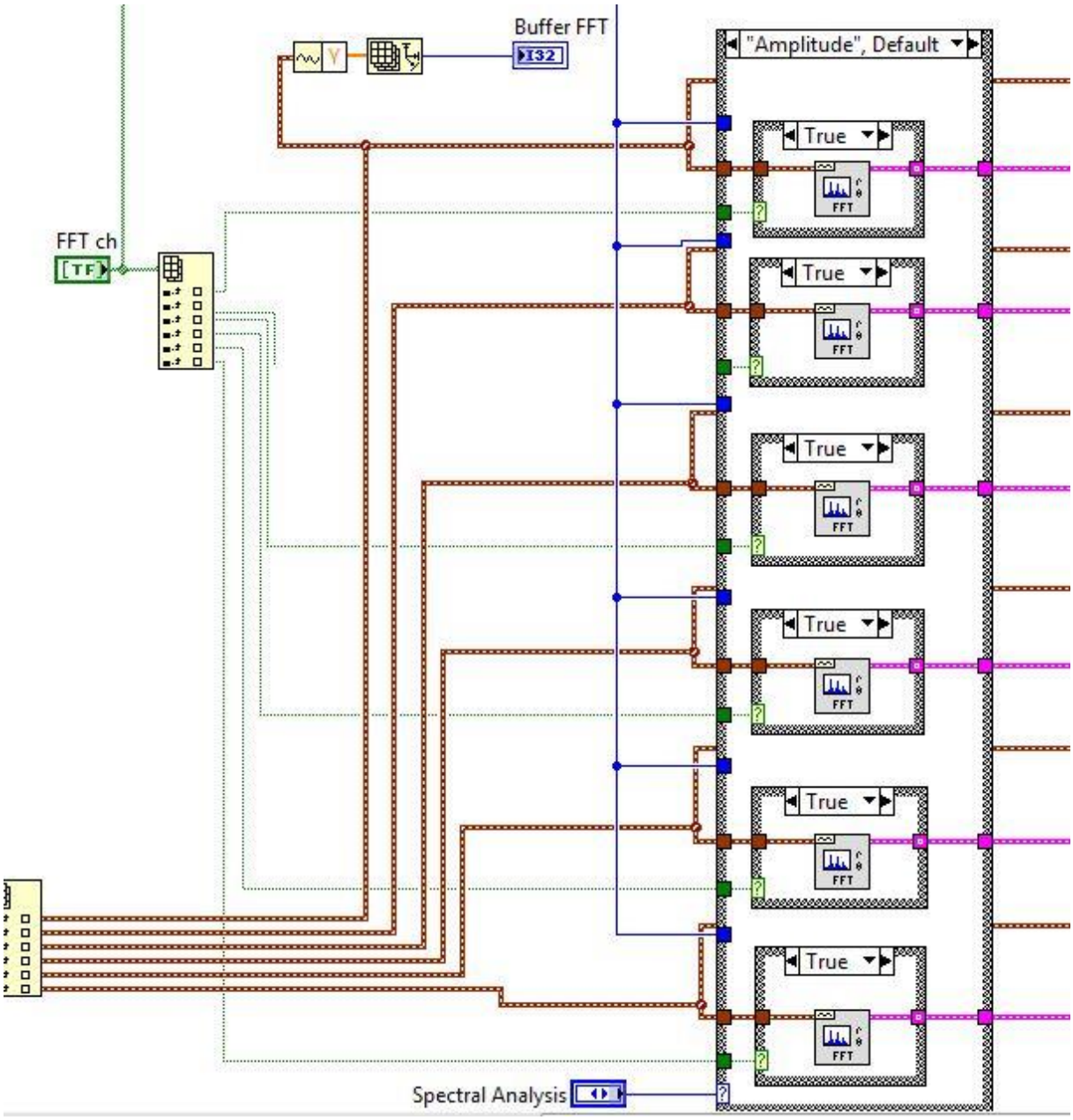


Figura 64: FFT

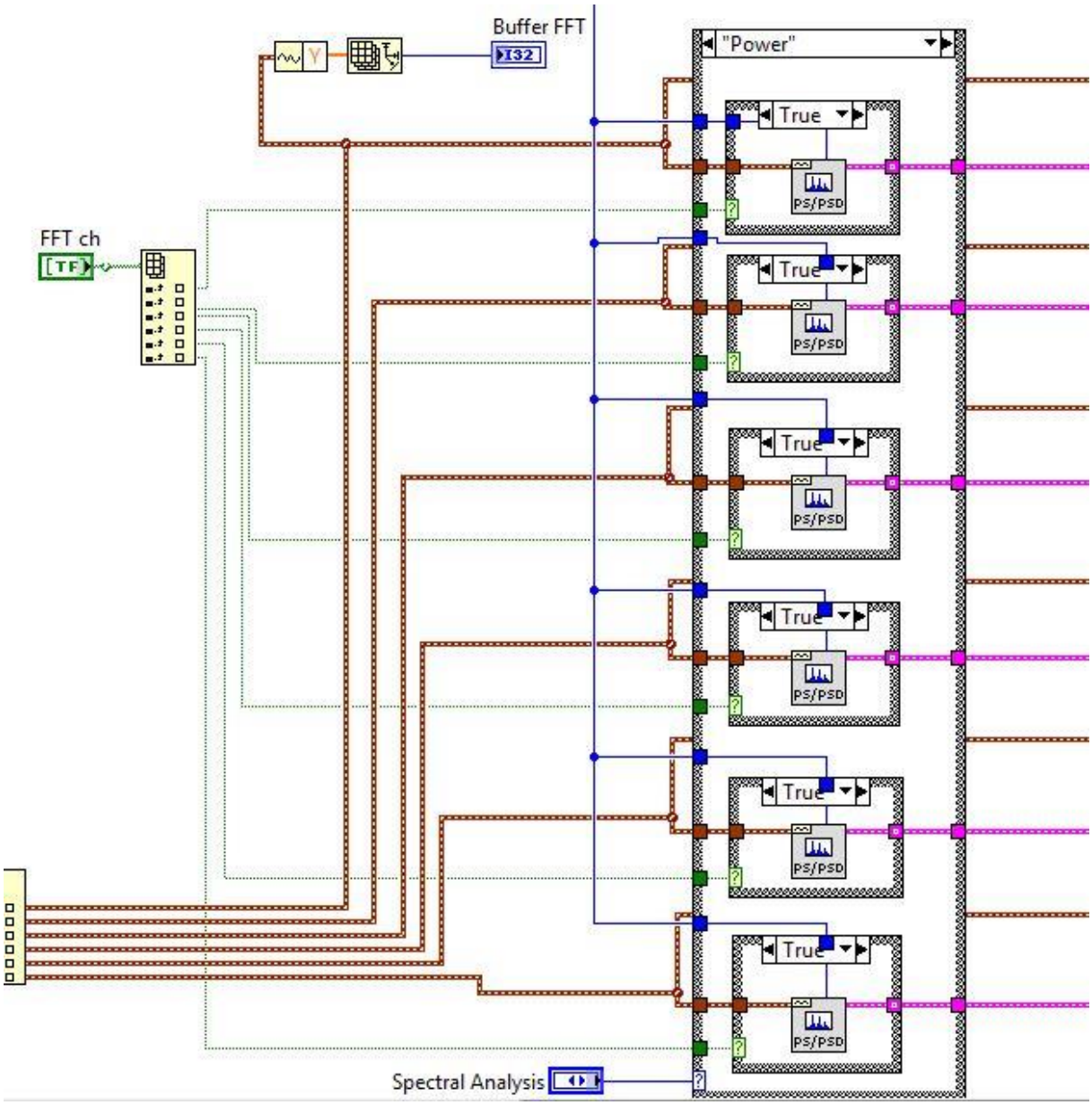


Figura 65: PSD / Power Spectrum

L'analisi è operata attraverso dei VI della "waveform mesurement palette", comodi perché hanno già implementate tutte le funzioni che possono essere utili come ad esempio la finestratura, la media e la rappresentazione in dB, oltre a presentare il risultato in un formato direttamente visualizzabile ed utilizzabile semplicemente. Ai fini della visualizzazione è anche possibile scegliere la rappresentazione come spettro degli ordini operando la conversione attraverso l'informazione degli RPM corrispondenti al buffer di dati in esame.

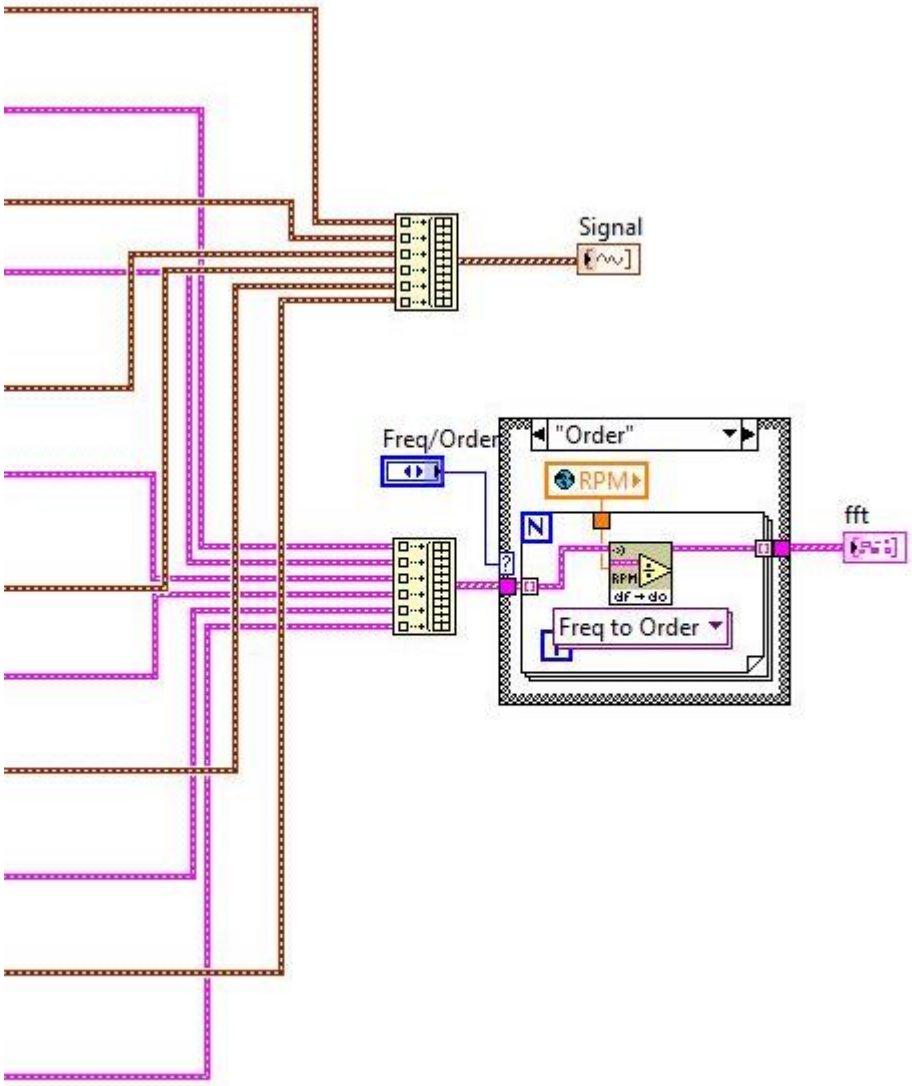


Figura 66: eventuale conversione da frequenza ad ordini

In uscita dal VI, oltre agli indicatori **Signal** e **fft**, viene inviato anche un array di cluster contenete gli indici calcolati su base tempo Crest Factor, Peak ed RMS.

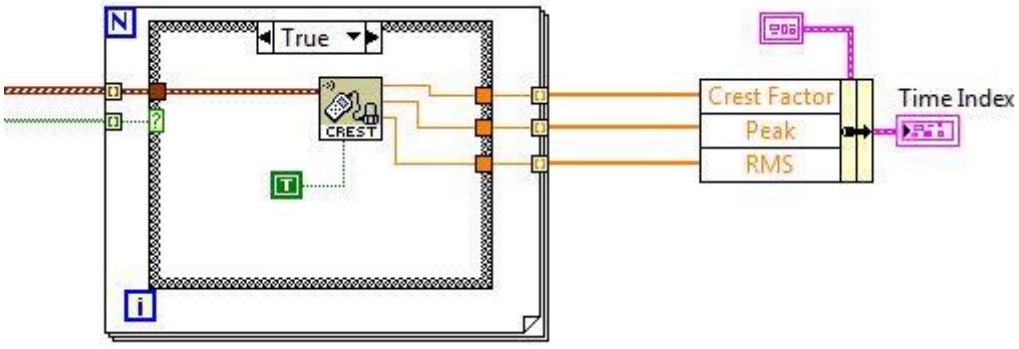


Figura 67: calcolo Crest Factor, Peak e RMS

3.4.2 *Map building*

Prendendo ad esempio il dispositivo delta-Aalyser è stata implementato un loop (sempre a 50ms) in grado di costruire una mappa del comportamento vibrazionale del motore. Le variabili scelte per l'identificazione del punto di funzionamento al momento sono semplicemente il carico richiesto al motore (espresso come percentuale) e gli RPM. Le mappe, una per ogni possibile accelerometro montato, sono salvate nella variabile globale **MAP_clust**, inizializzata con degli zeri all'avvio del programma. Per ogni singolo accelerometro mappato viene salvato un cluster con:

- **Nome**: per identificare l'accelerometro.
- **Mappa:** un array 3D che contiene per ogni valore di **Alpha** e **RPM** il corrispondente vettore delle linee spettrali.
- **Weigths:** un array 2D basato su **Alpha** e **RPM** che contiene il numero di passaggi per la stessa condizione di funzionamento con cui è stata costruita la mappa vibrazionale.

L'inizializzazione dei vettori delle condizioni di funzionamento e della mappa avviene all'avvio del software secondo quanto stabilito dall'utente:

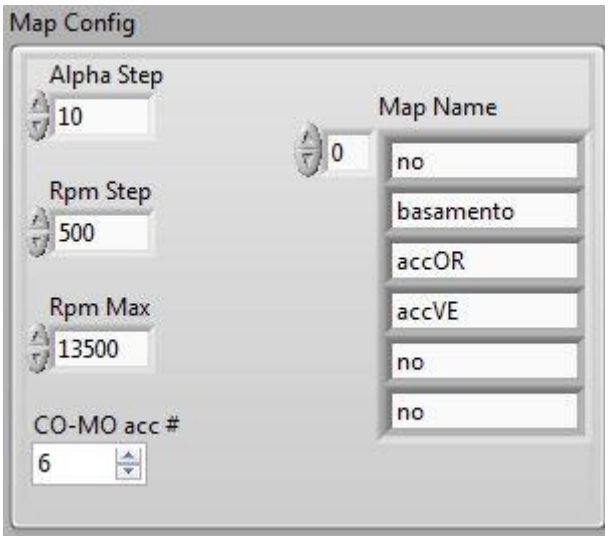


Figura 68: configurazione da pannello frontale degli step di Alpha e RPM con cui costruire la griglia della mappa, oltre alla selezione del numero e dei nomi delle mappe da inizializzare

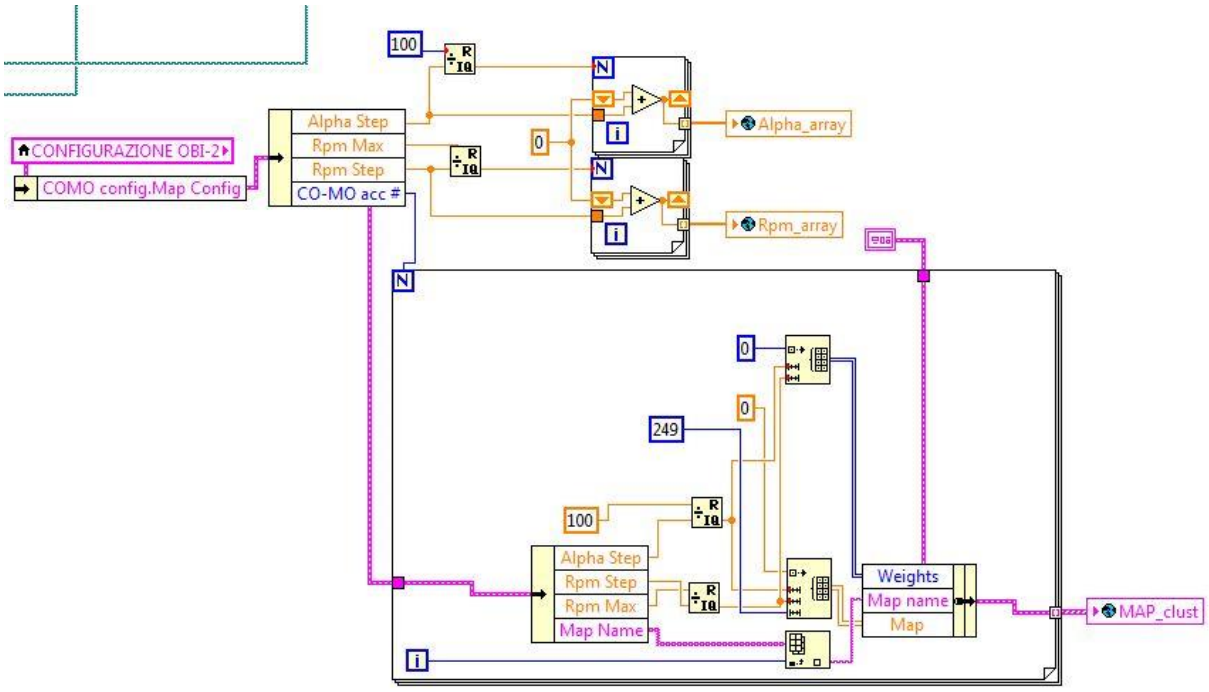


Figura 69: inizializzazione della mappa e dei vettori di RPM e Alpha

Al momento per ogni condizione d funzionamento vengono salvate le prime 250 linee spettrali, che, con un T_loop di 50 ms, corrispondono ad una banda da 0 a 5 kHz. È possibile impostare un range spettrale differente per la mappa e per la costruzione del trendindex, ma come si vedrà in nel capitolo 4 questo è un buon range di partenza che consente di evitare le frequenze d'interesse della detonazione e di ridurre il numero di elementi da maneggiare.

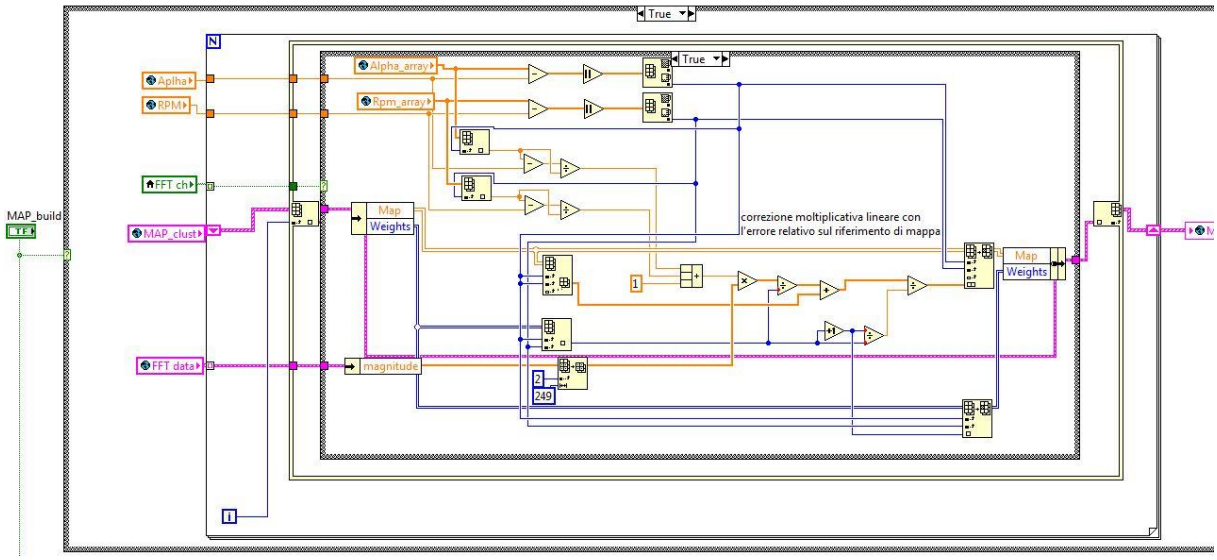


Figura 70: codice di costruzione della mappa

Il codice, abilitato dal pulsante **MAP_build**, si sviluppa all'interno di una in-place structure in cui gli elementi della mappa vengono via via aggiornati. A partire dagli ingressi Alpha e RPM viene stabilita la condizione di funzionamento attuale e vengono individuati gli indici dell'elemento corrispondente nella mappa.

$$RPM_{mappa_i} = min\left|RPM_{mappa} - RPM\right| \qquad \rightarrow \qquad i_{rpm}$$

$$Alpha_{mappa_i} = min\left|Alpha_{mappa} - Alpha\right| \qquad \rightarrow \qquad i_{alpha}$$

Successivamente l'elemento della mappa viene aggiornato eseguendo la media pesata con lo spettro attuale **FFT data:**

$$Mappa_{new} = Mappa + \frac{FFT\ data}{weigth}$$

L'elemento corrispondente della matrice dei pesi vieni aumentato di 1 ogni volta che lo stesso elemento della mappa viene aggiornato. Lo spettro da salvare, a rigore, dovrebbe essere interpolato bi-linearmente, elemento per elemento, su RPM e Alpha, ma il costo computazionale sarebbe troppo elevato. Inoltre, in questa fase è impossibile eseguire un'interpolazione fra diversi valori della mappa dato che è ancora in costruzione. **FFT data** viene quindi corretto tramite un coefficiente moltiplicativo costruito come segue:

$$FFTdata_{corr} = FFTdata * (1 + Coeff_{RPM} + Coeff_{Alpha})$$

Per ottenere i coefficienti si svolgono due interpolazioni lineari indipendenti (l'interpolazione non è bilineare) su RPM e Alpha, che tengono conto delle differenze fra giri e carico reali e di mappa. Facendo riferimento ai soli giri (per il carico il procedimento è identico) una volta identificata la condizione di funzionamento di riferimento si può affermare che:

$$RPM_{mappa_i} - s_{inf} < RPM < RPM_{mappa_i} + s_{sup}$$

Con s_{sup} e s_{inf} , che in caso di griglia della mappa non equi-spaziata, possono anche essere differenti:

$$s_{sup} = \frac{RPM_{mappa_{i+1}} - RPM_{mappa_i}}{2}$$

$$s_{inf} = \frac{RPM_{mappa_i} - RPM_{mappa_{i-1}}}{2}$$

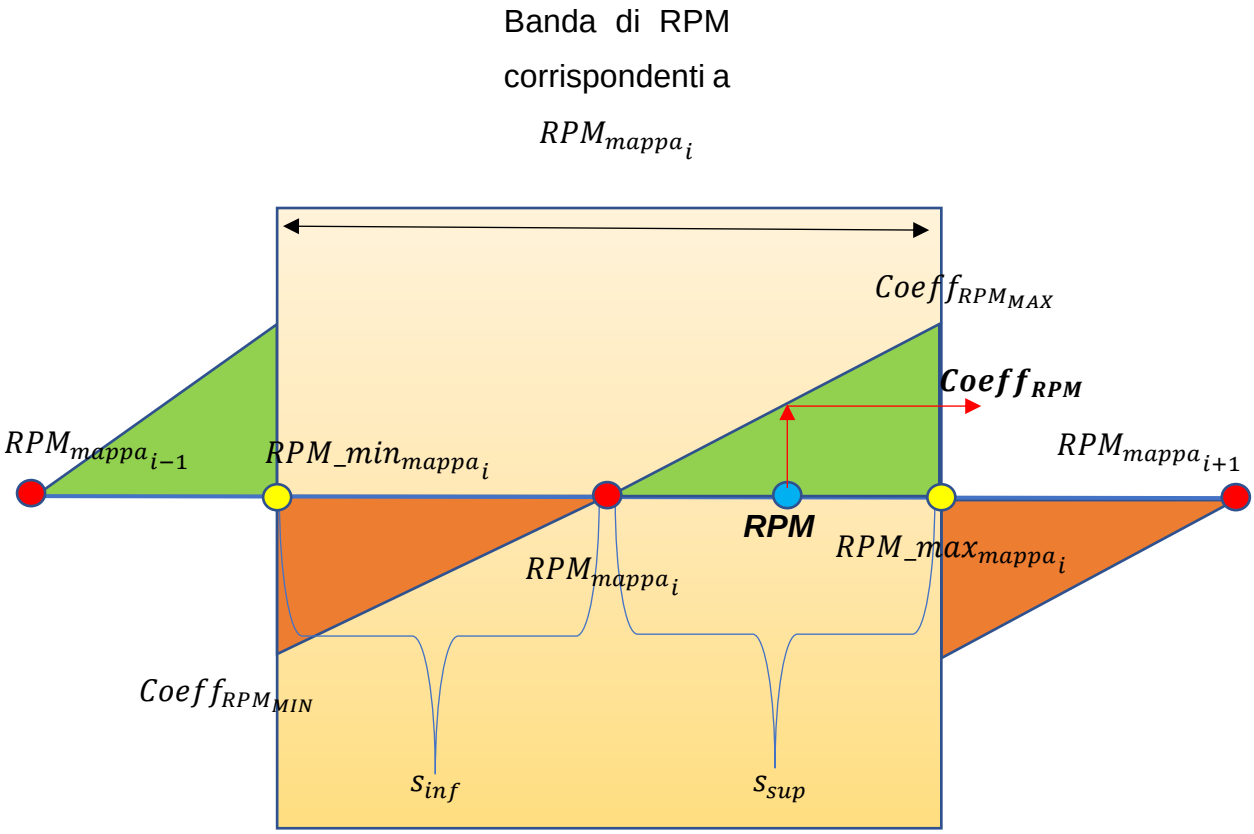


Figura 71: rappresentazione grafica del calcolo dei coefficienti di correzione

Il coefficiente correttivo da applicare viene quindi trovato come interpolazione lineare fra gli estremi della banda della condizione di funzionamento, a cui sono assegnati i valori (che oscillano attorno allo 0):

$$Coeff_{RPM_{MAX}} = \frac{RPM\_max_{mappa_i} - RPM_{mappa_i}}{RPM_{mappa_i}} = \frac{(RPM_{mappa_i} + s_{sup}) - RPM_{mappa_i}}{RPM_{mappa_i}}$$

$$Coeff_{RPM_{MIN}} = \frac{RPM\_min_{mappa_i} - RPM_{mappa_i}}{RPM_{mappa_i}} = \frac{(RPM_{mappa_i} - s_{inf}) - RPM_{mappa_i}}{RPM_{mappa_i}}$$

Interpolando linearmente:

$$y = y_1 + \frac{(y_2 - y_1)}{(x_2 - x_1)} * (x - x_1)$$

$$Coeff_{RPM} = Coeff_{RPM_{MIN}} + \frac{\left(Coeff_{RPM_{MAX}} - Coeff_{RPM_{MIN}}\right)}{\left(RPM\_max_{mappa_i} - RPM\_min_{mappa_i}\right)} * (RPM - RPM\_min_{mappa_i})$$

$$Coeff_{RPM} = \frac{RPM\_min_{mappa_i} - RPM_{mappa_i}}{RPM_{mappa_i}} + \frac{RPM\_max_{mappa_i} - RPM\_min_{mappa_i}}{RPM_{mappa_i}} * \frac{RPM - RPM\_min_{mappa_i}}{RPM\_max_{mappa_i} - RPM\_min_{mappa_i}}$$

$$Coeff_{RPM} = \frac{RPM\_min_{mappa_i} - RPM_{mappa_i}}{RPM_{mappa_i}} + \frac{RPM - RPM\_min_{mappa_i}}{RPM_{mappa_i}}$$

$$Coeff_{RPM} = \frac{RPM - RPM_{mappa_i}}{RPM_{mappa_i}}$$

Si giunge quindi infine a dei semplici indici correttivi, che, sebbene siano basati su ipotesi non molto robuste, consentono di tenere in conto lo scostamento dalle condizioni di riferimento. Tenendo conto del fatto che la correzione viene apportata sullo spettro da salvare, essa deve essere di segno opposto alla differenza che la genera:

$$Coeff_{RPM} = \frac{RPM_{mappa_i} - RPM}{RPM_{mappa_i}}$$

$$Coeff_{Alpha} = \frac{Alpha_{mappa_i} - Alpha}{Alpha_{mappa_i}}$$

Quando l'utente conclude manualmente la costruzione della mappa o quando ne impone il salvataggio, l'array di cluster **MAP_clust** viene salvato in memoria in formato XML e può essere caricato al nuovo avvio del software.

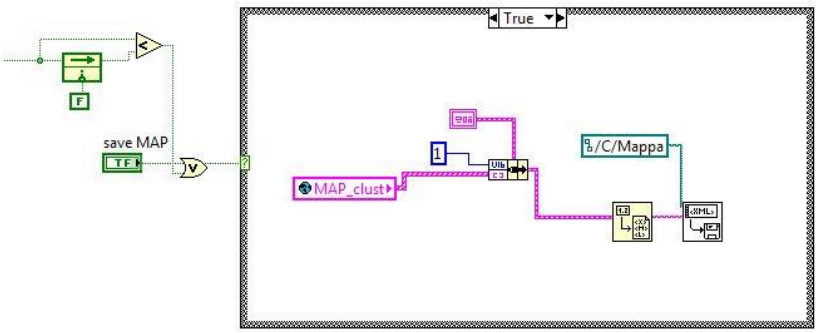


Figura 72: salvataggio in formato XML della mappa

3.4.3 ***Trendindex***

Il subVI Trend Index costruisce l'indice di degrado ed attraverso il confronto con delle soglie preimpostate fornisce in uscita eventuali allarmi.

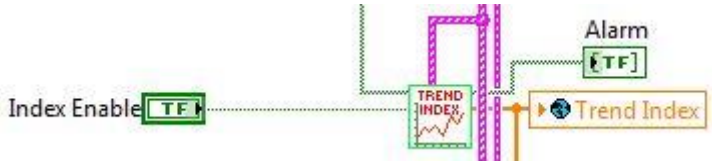


Figura 73: VI Trend Index

La determinazione della condizione di funzionamento e l'applicazione delle correzioni è inversa, ma identica a quanto visto per la costruzione della mappa, che in questo caso viene utilizzata come riferimento per il confronto con lo spettro di vibrazione attuale. L'indice viene calcolato sulla base dei primi 250 elementi esclusi i primi 2 componenti (componenti media ed a bassissima frequenza):

$$Trendindex = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{FFT\ data_i}{Map_i * (1 + Coeff_{Alpha} + Coeff_{RPM})} - 1\right|$$

I trendindex così calcolati vengono poi confrontati con le soglie impostate per i rispettivi canali.

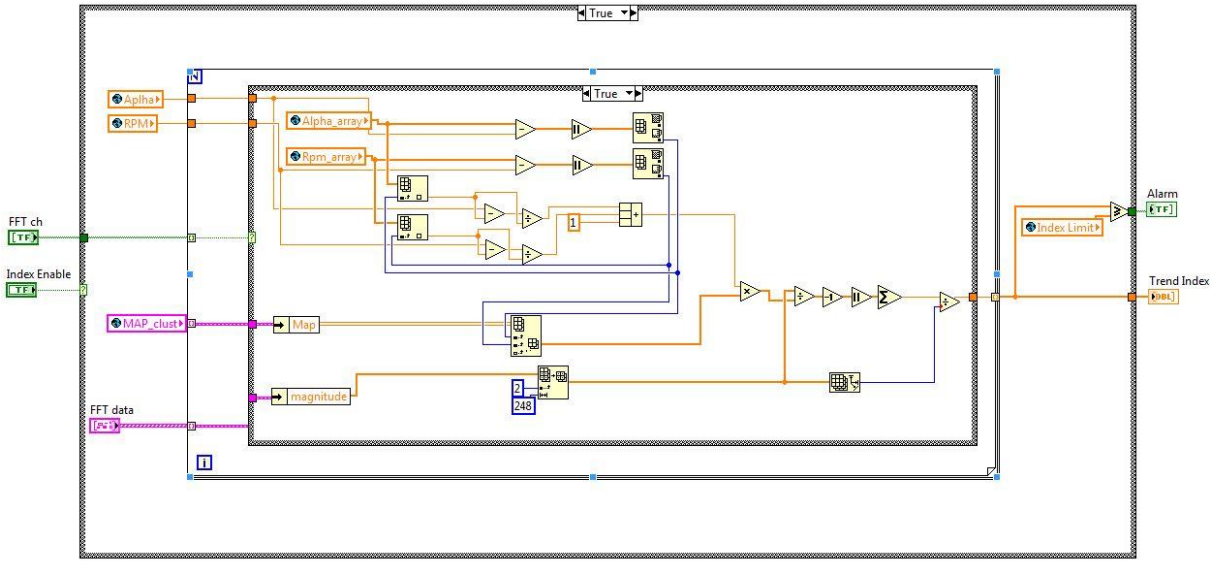


Figura 74: codice di calcolo del trendindex

3.5 TEST DEL SOFTWARE

La versione definitiva del software è stata testata nelle sue funzionalità utilizzando un simulatore in grado di generare delle uscite analogiche basate sulla lettura di un file txt. La simulazione è stata svolta utilizzando una reale registrazione durante un intero giro di pista (circa 95s) dei seguenti parametri:

- Segnale ruota fonica
- Segnale 0-10V della richiesta di coppia del pilota (manopola acceleratore)
- Accelerometro posto sul basamento motore
- Accelerometro posto sulla testata orizzontale
- Accelerometro posto sulla testata verticale

Il file è stato eseguito in loop ed ha dato modo di simulare l'acquisizione dei segnali come se Miracle^2 fosse collegato direttamente ai cablaggi dei sensori installati sul motore durante il run di durata. Questo ha dato modo di verificare in particolare le funzioni di costruzione della mappa e di calcolo del trendindex, cosa non possibile collegando gli ingressi di Miracle semplicemente ad un generatore di funzione. Sebbene la stabilità del software e le risorse impiegate per il calcolo siano sicuramente ottimizzabili, tutte le caratteristiche implementate sono risultate funzionanti.

3.5.1 *Interfaccia*

Il test tramite simulatore ha consentito anche di approfondire il funzionamento dell'interfaccia, attualmente a carico del processore real-time, che consente le seguenti funzionalità/impostazioni attraverso i rispettivi controlli/indicatori:

- **Save Mode**: selezione della modalità di salvataggio: manuale o con counter
- **REC**: attivazione e visualizzazione dello stato di salvataggio
- **SAVE selection**: cluster che consente la selezione dei canali e del tipo di dati da salvare oltre a gestire l'impostazione del trigger sui canali
- **save MAP**: salvataggio manuale della mappa
- **MAP build**: abilitazione della costruzione della mappa
- **time stamp**: data e ora correnti
- **Spectral Analysis** (pulsante): abilitazione dell'analisi spettrale

- **Spectral Analysis**: selezione del tipo di analisi spettrale da eseguire: di ampiezza o di potenza.
- **export mode**: in caso di analisi in potenza è possibile scegliere fra PSD e Power Spectrum
- **Freq/Order**: selezione della visualizzazione dello spettro in frequenza o come spettro degli ordini
- **FFT ch**: vettore di booleani che consente di attivare l'analisi spettrale per i primi 6 canali
- **Index Enable**: controllo per l'avvio del calcolo del trendindex
- **Buffer FFT**: indicatore che riporta la lunghezza del buffer (per il singolo canale) su cui viene eseguita l'analisi spettrale. In caso di corretto funzionamento del FIFO loop, per 50ms ad 80kHz il valore deve essere fisso a 4000 campioni.
- **delay Sample-FFT**: indicatore che riporta il ritardo fra campionamento e pubblicazione dei risultati dell'analisi.
- **CAN read**: eventuale indicatore per la visualizzazione dei dati in lettura dalla CAN
- **TWGRDM**: indicatore che riporta l'attuale valore di coppia richiesta
- **RPM**: indicatore che riporta l'attuale numero di giri del motore
- **Grafici**: aggiornati ogni 250ms da un apposito loop

    - Segnali su base tempo
    - Spettri
    - Andamento del trendindex

Sono inoltre presenti in basso a sinistra degli indicatori che riportano informazioni relative alle iterazioni dei loop utilizzate per il debugging del software.

Le impostazioni dei parametri di funzionamento (T_loop, n ch ecc), dei filtri e dei settaggi per i canali sono invece implementati in un cluster di controlli integrato con il cluster dei settaggi di OBI.

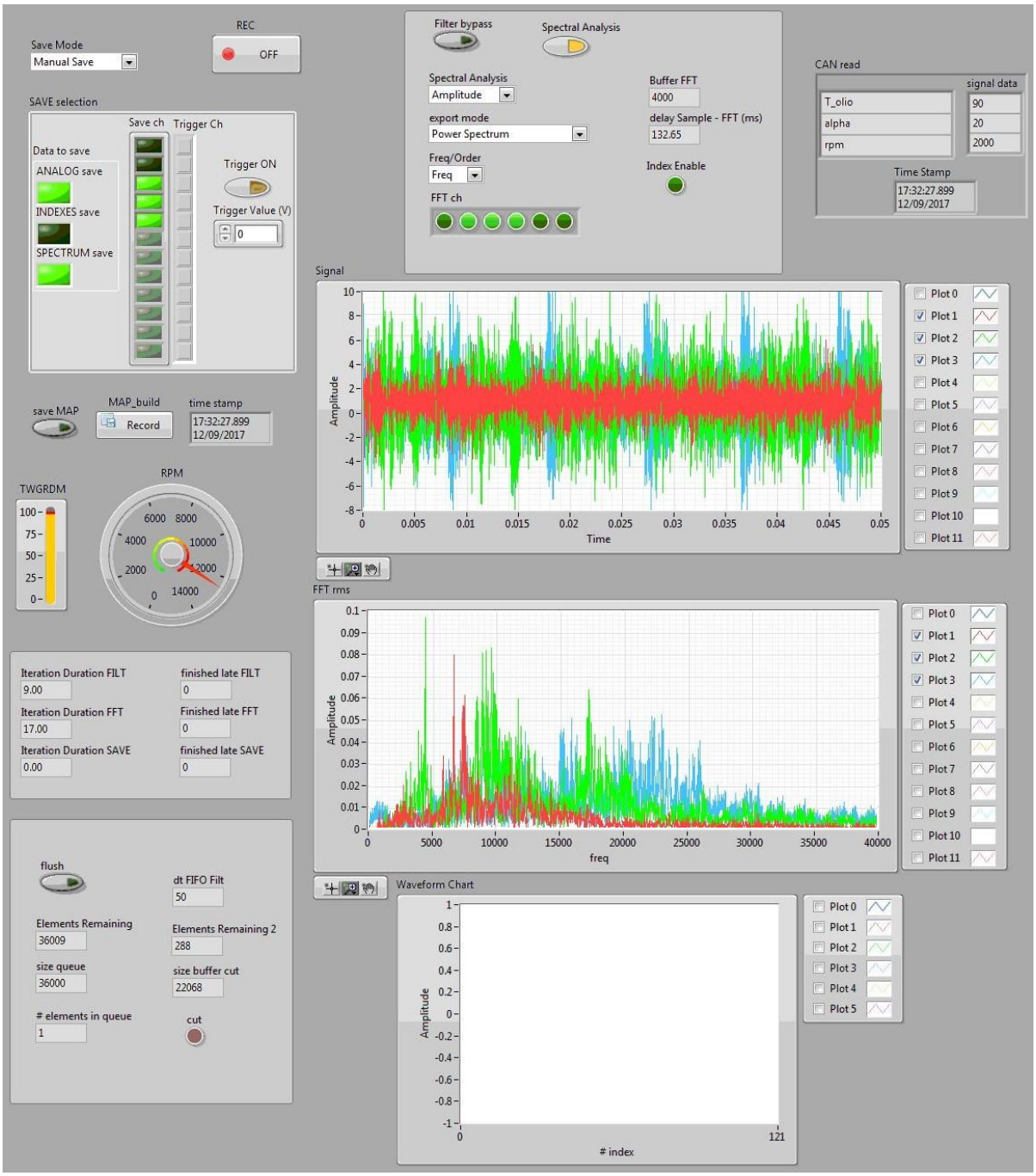


Figura 75: interfaccia Co-Mo durante il funzionamento in simulazione

3.6 SVILUPPI FUTURI

Dal punto di vista generale il software può essere sicuramente alleggerito e rivisto per adattarlo in modo più specifico alle esigenze del condition monitoring, liberando risorse dedicabili all'analisi del segnale ed eliminando controlli/variabili/indicatori presenti su OBI che sul software Co-Mo non sono necessari. Più nello specifico sono invece pianificabili degli interventi sostanziali sui tre livelli del codice.

FPGA

- Maggiore separazione fra la linea di flusso che porta all'analisi spettrale e quella del calcolo degli indici sintetici. Una maggiore distinzione può consentire l'utilizzo di filtri e frequenze di campionamento più specifici per le rispettive esigenze.
- Implementazione dell'FFT nell'FPGA, liberando notevoli risorse sul real-time che al momento è al limite come potenza di calcolo utilizzata.
- Implementazione del calcolo di indici sintetici (RMS, picco ecc.) su finestre angolari diverse da quella di combustione per valutare lo stato di integrità di componenti meccanici. È pensabile ad esempio valutare lo stato delle valvole osservando l'andamento del segnale accelerometrico nell'intorno della loro chiusura.
- Convertire il campionamento da base tempo a base angolo per sfruttare lo spettro degli ordini nel calcolo del trendindex.

Real-time

- Automatizzazione della procedura della costruzione della mappa
- Implementazione di funzionalità di salvataggio intelligente in caso di rilevamento di malfunzionamenti sul motore
- Verificare ed adattare il funzionamento della comunicazione CAN ed Ethernet

Host

- Costruzione dell’interfaccia di controllo e visualizzazione su PC host, lasciando su real-time solamente il minimo indispensabile per il funzionamento in modalità stand alone del sistema.
- Aggiunta di funzionalità di monitoraggio dei componenti basate sullo spettro

Capitolo 4
CAMPAGNA SPERIMENTALE E RISULTATI

In affiancamento alla costruzione del software è stata svolta una campagna sperimentale volta al raccoglimento del maggior numero di dati possibili da analizzare offline per simulare le migliori strategie di diagnosi. L'attività ha riguardato l'utilizzo di accelerometri per il monitoraggio del comportamento vibrazionale del motore dal rodaggio fino allo smontaggio, in particolare la raccolta dati più significativa si è avuta durante due run di durata consecutivi, in cui sono stati simulati circa una quarantina di giri di pista ognuno. L'obiettivo principale di questo capitolo è identificare un modo efficace per costruire indici (trendindex), basati sulle differenze spettrali fra il funzionamento attuale e la baseline, rilevata a motore nuovo, significativi del degrado del motore.

4.1 SENSORI UTILIZZATI E POSIZIONE

I sensori utilizzati, come visto in precedenza, sono accelerometri, ma ne sono stati provati diversi con caratteristiche fra loro piuttosto differenti. Sono stati confrontati sensori piezoelettrici di due differenti famiglie:

IEPE (Integrated Electronics PiezoElectric) o **ICP** (Integrated Circuit Piezoelectric):
Includono un amplificatore di carica integrato e sia il segnale che l'alimentazione passano attraverso un cavo coassiale. Rispetto ad un sensore in carica, che richiede un amplificatore esterno, vanno alimentati, ma garantiscono un'ottima robustezza del segnale con cablaggi semplici. Il sensore deve essere alimentato a corrente costante da 2 a 20 mA fra 18 e 28V. La corrente viene mantenuta costante da un diodo o da un equivalente circuito, mentre il bias di tensione viene rimosso dalla misura grazie alla presenza di un condensatore. I vantaggi di un sensore ICP rispetto ad un normale sensore senza circuito integrato di pre-amplificazione sono principalmente la sensibilità sulla tensione di uscita costante, indipendente dalla lunghezza e dalla

capacità del cavo, e l'uscita a bassa impedenza, che permette la trasmissione del segnale, virtualmente senza perdite, su cavi anche molto lunghi ed in ambienti rumorosi. Con questo sensore non è possibile rilevare accelerazioni continue.

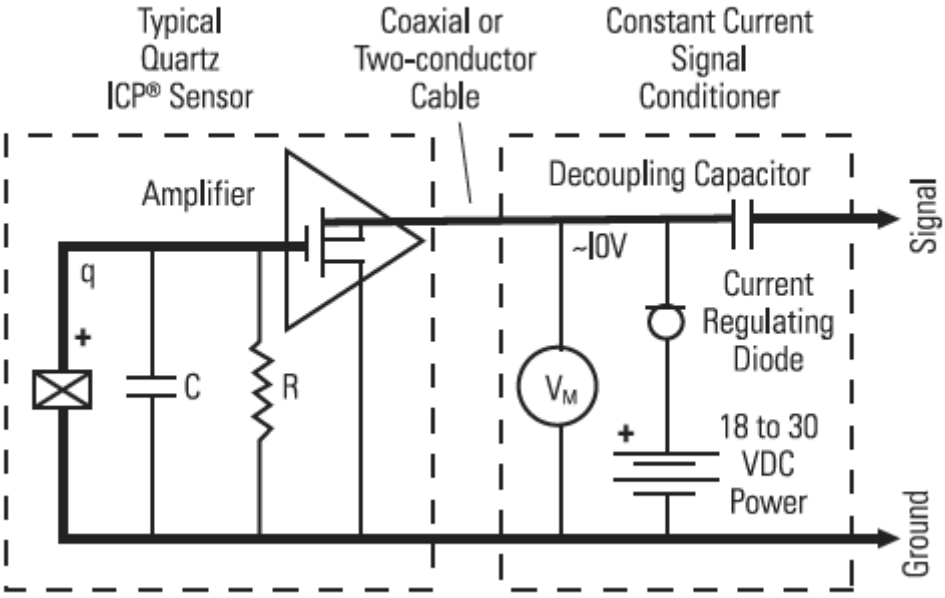


Figura 76: Tipico schema elettrico di un sensore IEPE e del modulo dedicato all'alimentazione

L'accelerometro a dispozizione era un IMI 607M126 con le seguenti caratteristiche:

Sensibilità (±20%): 10 mV/g

Range di misura: ±500g

Range di frequenze: 0,5 – 10.000 Hz

Frequenza di Risonanza: 25kHz

Figura 77:IMI 607M126

Knock Accelerometer:

La seconda classe di accelerometri utilizzati sono stati dei semplici accelerometri da knock, che, pur essendo sempre basati sul principio di funzionamento piezoelettrico non necessitano di nessun modulo di condizionamento esterno e possono essere acquisiti direttamente in tensione dalla centralina motore o nel nostro caso da Miracle[2]. Questi accelerometri sono dedicati alla rilevazione della detonazione e presentano un range di frequenze generalmente compreso da almeno 1kHz fino ad oltre 25-30kHz, ma il segnale di output in volt non è molto lineare con l'eccitazione meccanica.

L'accelerometro a disposizione per questa classe era un Bosch Knock Sensor KS-P:

Technical Specifications	
Mechanical Data	
Male thread (for cast)	M8x25
Male thread (for Al)	M8x30
Installation torque	20±5 Nm
Weight w/o wire	48 g
Protection	IP 54
Electrical Data	
Range of frequency	1 to 20 kHz
Sensitivity at 5 kHz	26 ± 8 mV/g
Max. sensitivity changing (life-time)	-17 %
Linearity between 5 to 15 kHz (from 5 kHz value)	-10 to 20 %
Linearity between 15 to 20 kHz (linear increasing with freq)	20 to 50 %
Main resonance frequency	> 25 kHz
Impedance	> 1 MΩ

Figura 78: Bosch knock sensor KS-P

Sebbene il sensore IEPE sia più adatto a misurare frequenze basse, per via delle complicazioni dovute al montaggio (la filettatura di fissaggio fornita non è metrica) ed alla necessità di utilizzare in modo continuativo un modulo per l'alimentazione ed il condizionamento, è stato scelto di utilizzare l'accelerometro da knock sia per l'applicazione condition monitoring, sia per il rilevamento della detonazione. La sensibilità ai primi ordini del motore è piuttosto bassa visto che il range di funzionamento è compreso fra 1kHz e 20kHz. Va tuttavia ricordato che il fine del rilevamento accelerometrico non è la corretta rappresentazione del fenomeno vibratorio, ma il calcolo di indici sintetici che non hanno di fatto alcun significato fisico direttamente riconducibile alla misurazione. Con la prospettiva di uno sviluppo futuro del sistema, l'estrema semplicità d'installazione ed il basso costo rendono preferibili questi sensori anche per l'applicazione on-board di serie; sono utilizzati peraltro con successo anche con il sistema delta-Analyser di Reilhofer KG.

L'installazione dell'accelerometro dedicato al rilevamento del rumore strutturale per l'applicazione condition monitoring è stata effettuata sul lato sinistro del basamento motore, in corrispondenza di una filettatura M8, installazione suggerita da precedenti esperienze con il sistema Reilhofer.

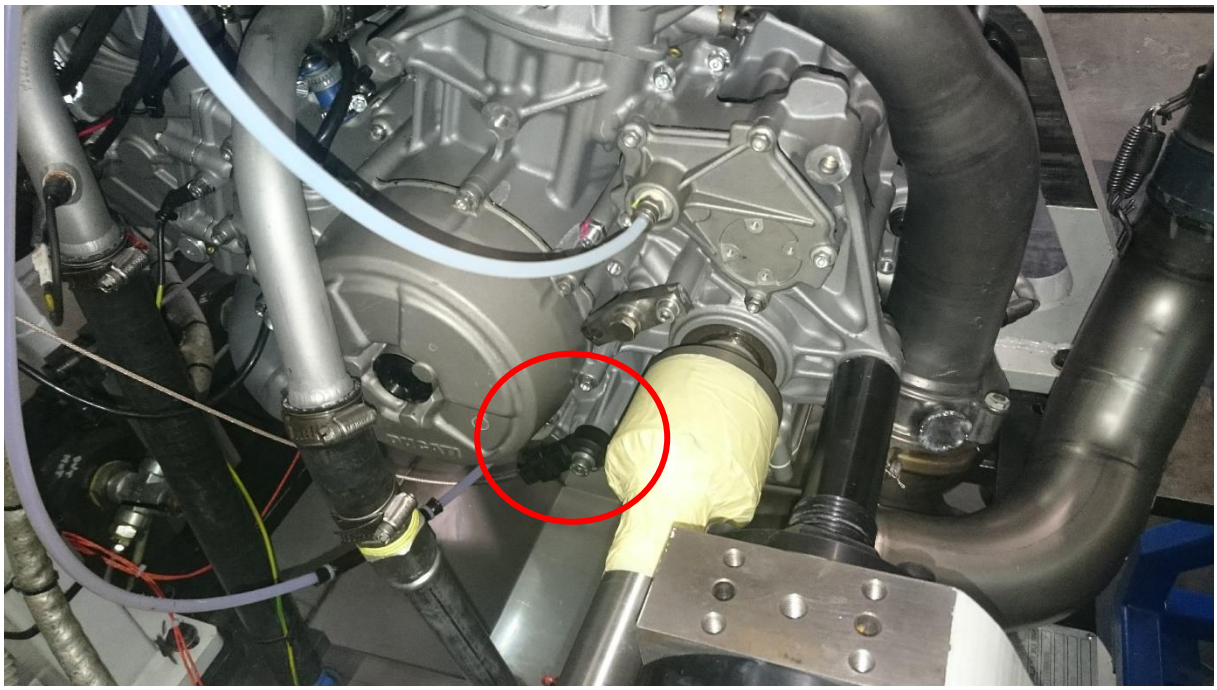


Figura 79: installazione accelerometro basamento

Per quanto riguarda l'installazione degli accelerometri per il rilevamento della detonazione si faccia riferimento al capitolo 5.

4.2 PIANO SPERIMENTALE E DI ANALISI

La campagna sperimentale ha previsto la registrazione del segnale proveniente dall'accelerometro posto sul basamento durante 2 test di durata, eseguiti subito dopo il rodaggio del motore nuovo, consentendo quindi di disporre di dati che potenzialmente possono mostrare il degrado dello stato meccanico. I due run registrati sono composti rispettivamente di 38 e 35 giri in cui viene simulato il funzionamento del motore come se si trovasse in pista installato sulla moto. Oltre al segnale accelerometrico sono stati registrati (o in alcuni casi ricostruiti) i parametri di banco che permettono di stabilire la condizione di funzionamento del motore. Sono stati quindi sincronizzati e inseriti in un unico file Matlab:

- Segnale accelerometrico basamento (variabile **bas**)
- Giri motore (variabile **RPM**)
- Richiesta di coppia impartita dal pilota tramite la manopola del gas (variabile **TWGRDM**: Twist Grip Demand)

In realtà la richiesta di coppia del pilota viene successivamente gestita dal controllo motore, che interviene modulando parametri come l'intervento del traction control o l'apertura diversificata delle farfalle per i singoli cilindri, ma nel complesso è la variabile che globalmente rappresenta meglio lo stato di carico del motore. Questi dati sono stati poi utilizzati come punto di partenza per **simulare offline**, tramite procedura Matlab, quanto dovrebbe essere svolto dal software Co-Mo eseguito su Miracle. Le fasi di simulazione sono essenzialmente due:

- **Costruzione della mappa di riferimento**: immediatamente dopo il rodaggio è prevista una fase in cui il sistema possa apprendere il comportamento vibrazionale del motore costruendosi una mappa di spettri di riferimento registrati in base a carico (TWGRDM) e regime motore (RPM).

- **Calcolo del trendindex**: durante il funzionamento del motore viene continuamente rilevato il comportamento vibratorio del motore e confrontato con il relativo spettro salvato in mappa per le stesse condizioni di funzionamento. A partire da questo confronto è possibile costruire degli indici, che, attraverso le differenze nelle componenti spettrali, rappresentino lo stato di salute del motore e siano in grado di evidenziare guasti meccanici.

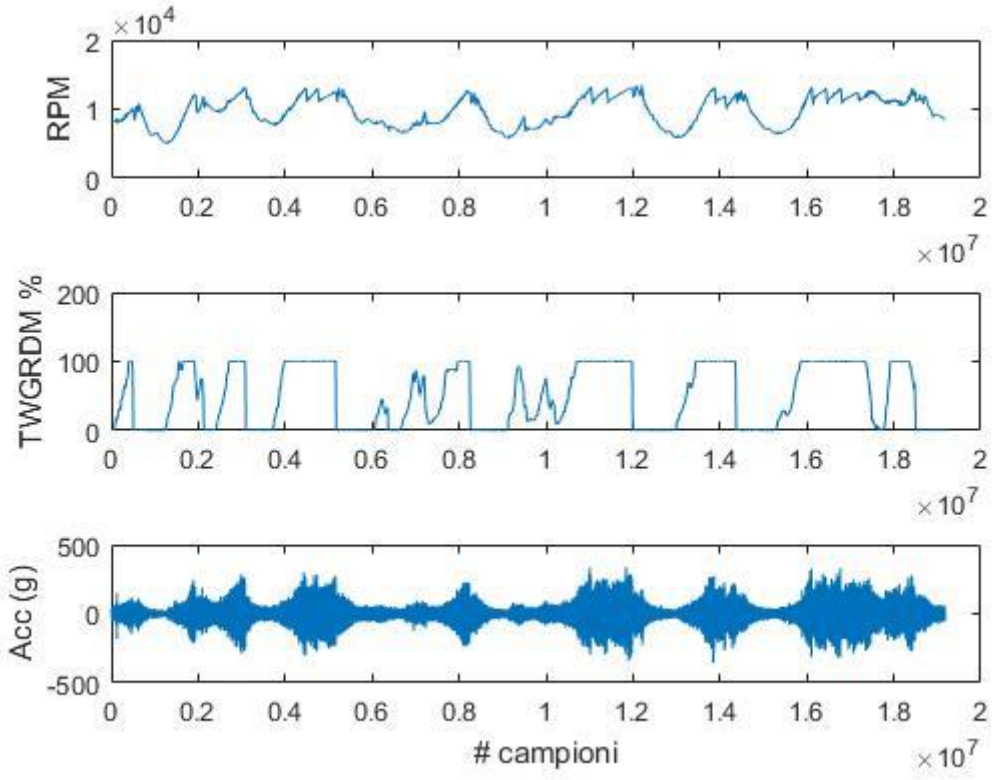


Figura 80: RPM, TWGRDM e bas durante un giro di pista

Così come mostrato sul software in linguaggio Labview, i segnali vengono acquisiti ed analizzati per buffer di campioni, corrispondenti a quello che sarebbe il T_loop. Su questi buffer, e quindi a scadenza temporale fissa pari a T_loop, viene ricavato lo spettro eseguendo l'FFT che può essere dedicato alla costruzione della mappa o al calcolo del trendindex. Nei sotto capitoli successivi verranno presentate le modalità di analisi e l'influenza dei principali parametri sui risultati.

4.3 COSTRUZIONE MAPPA

La costruzione della mappa avviene attraverso una procedura Matlab dedicata, che utilizza le informazioni di RPM e TWGRDM medi sul buffer analizzato per salvare lo spettro del segnale accelerometrico, rilevato sul medesimo buffer, in corrispondenza del corretto elemento di un array. In realtà il comportamento vibratorio del motore non dipenderà solamente da giri e carico, ma anche le temperature dei fluidi avranno un impatto sul segnale rilavato, così come altri parametri di funzionamento, ma essendo questa la prima analisi svolta sui dati si è preferito concentrare l'attenzione sui parametri su cui è possibile dare affidamento in modo più robusto. In futuro è possibile prevedere il salvataggio di mappe multidimensionali più complesse o semplicemente prevedere dei coefficienti correttivi basati su parametri di funzionamento che non siano RPM e TWGRDM.

I segnali sono stati campionati con OBI ad una frequenza di 200kHz, pertanto se si assume il T_loop = 50ms (10000 campioni), si otterranno degli spettri composti da 5000 elementi con informazioni in frequenza da 0 a 99980 Hz. Questo spettro andrà a costituire un singolo "valore" di riferimento della mappa, che nella sua interezza sarà quindi un array 3D in cui gli spettri sono indicizzati sulla base del corrispettivo valore di RPM e TWGRDM. Immaginando di aver impostato 10 step di carico e 20 di giri si otterrà una matrice 10x20x5000. Importante è la scelta delle griglie di RPM e TWGRDM con cui riempire la mappa, che potrebbero essere prese anche con step non lineari, specialmente per il valore del carico.

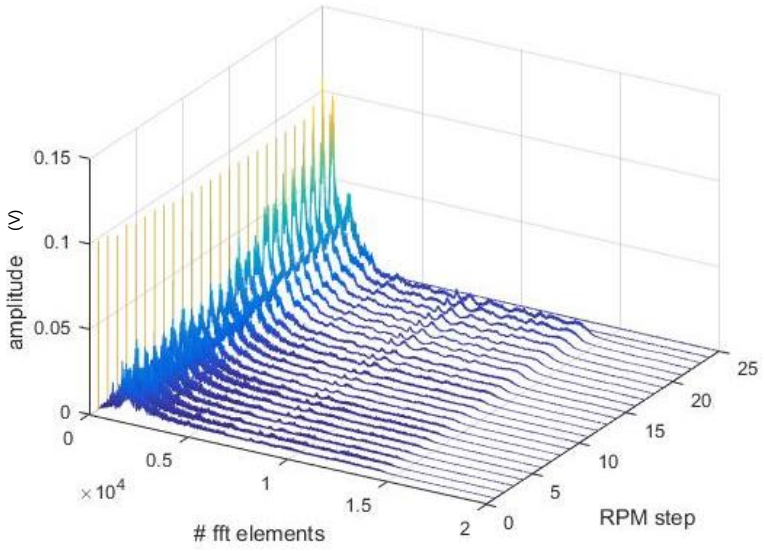


Figura 81: spettri salvati in mappa al variare degli RPM con TWGRDM del 100%

In realtà lo spettro non viene salvato così come calcolato, ma viene corretto utilizzando un coefficiente moltiplicativo che tiene conto dello scarto fra i reali valori di RPM e TWGRDM e quelli di mappa. Si utilizza questo approccio tramite coefficienti perché un'interpolazione su 5000 elementi sarebbe troppo onerosa computazionalmente per essere eseguita in real-time. Le formule impiegate sono le stesse viste nel capitolo 3, proprio per simulare le stesse identiche procedure, ma offline:

$$Spettro_{corr} = Spettro * (1 + Coeff_{RPM} + Coeff)$$

$$Coeff_{RPM} = \frac{RPM_{mappa} - RPM}{RPM_{mappa}}$$

$$Coeff_{Alpha} = \frac{Alpha_{mappa} - Alpha}{Alpha_{mappa}}$$

Il processo di costruzione della mappa dovrebbe essere idealmente eseguito in condizioni statiche, soffermandosi per diverso tempo sui punti di funzionamento stabiliti, mediando gli spettri ottenuti a pari condizioni, o al limite con rampe lente effettuate sia in salita che in discesa. Questo viene eseguito salvando il numero di "passaggi" nello stesso punto ed utilizzando questo dato come peso per l'aggiornamento della mappa:

$$Mappa_{new\,i} = Mappa_i + \frac{Spettro_i}{weigth_i}$$

Una sequenza di calibrazione pianificata, con lo stesso tempo di permanenza in condizioni statiche per ogni punto sarebbe l'ideale per produrre una mappa che includa il minimo effetto innescato dai transitori di giri e carico. Questi, se mediati con le altre condizioni e salvati, falsano il riferimento, che non è più rappresentativo del punto, e causano la produzione di trendindex più elevati di quelli attesi anche in condizioni perfettamente allineate con quelle di mappa.

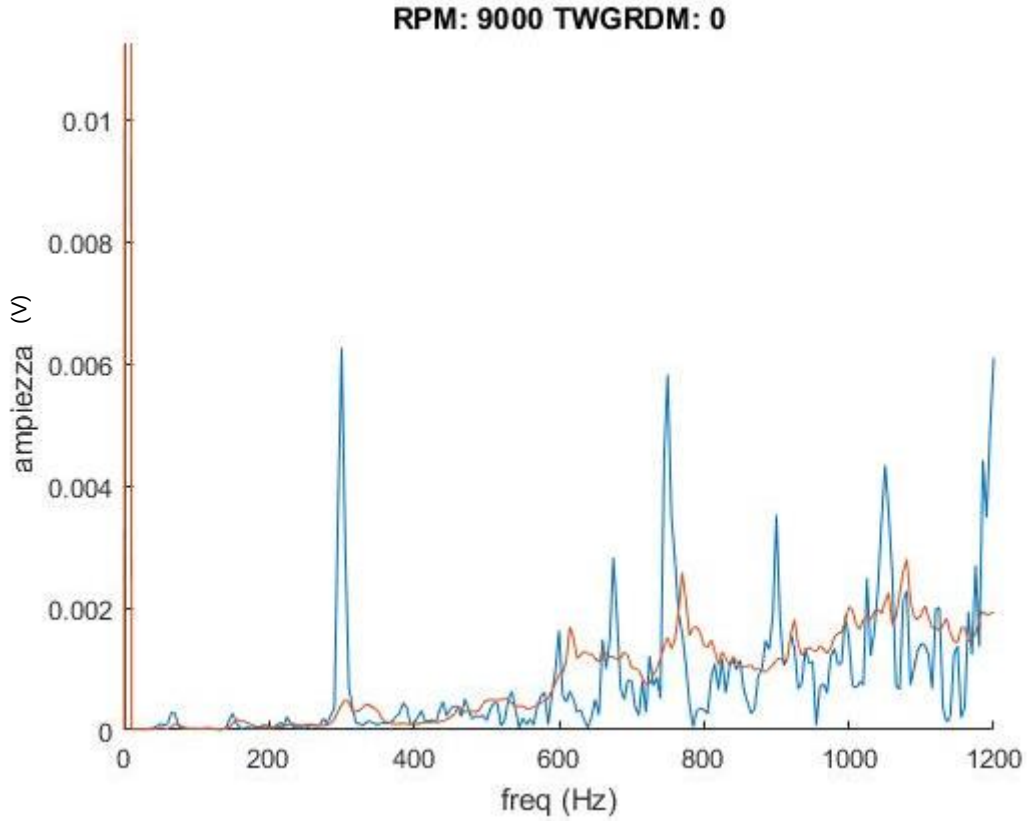


Figura 82: discrepanza marcata dei picchi fra mappa e spettro attuale, dovuta al sistematico salvataggio di transitori in mappa

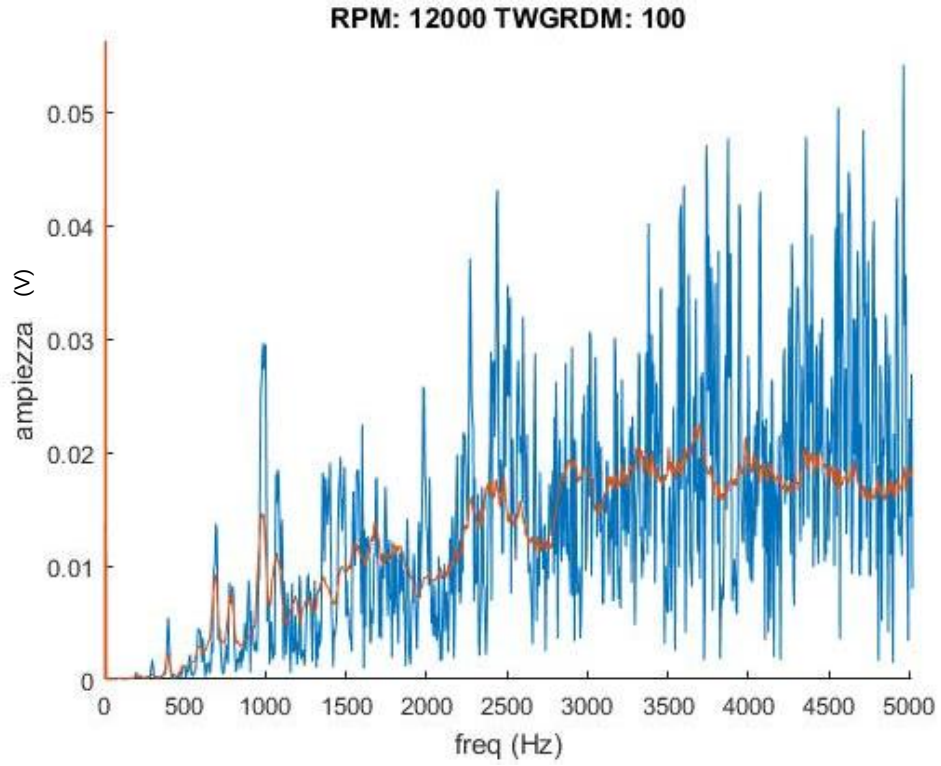


Figura 83: confronto fra uno spettro attuale ed il corrispettivo riferimento di mappa

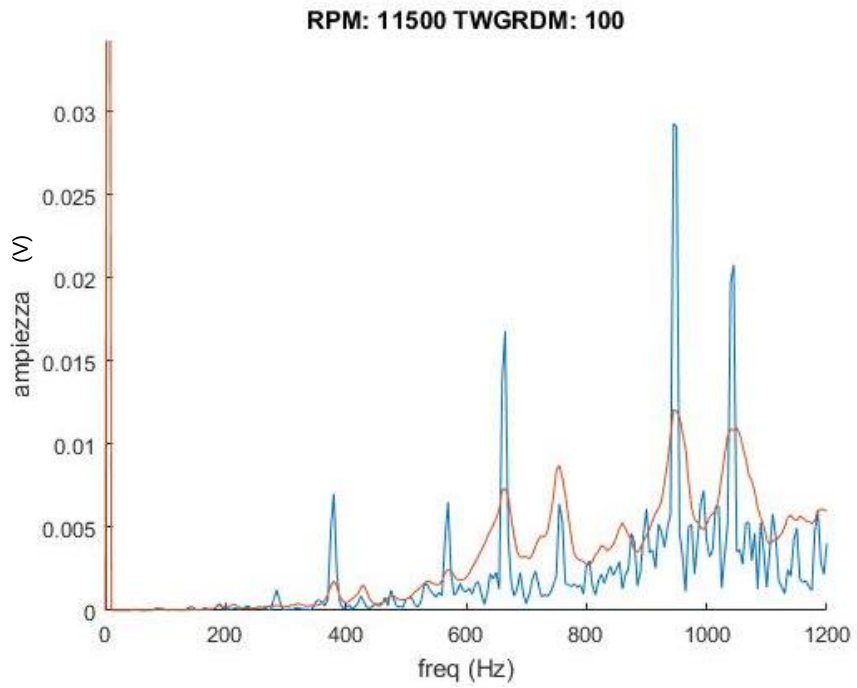


Figura 84: forte dispersione salvata in mappa, la cui possibile causa potrebbe essere la griglia degli RPM a maglie troppo ampie ed in situazione dinamica

Durante le prove eseguite non è stato possibile, per esigenze legate all'ordinario svolgimento delle attività di sala, operare una prova dedicata alla costruzione della mappa vibratoria, in quanto non era prevista alcuna attività di calibrazione. È stato perciò scelto di prendere a riferimento i primi 5 giri della durata per la costruzione della mappa, eseguiti quindi con il motore ancora "fresco". Parametri molto importanti per il calcolo di indici significativi sono:

- **Lunghezza buffer di analisi**: un buffer ampio consente una maggiore risoluzione spettrale, ma viene maggiormente influenzato dai transitori. Viceversa per il caso di buffer ridotti, che a causa della ridotta risoluzione può risultare più robusto alla dispersione.
- **Densità della griglia della mappa:** una griglia con maglie strette e opportunamente spaziate consente una maggiore precisione nella costruzione della mappa spettrale ed una migliore corrispondenza in condizioni operative durante il calcolo del trendindex. Tuttavia una griglia eccessivamente stretta, oltre a essere pesante da gestire da parte del processore e della RAM, può portare a spettri di riferimento costruiti con troppo poche ripetizioni ed al limite

alla presenza di “buchi”. L’introduzione dell’overlap sui buffer analizzati mitiga il numero di punti che rimangono scoperti nella mappa.

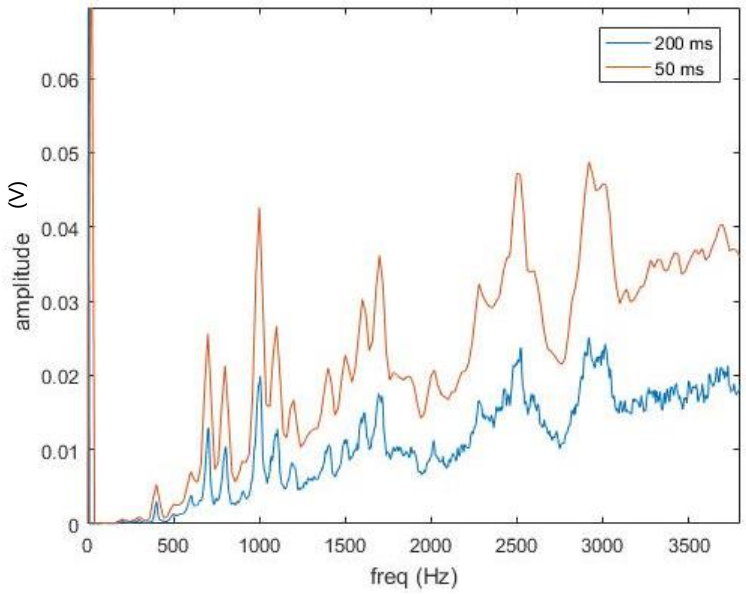


Figura 85: esempio della stessa condizione di funzionamento salvata in mappa con acquisizioni da 200ms e 50 ms

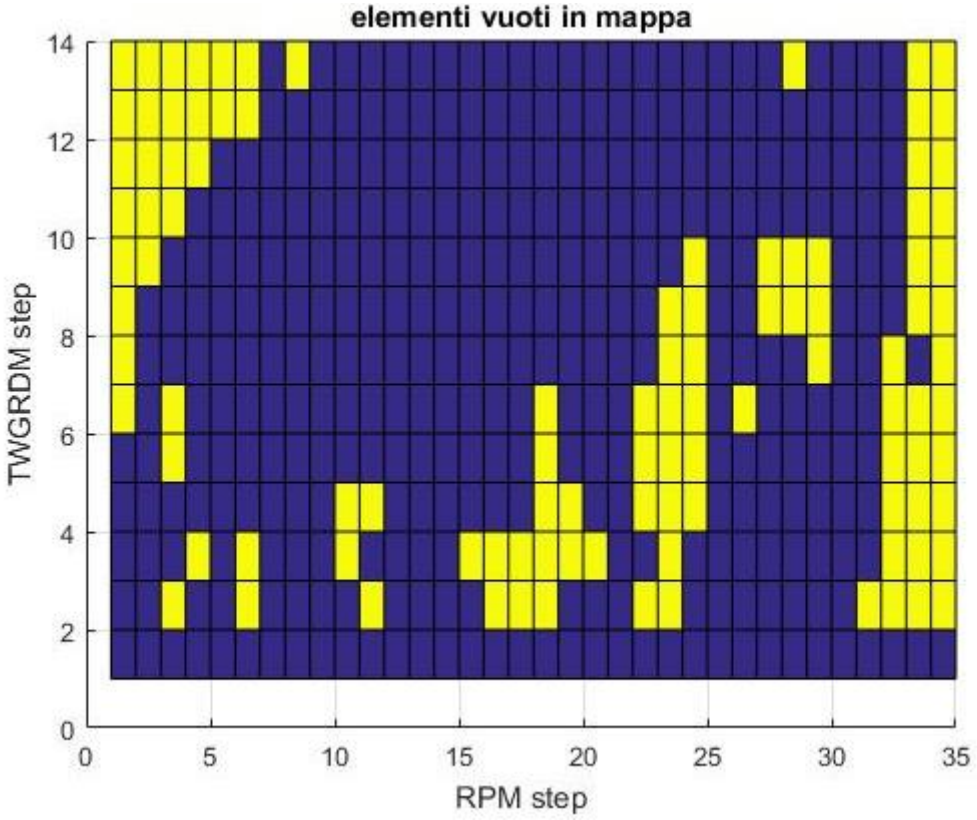


Figura 86: elementi vuoti in mappa 50 ms (125)

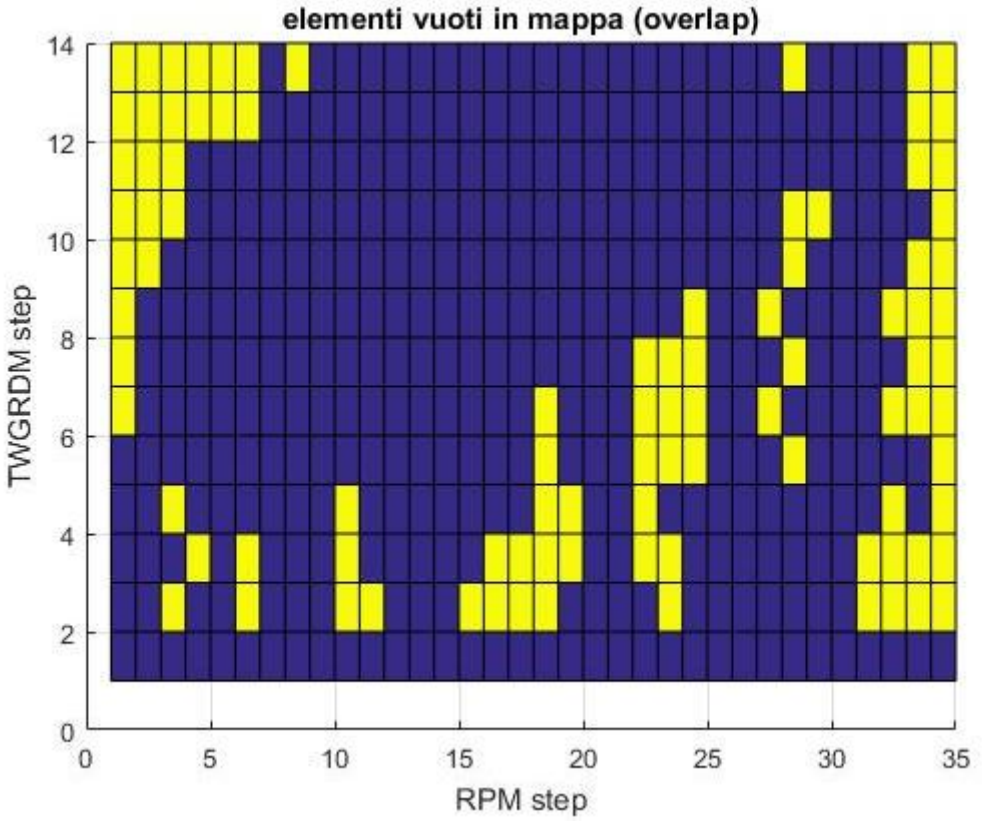


Figura 87: elementi vuoti in mappa 50 ms con overlap del 66% (116)

Durante la costruzione della mappa la scelta di un buffer di analisi corto, come potrebbe essere 50 ms, consente di utilizzare una griglia della mappa più stretta perché le variazioni di giri e carico saranno comunque modeste. Invece l'utilizzo di periodi d'acquisizione più lunghi, come ad esempio 200 ms, espone maggiormente al rischio di salvare transitori più lunghi, che al limite potrebbero avere un'ampiezza maggiore delle maglie della griglia stessa. D'altra parte durante la fase di calcolo dell'indice, per sfruttare al meglio la maggiore risoluzione spettrale è necessario disporre di maglie strette. In definitiva si può affermare che il corretto settaggio del tempo d'acquisizione e della griglia di punti che compongono la mappa dipende molto dalla procedura di calibrazione che è possibile attuare (procedura pianificata o dinamicamente) e dalle condizioni operative che il motore dovrà affrontare.

Per evitare il più possibile di includere in mappa le situazioni di transitorio più violento, come ad esempio le cambiate o le scalate, è comunque implementabile un controllo condizionale che preveda il salvataggio dei risultati dell'analisi spettrale solamente per

i buffer di campioni che siano sufficientemente “statici”. Un semplice esempio potrebbe essere basato sulle condizioni seguenti:

- $\max(RPM) - \min(RPM) < soglia_{rpm}$

  &

- $\max(TWGRDM) - \min(TWGRDM) < soglia_{tw}$

Logiche di questo tipo potrebbero essere usate anche per gestire la fase di calibrazione del sistema durante un’eventuale applicazione on-board. Le soglie andranno scelte opportunamente a seconda della lunghezza del buffer temporale di analisi e delle dimensioni delle maglie della mappa.

4.4 TRENDINDEX

L’obiettivo finale del sistema è quello di indicare all’utente o ad altri dispositivi di controllo quale sia l’attuale stato di salute del motore. Il modo più efficacie di fornire questa informazione è sicuramente la costruzione di uno o più indici sintetici, sui quali poter porre dei limiti di soglia per l’accettabilità. Come detto più volte nel corso dell’elaborato, l’idea è quella di confrontare in modo continuo lo spettro del buffer di dati campionato correntemente con lo spettro di riferimento salvato in mappa per le stesse condizioni. A partire dai due spettri l’intenzione è quella di produrre uno scalare il cui valore tenga conto delle differenze sulle singole bande spettrali e l’implementazione di base era stata pensata inizialmente come:

$$T_{index} = \sum_{i=1}^{N} |(Spettro\_ref_i - Spettro_i)|$$

Con N numero delle componenti spettrali

La sommatoria può essere anche eseguita solamente su certe componenti spettrali (e quindi frequenze), rendendo quindi possibile scegliere il range ottimale o porre

l’attenzione su particolari fenomeni. Il valore assoluto è necessario per impedire che differenze di segno opposto si annullino, inficiando la capacità di riconoscere due spettri sostanzialmente differenti.

L’indice così calcolato presenta tuttavia i seguenti difetti:

- A parità di differenze relative le componenti a maggiore ampiezza hanno un peso maggiore

- Non confrontabilità con altri indici calcolati su range di frequenze diversi

Per risolvere il secondo dei problemi si può eseguire una normalizzazione sul numero delle componenti della sommatoria:

$$T_{index} = \frac{1}{N}\sum_{i=1}^{N}|(Spettro\_ref_i - Spettro_i)|$$

Tuttavia la disparità di peso fra le componenti spettrali rimane e può avere un effetto molto negativo dal momento che non è assicurato che le componenti più ampie siano quelle più significative per l’analisi. In particolare l’utilizzo di un accelerometro molto poco sensibile alle frequenze più basse, come nel nostro caso, può produrre problemi sul calcolo visto che quelle di maggiore interesse sono probabilmente sono comprese al massimo fino a 4-5kHz (20ª - 25ª armonica di ciclo a 12000rpm)

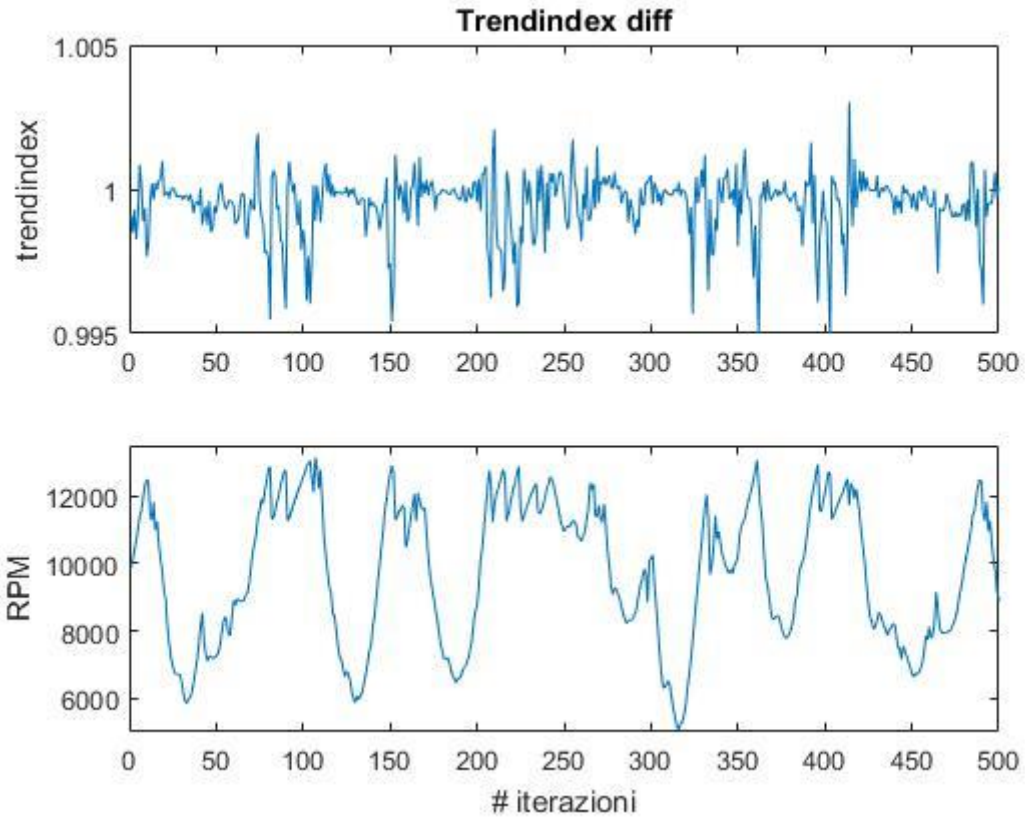


Figura 88:esempio di trendindex basato su differenze normalizzato nel corso di un giro

Per questo si è scelto di evolvere il calcolo dell'indice come segue:

$$T_{index} = \frac{1}{N}\sum_{i=1}^{N}\left|\left(\frac{Spettro_i}{Spettro\_ref_i} - 1\right)\right|$$

Questo indice, sebbene sia computazionalmente più pesante per la presenza delle divisioni, tiene conto delle differenze sulle linee spettrali in modo relativo sulle rispettive ampiezze ed inoltre è normalizzato sul numero delle componenti interessate calcolo.

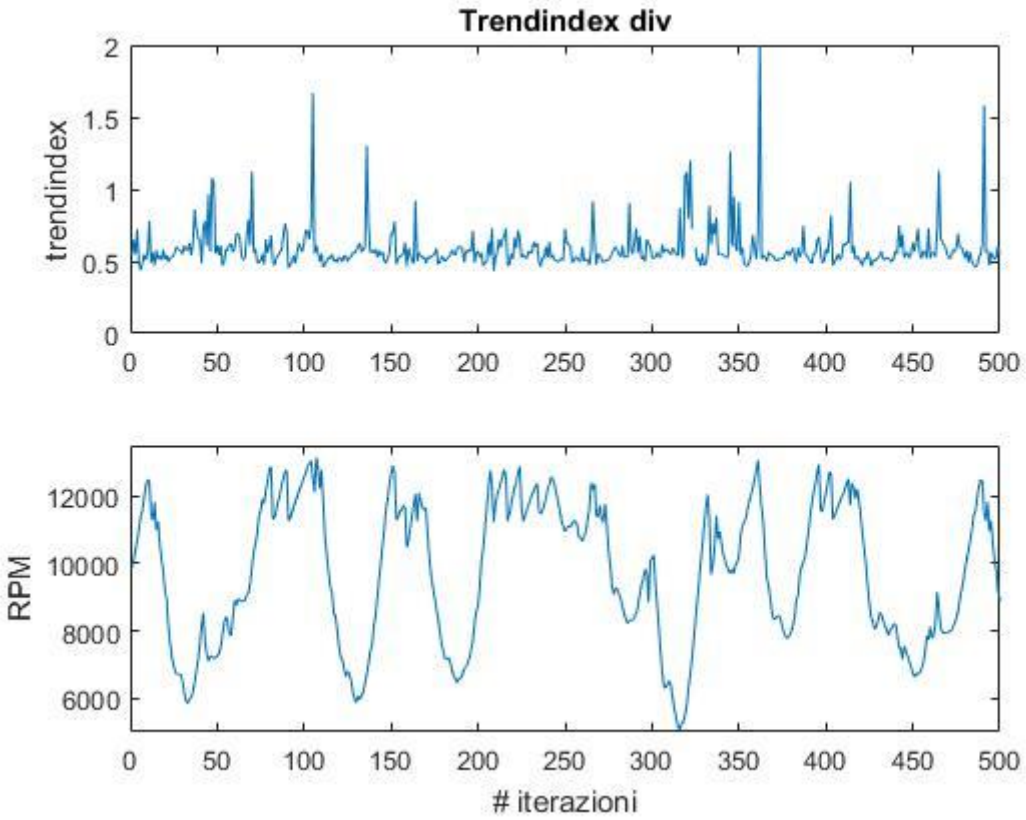


Figura 89:esempio di trendindex basato su divisioni nel corso di un giro

Dai due grafici che mostrano il trendindex in parallelo con gli RPM è possibile notare che, come prevedibile, durante cambiate e scalate si hanno dei picchi nell'indice, sia dovute alle vibrazioni indotte, sia alle discrepanze fra le attuazioni effettive e quelle apparenti (TWGRDM risulta massimo durante le cambiate anche se il motore non produce coppia positiva). Inoltre come accennato in precedenza è visibile che l'indice basato su rapporti è più sensibile alle variazioni.

È possibile pensare anche di introdurre delle tolleranze che tengano conto della naturale variabilità dello spettro a parità di condizioni di funzionamento. Visto che per il funzionamento real-time è impensabile calcolare ed utilizzare la deviazione standard per ogni componente spettrale su ogni elemento della mappa, come primo approccio è possibile pensare di implementare una fase di "calcolo delle tolleranze", immediatamente successiva alla costruzione della mappa, nei seguenti modi:

- **Tolleranza globale sul trendindex**:
  Costruzione di una matrice 2D con le stesse modalità dei riferimenti spettrali, in cui salvare il valore medio del trendindex calcolato per ogni condizione.

$$T_{index_{tol}} = T_{index} - tol$$

- **Tolleranza sulle linee spettrali**:
  Costruzione di una matrice 2D con le stesse modalità dei riferimenti spettrali, in cui salvare la media dello scostamento sulle singole componenti spettrali (trendindex diviso per il numero delle componenti spettrali).

$$T_{index_{tol}} = \frac{1}{N}\sum_{i=1}^{N}\left|\left(\frac{Spettro_i}{Spettro_{ref}{}_i} - 1\right)\right| * \left[\left|\frac{Spettro_i}{Spettro_{ref}{}_i} - 1\right| > tol_i\right]$$

Il primo metodo consente solo di sottrarre la tolleranza all'indice calcolato, senza alcun reale impatto sulla procedura di calcolo, ma si tratta di approccio piuttosto robusto. Nel secondo caso invece, a patto di accettare l'approssimazione che le differenze siano equamente distribuite su tutte le linee spettrali, è possibile modificare in modo abbastanza radicale il calcolo dell'indice, ammettendo alla sommatoria solamente gli elementi della divisione fra spettri che superano la tolleranza.

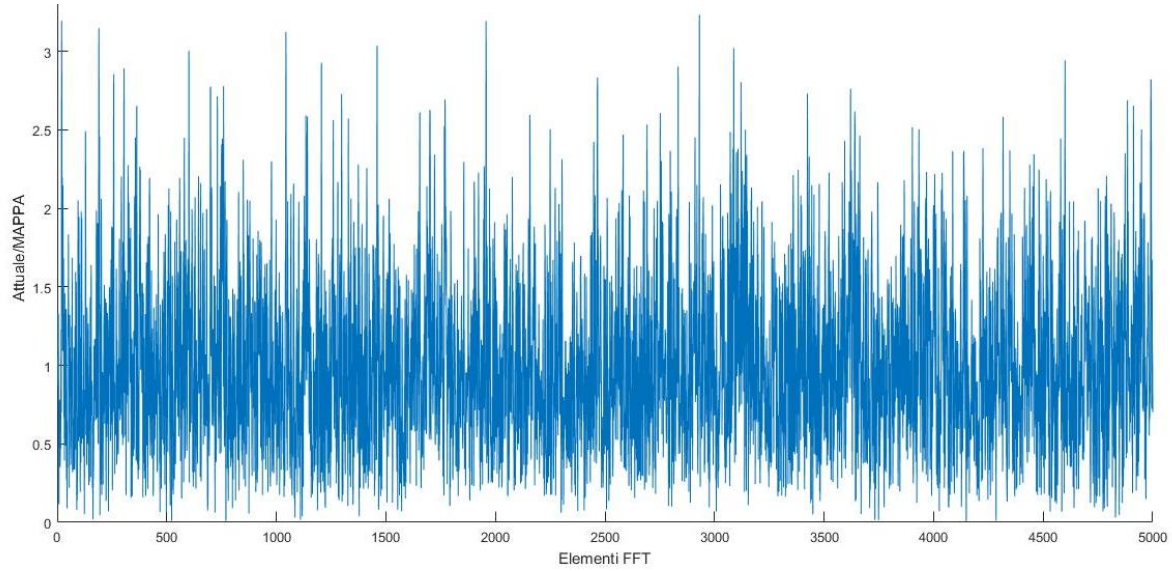


Figura 90: esempio di distribuzione degli scostamenti delle linee spettrali (attuale / mappa)

Per i seguenti confronti sull'intera vita del motore sono stati scelti i tre seguenti range di frequenze per il calcolo degli indici (La componente media e quelle ad essa prossime sono escluse dal calcolo):

- Da **25 Hz a 40 kHz**: tiene conto di praticamente tutto il contenuto in frequenza producibile dal motore (**trendindex 40k**)
- Da **25 Hz a 5 kHz**: esclude le frequenze oltre i 5 kHz per evitare di tenere in considerazione i fenomeni detonanti. (**trendindex 5k**)
- Da **25 Hz a 2 kHz**: valuta l'indice solamente sulle prime armoniche di ciclo. (**trendindex 2k**)

È ovviamente possibile scegliere range differenti e tenere conto di bande multiple o mirate in frequenza su particolari componenti (anche mobili con gli rpm), ma come prima analisi si è preferito mantenere l'approccio il più semplice possibile.

4.4.1 *Confronto sullo stesso giro di diverse impostazioni*

Come evidenziato nel paragrafo 4.3 la scelta dell'abbinamento fra buffer di analisi e griglia della mappa è un argomento delicato e piuttosto influente sui risultati, perciò è stato scelto di porlo come una costante e per tutte le prove successive la mappa è stata costruita su una griglia con step di 250 RPM e step di TWGRDM non lineari, più fitti vicino allo 0:

```
RPM_grid=linspace(5000,13500,(13500-5000)/250+1);

TWGRDM_grid=[0 2 5 8 12 18 25 35 45 55 65 75 85 95]
```

A causa della procedura di costruzione della mappa eseguita in condizioni dinamiche, nonostante l'utilizzo di un overlap del 66%, alcuni punti attraversati durante la prova in esame non hanno valori di riferimento registrati in matrice e sono presenti come NaN. L'indice fino a 40 kHz è meno sensibile alle differenze spettrali della parte meccanica perché ne media l'effetto su un numero di elementi molto ampio rappresentativi di anche di frequenze poco interessate dal rumore meccanico. Al contrario l'indice fino 2 kHz è probabilmente costruito su un range troppo restrittivo per essere rappresentativo dello stato di salute generale del motore, perciò è stato deciso di proseguire l'analisi concentrandosi su quello da 5kHz.

50 ms

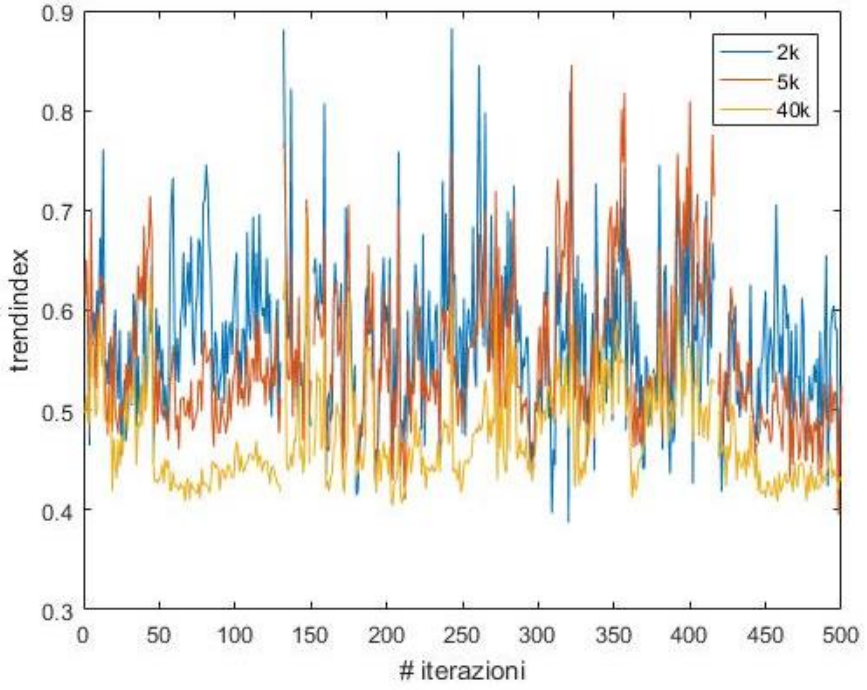


Figura 91: Confronto del trendindex calcolato sui diversi range durante un giro di pista. Buffer di analisi 50 ms

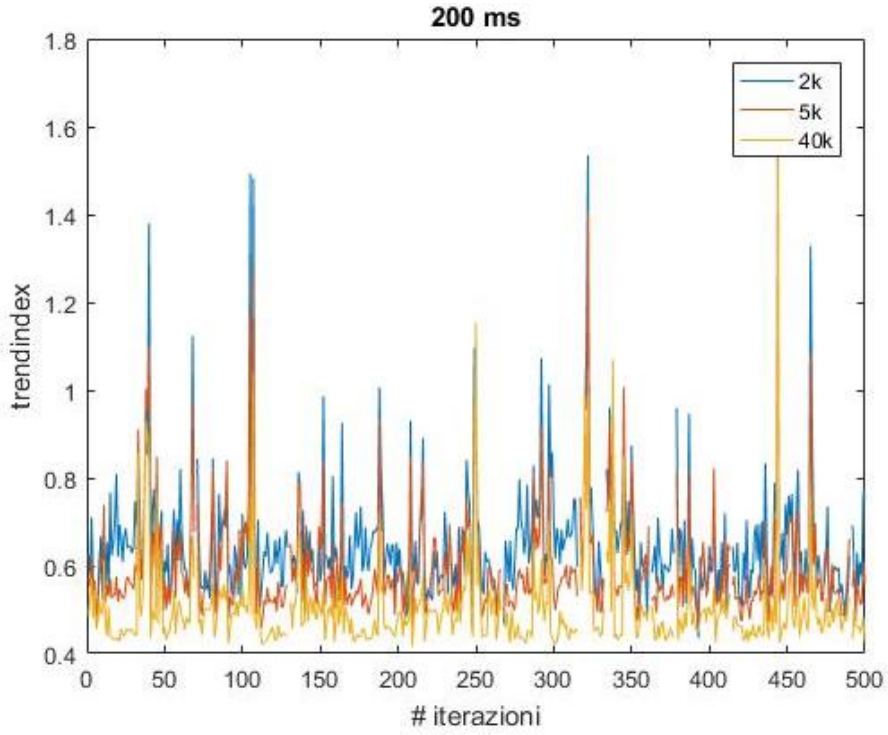


Figura 92: Confronto del trendindex calcolato sui diversi range durante un giro di pista. Buffer di analisi 200 ms

Per ovviare agli improvvisi aumenti del trendindex durante le cambiate è possibile applicare un filtro a valle del calcolo, che smorzi le frequenze più alte dell'indice. Questo approccio potrebbe compromettere la tempestività della rilevazione di rotture improvvise, per cui è comunque necessario mantenere attivo un controllo anche sulla rumorosità globale del motore (RMS). Di seguito è stato applicato un filtro mediano su 3 elementi (**filtro di Hampel**), che elimina gli outlier e gli spike di un solo elemento senza addolcire in modo eccessivo il segnale: l'elemento centrale della finestra viene sostituito con la mediana.

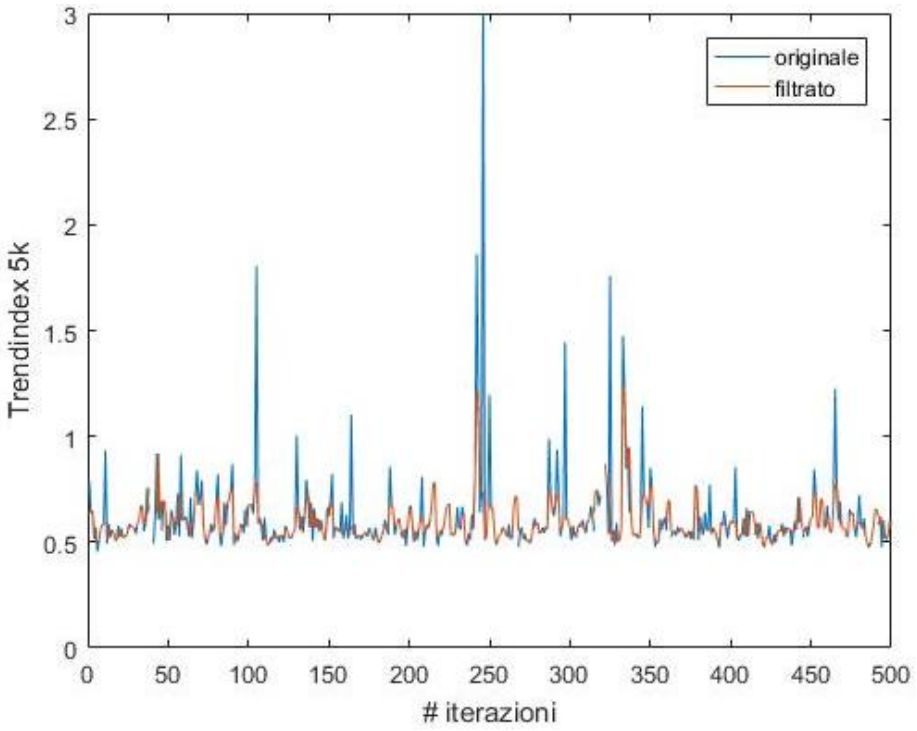


Figura 93: segnale filtrato con filtro di Hampel (buffer 200 ms)

Operativamente, durante il calcolo in real-time, potrebbe essere usato un mascheramento quando ci sono brusche variazioni di RPM: in questi casi, semplicemente il trendindex verrebbe calcolato, ma poi ignorato.

L'RMS del segnale accelerometrico durante un giro pista risulta estremamente correlato col numero di giri:

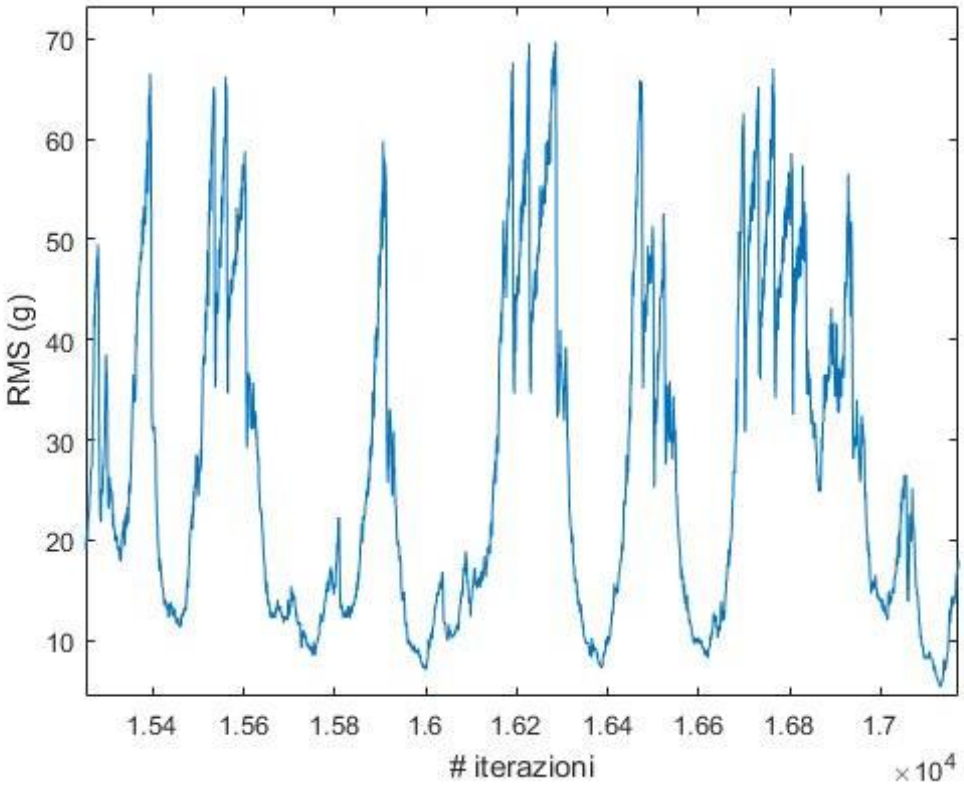


Figura 94: RMS durante un giro di pista

4.5 TRENDINDEX DURANTE LA VITA DEL MOTORE

Di seguito sono riportati i risultati della simulazione di calcolo del trendindex su tutti e 73 (38+35) i giri della prova di durata a cui è stato sottoposto il motore. Sebbene nelle procedure di calcolo siano stato inclusi più correttivi possibile per evitare l'influenza di errori sui dati, alcuni punti evidenziano degli indici anomali a causa di:

- Giunzioni di diverse registrazioni e seguente non perfetta corrispondenza fra i parametri di funzionamento reale e quelli ricostruiti sul giro per i primi campioni.
- Valori anomali dovuti ad errori di misura sull'accelerometro o sulla ruota fonica preposta al rilevamento dei giri motore

Va puntualizzato che il secondo run, quello da 35 giri, è stato interrotto a causa di un trafilamento d'acqua dentro al carter motore, dovuto alla rottura della guarnizione di testa.

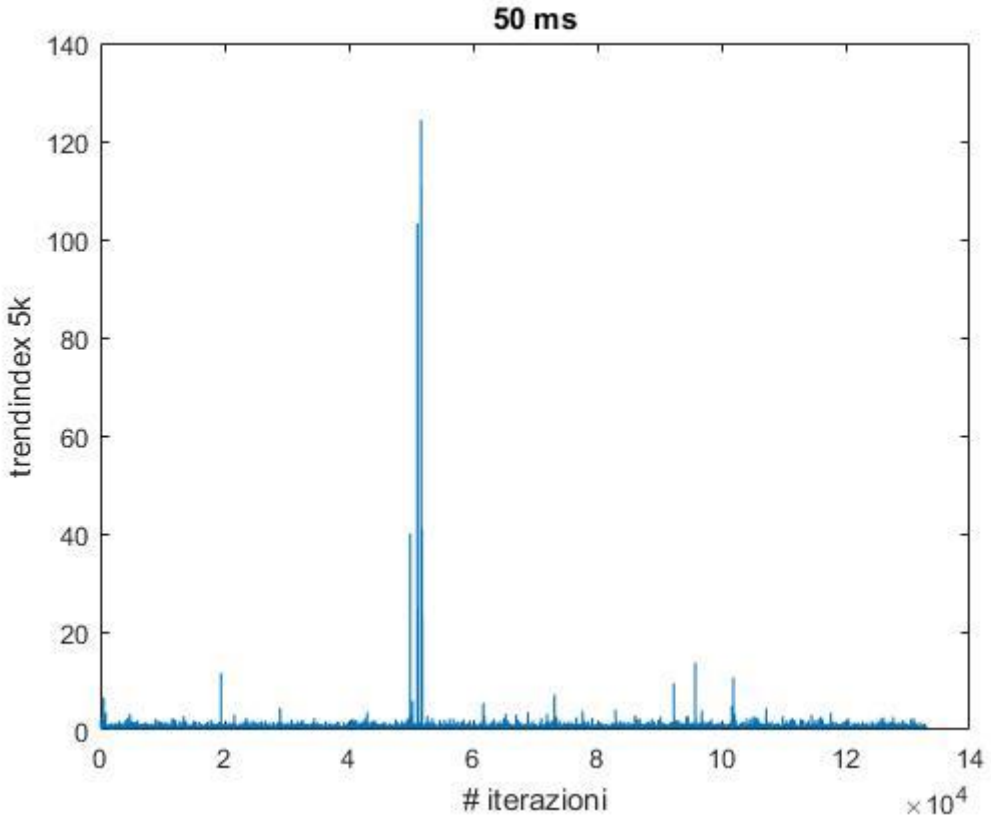


Figura 95: trendindex 50 ms

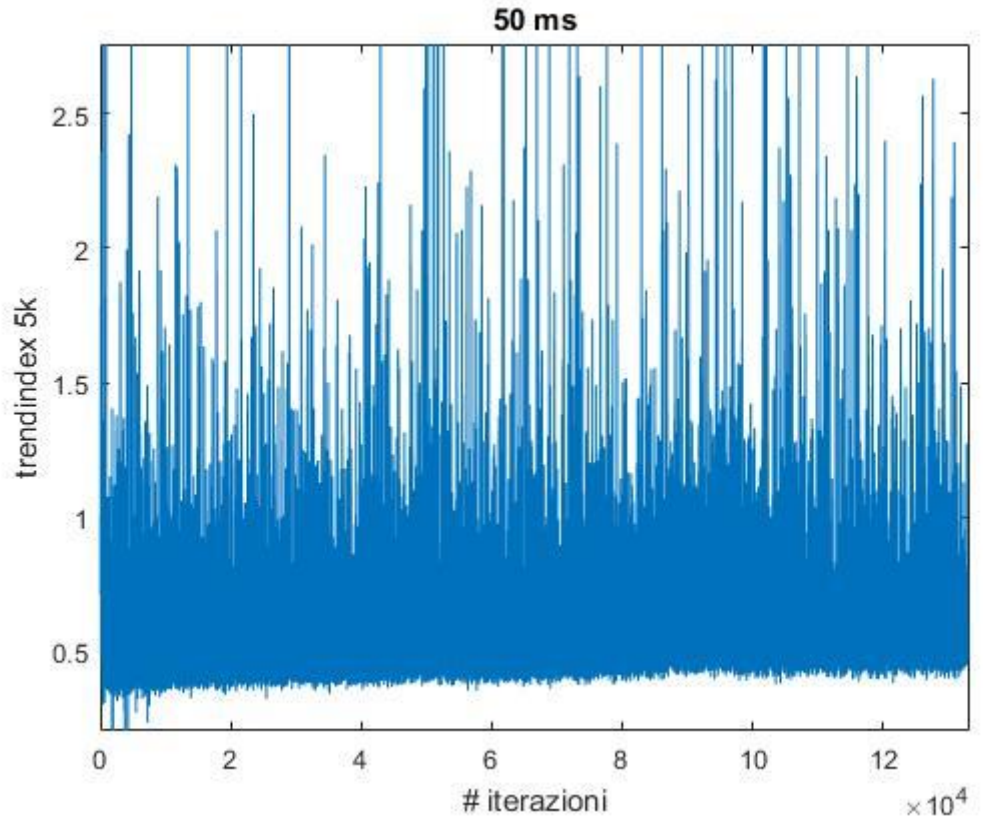


Figura 96: trendidex 50 ms zoom

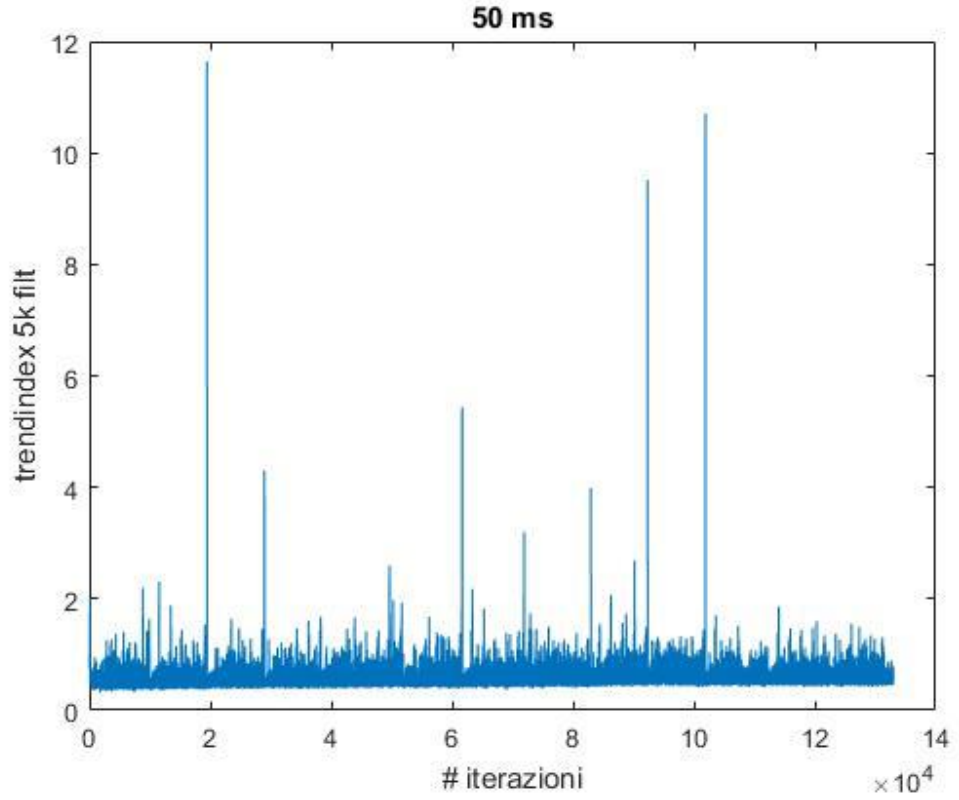


Figura 97: trendindex 50 ms con filtro di Hampel

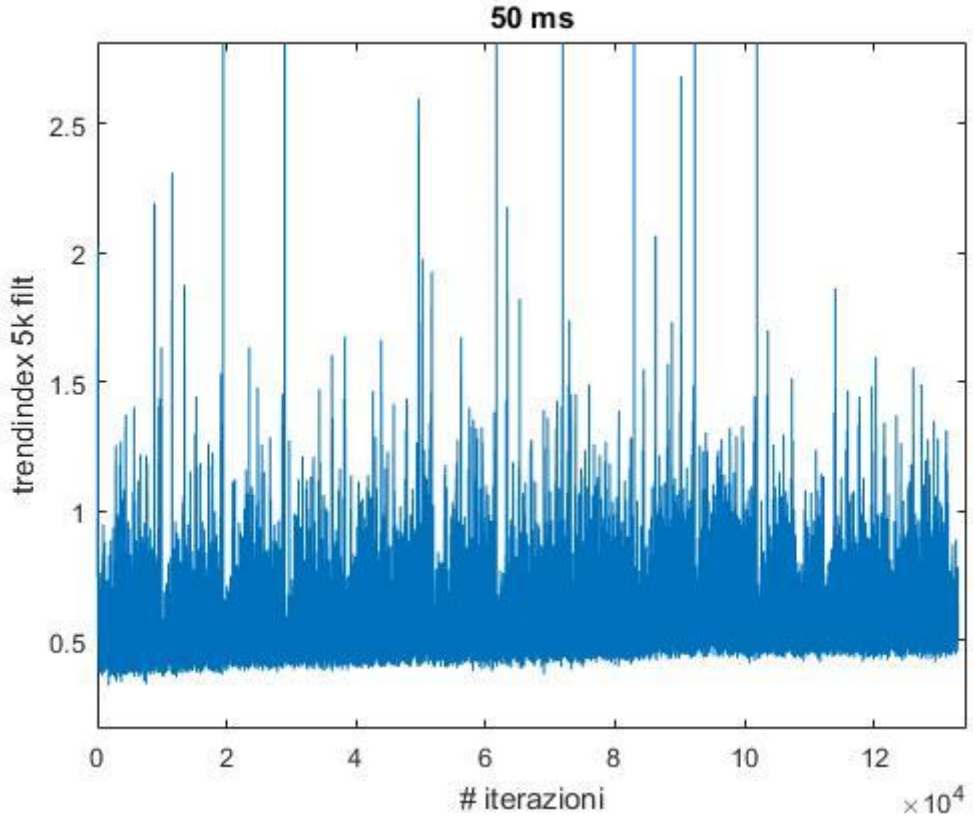


Figura 98: trendindex 50 ms con filtro di Hampel zoom

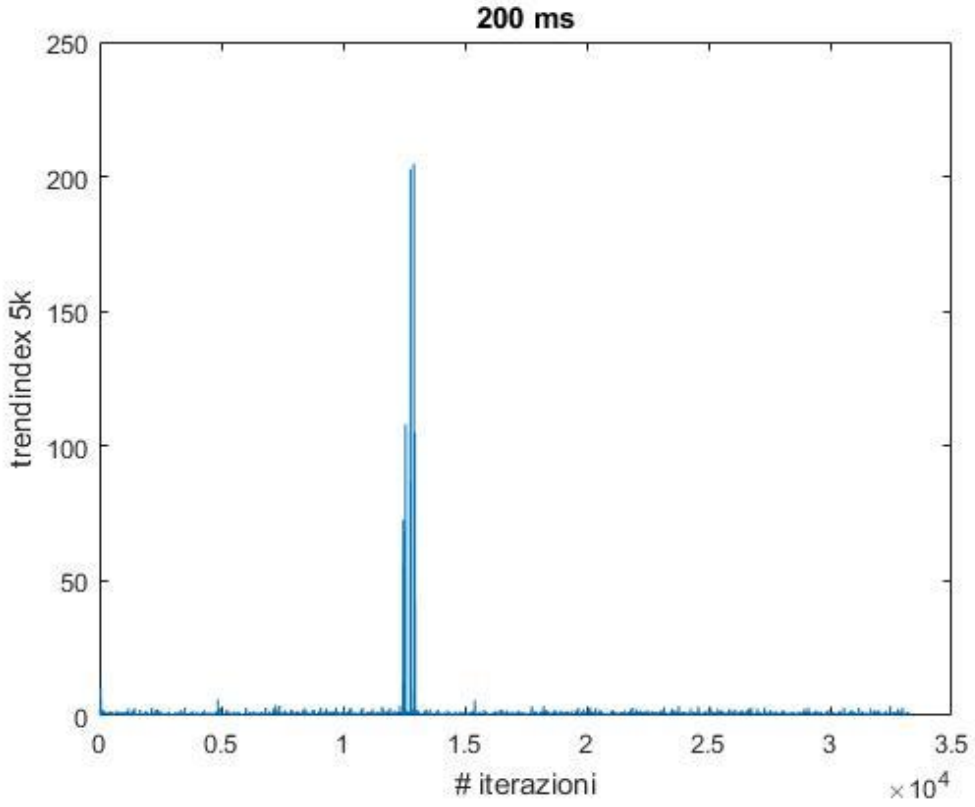


Figura 99: trendindex 200 ms

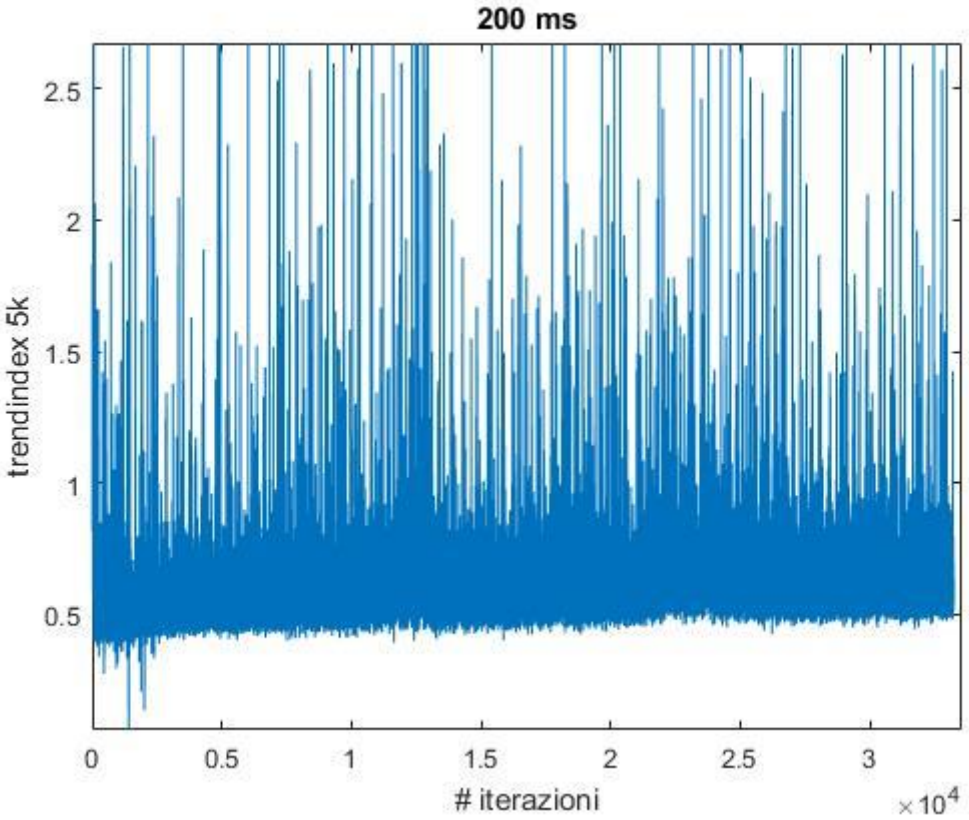


Figura 100: trendindex 200 ms zoom

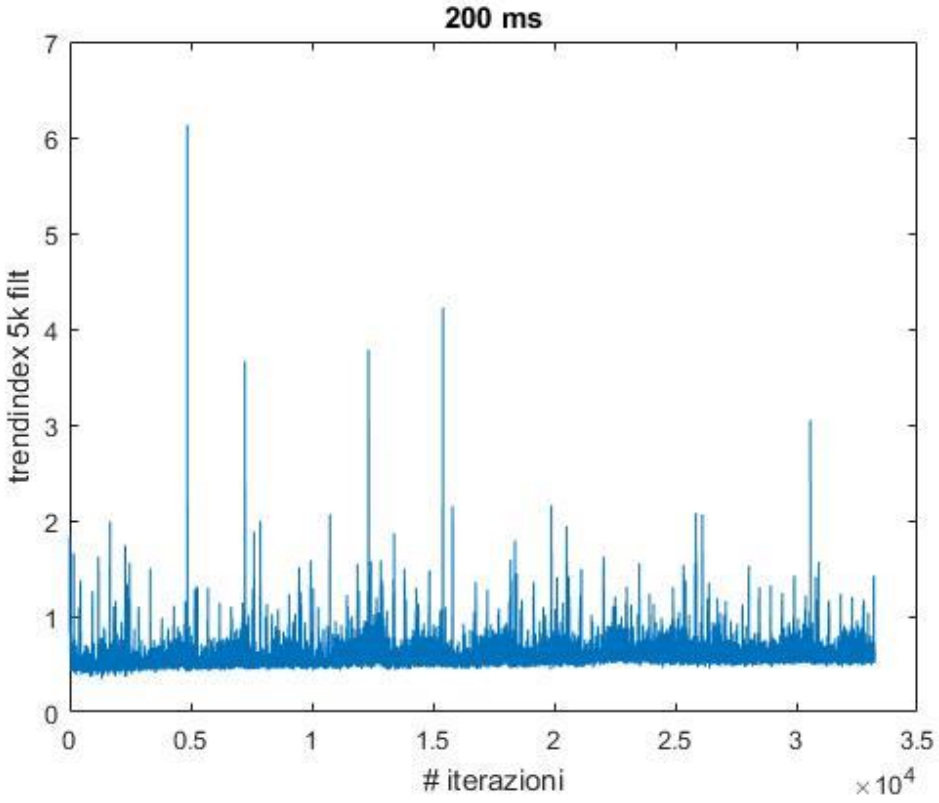


Figura 101: trendindex 200 ms con filtro di Hampel

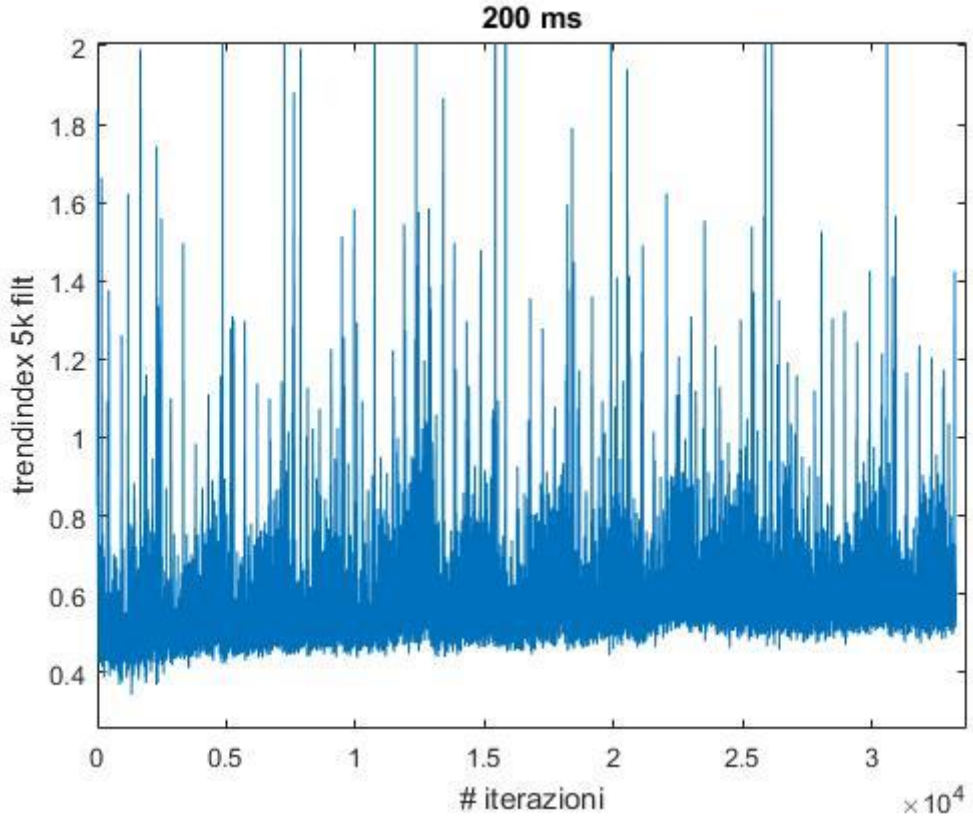


Figura 102:trendindex 200 ms con filtro di Hampel zoom

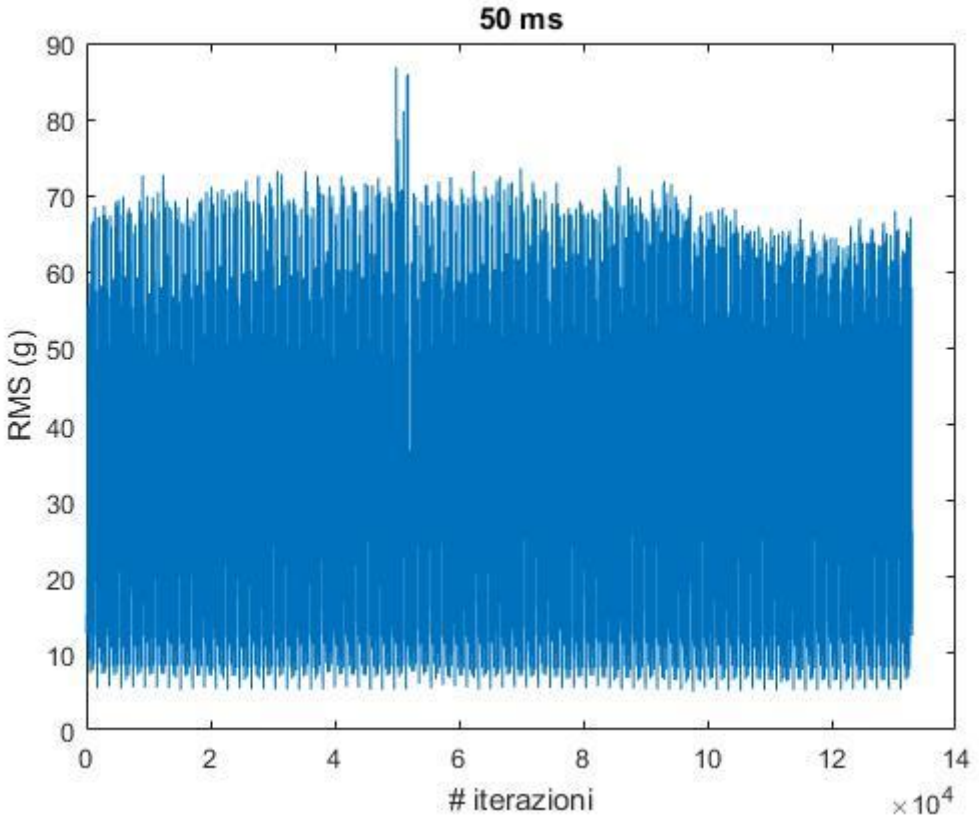


Figura 103: RMS 50 ms

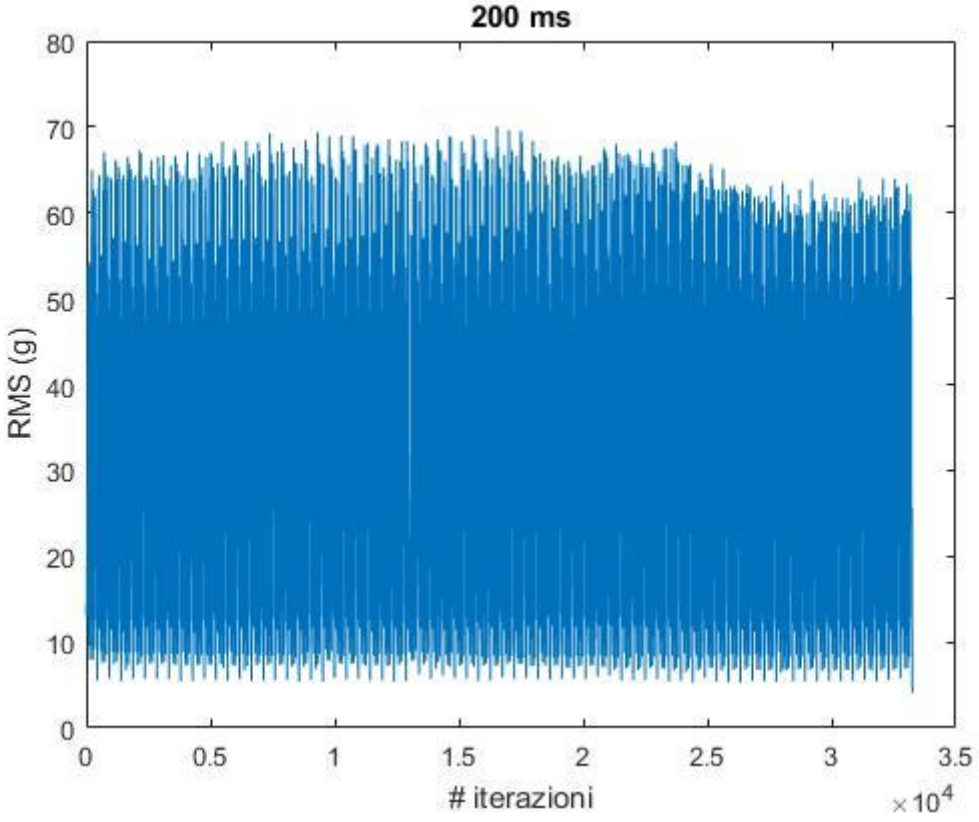


Figura 104: RMS 200 ms

Osservando i risultati filtrati che escludono i maggiori outlier risulta abbastanza evidente, sia per l'analisi a 50 ms che per quella a 200 ms, che l'indice di trend subisca un leggero incremento del suo livello di base, riscontrabile sull'inviluppo inferiore dell'indice stesso. Sebbene l'indice sia sicuramente troppo acerbo nella sua formulazione per poter essere utilizzato in modo attivo per la diagnosi, risulta sicuramente incoraggiante per i risultati che fornisce se confrontato con l'RMS, che anzi diminuisce verso la parte finale della prova.

I picchi estremamente elevati presenti nell'andamento non filtrato sono frutto di un aumento repentino di diversi ordini dell'ampiezza del segnale registrato dall'accelerometro (su tutte le frequenze) riscontrato in particolare in una registrazione (l'intera durata non è stata svolta come unico run ma il motore è stato fermato alcune volte).

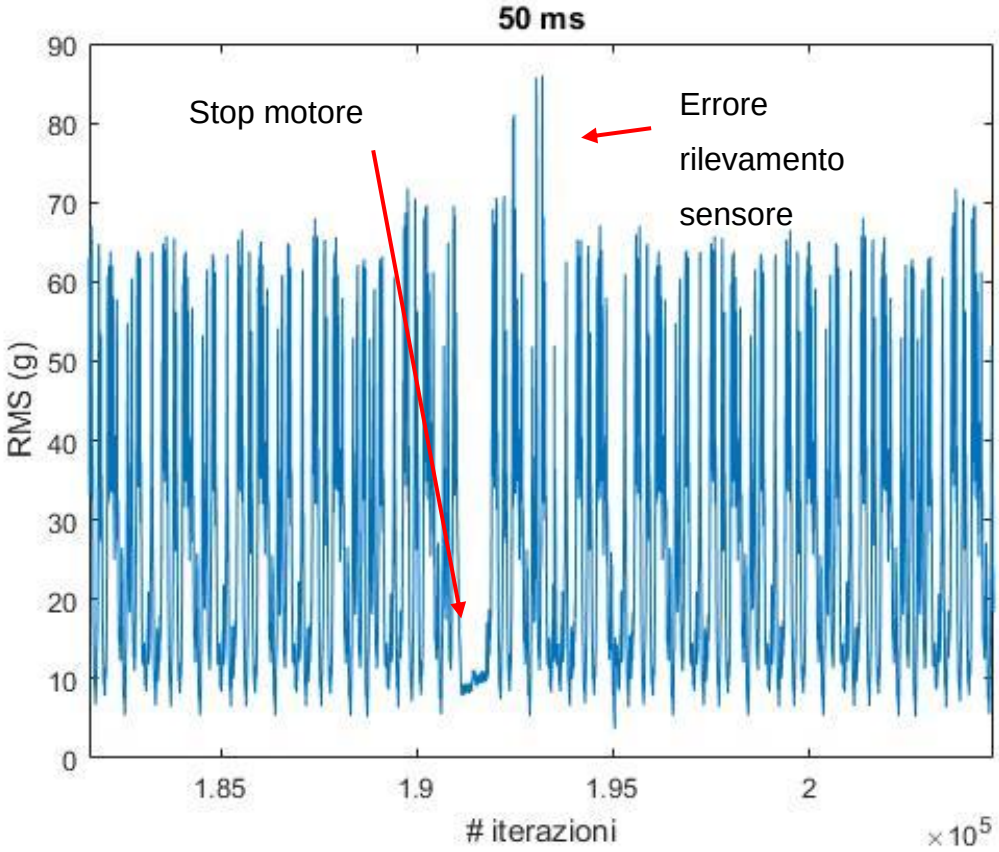


Figura 105: picchi in corrispondenza del riavvio del motore

Durante la giunzione dei file di registrazione è stato accuratamente evitato di includere le parti di arresto del motore (mostrata invece nella figura 98), mentre gli outlier, causati da errori o problemi sull'acquisizione da parte del sensore, possono essere semplicemente ignorati o filtrati.

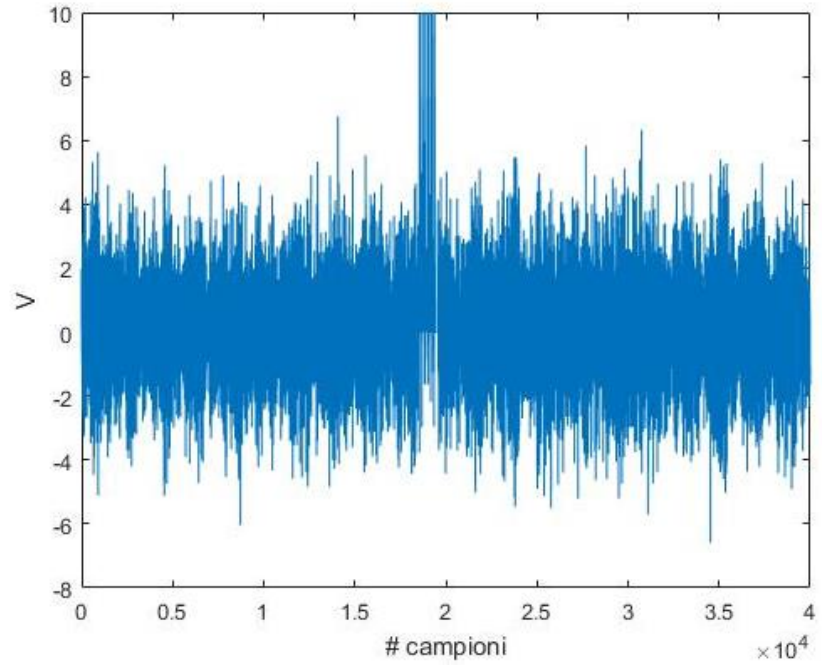


Figura 106: anomalia sull'acquisizione

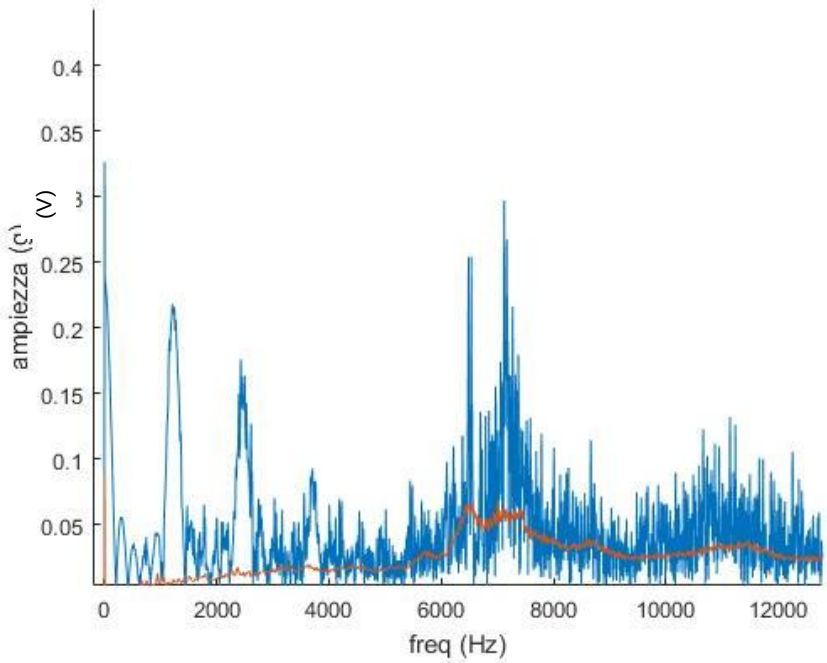


Figura 107: anomalia: confronto fra spettro attuale e di riferimento che causa l'outlier sul trendindex calcolato

4.6 **CONCLUSIONI**

Le analisi condotte hanno mostrato l'importanza di numerosi parametri correlati fra loro, le cui conseguenze sui risultati finali non sono univocamente determinate, ma dipendono dalle condizioni di impiego del motore. Tuttavia, fissate alcune variabili e manipolando opportunamente i dati in possesso è stato **possibile identificare negli indici calcolati un trend riconducibile al degrado del motore** nel corso delle prove di durata eseguite, con un contenuto informativo sullo stato di salute molto superiore a quello fornito dall'RMS.

Questo approccio alla diagnosi dei guasti risulta promettente, ma necessita di ulteriori campagne sperimentali e dell'approfondimento delle seguenti tematiche in relazione alle condizioni operative che il motore deve affrontare:

- Corretta scelta del tempo di analisi
- Corretta scelta della griglia della mappa
- Diagnosi mirate alle singole tipologie di guasti basate su differenti range di frequenze
- Valutazione delle tolleranze operative e delle soglie limite per il trendindex
- Calcolo del trendindex in presenza di guasti reali

Inoltre, come accennato anche nel capitolo 3.6 è possibile pensare di convertire l'intera analisi dal dominio delle frequenze a quello degli ordini, rendendo più facile l'individuazione delle armoniche legate ai singoli componenti e rendendo meno problematica la presenza di transitori. Un approccio di questo tipo consentirebbe anche di costruire indici molto più selettivi, calcolati pesando diversamente o selezionando le componenti spettrali d'interesse.

Capitolo 5
DIAGNOSI DETONAZIONE TRAMITE ACCELEROMETRI

Per il rilevamento di cicli detonanti non è sempre possibile disporre dei sensori di pressione in camera per via del loro alto costo e del condizionamento del segnale, tendenzialmente solamente durante le fasi di calibrazione o se si sta lavorando su di un motore da Formula1 si hanno a disposizione i sensori di pressione in camera. Diventa perciò importante avere a disposizione altri strumenti per diagnosticare l'insorgenza di knock e predisporre un controllo in feedback per salvaguardare il motore. Con particolare riferimento al motore in esame, situazioni di questo tipo possono presentarsi quando il motore è installato sulla moto in pista oppure durante le prove di durata al banco prova. L'intenzione delle prove sperimentali e delle analisi svolte è stata quella di verificare la realizzabilità di un controllo detonazione basato su accelerometri, in modo da consentirne l'applicabilità più diffusamente durante le attività. Sono state quindi svolte delle prove mirate a confrontare gli indici di detonazione ricavati dal segnale di pressione e da quello accelerometrico, prendendo come riferimento il primo e valutando la correlazione fra i due.

5.1 POSIZIONAMENTO SENSORI

I sensori utilizzati sono del tipo Bosch KS-P come quello impiegato sul basamento per il condition monitoring e da specifica hanno un range di frequenze di funzionamento dedicato al rilevamento detonazione (1 kHz – 20 kHz).

Ai fini dell'ottenimento della massima sensibilità alla detonazione, e quindi correlazione con gli indici di pressione, il posizionamento ed il fissaggio sono fondamentali. Tuttavia il carattere di prima investigazione delle prove e l'esigenza di un'installazione "plug&play" ha portato al montaggio degli accelerometri in modo non uguale sui cilindri e probabilmente in posizioni non ottimali. Si è comunque cercato di posizionare i sensori su parti rigide dei cilindri e che potessero trasmettere il più possibile le

frequenze tipiche dei fenomeni di detonazione. Sulla base di esperienze precedenti è stato perciò deciso che il piazzamento degli accelerometri dovesse avvenire in prossimità dei prigionieri della testata. La disponibilità di filettature libere sulle testate ha portato alla seguente installazione sui due cilindri:

Cilindro Verticale (cyl2)

L'installazione è stata effettuata tramite vite M8 su una filettatura posta su delle nervature collegate al prigioniero sinistro lato aspirazione (interno).

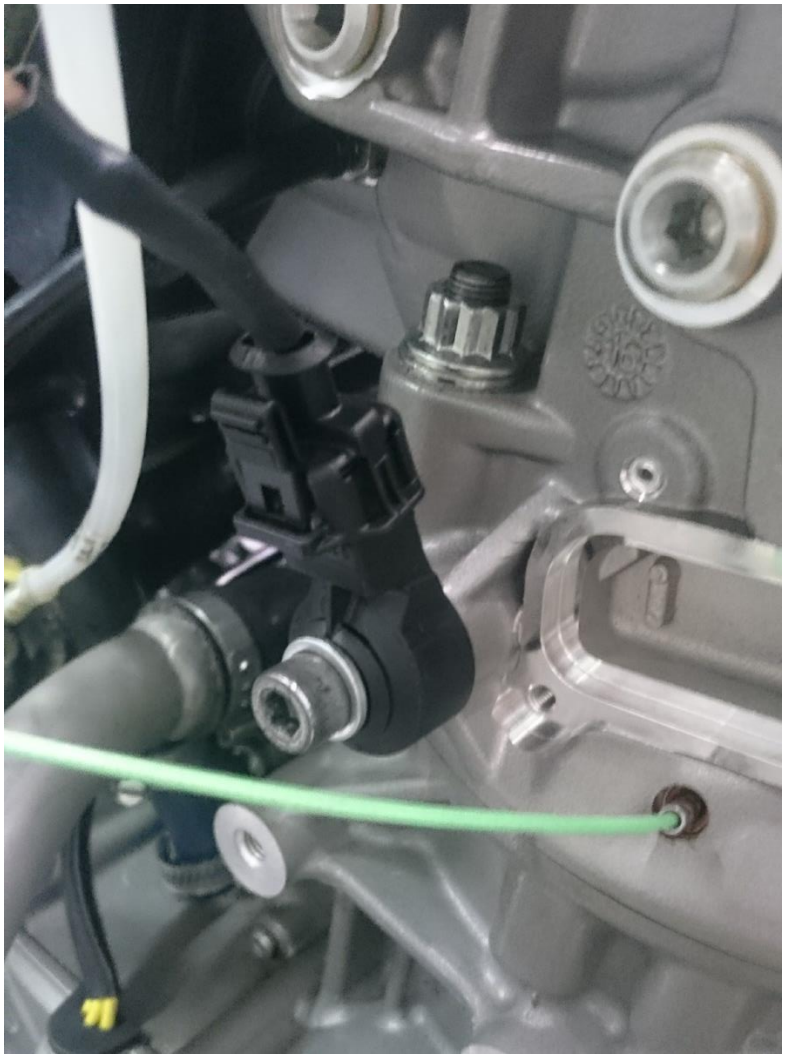


Figura 108: posizionamento accelerometro knock sul cilindro verticale

Cilindro Orizzontale (cyl1)

L'installazione su questo cilindro non è avvenuta su un'analoga nervatura sul prigioniero, bensì su una filettatura M6 posta sul cilindro poco al di sotto della congiunzione con la testata. Vicino al prigioniero destro lato scarico è presente una filettatura M6 che poteva essere utilizzata, ma sia per ragioni di installazione che per differenziare il piazzamento è stato deciso in fase di montaggio di non utilizzarla.

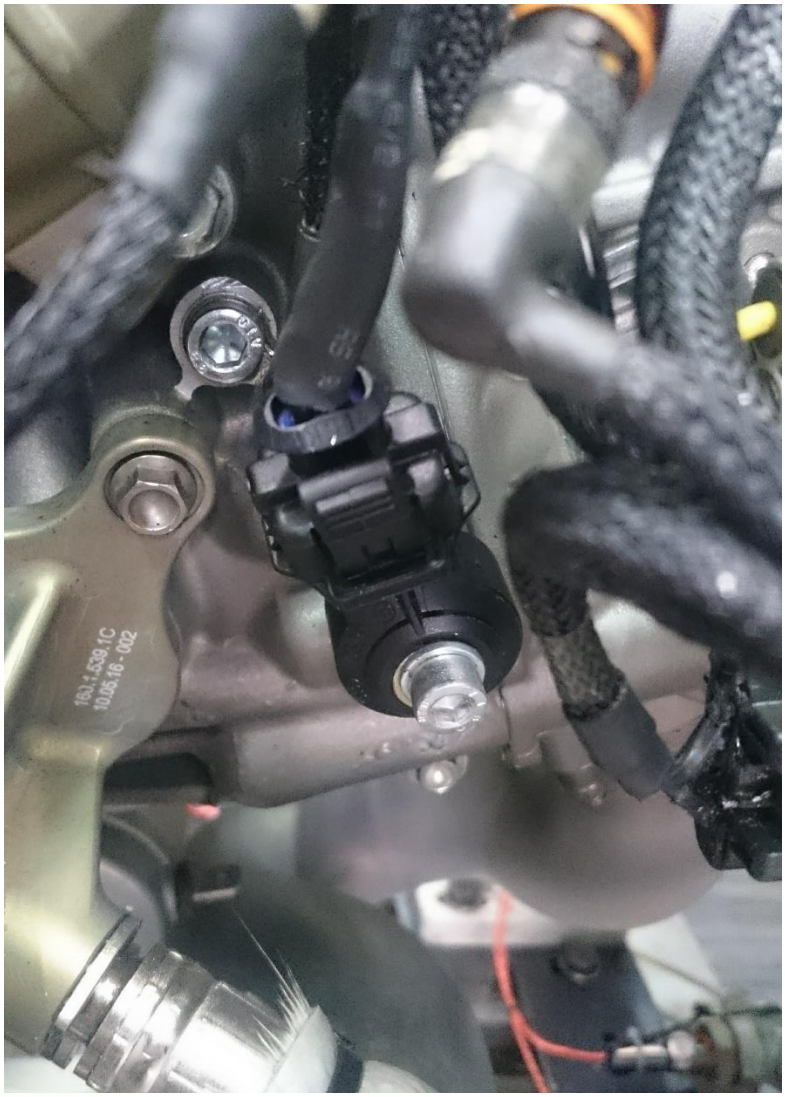


Figura 109: posizionamento accelerometro knock sul cilindro orizzontale

A causa della diversa filettatura utilizzata e del diverso posizionamento i dati provenienti dai 2 cilindri non sono direttamente confrontabili. Il serraggio è stato comunque posto per entrambi a 15 Nm, tenendosi al minimo consentito da Bosch per l'installazione (20±5 Nm).

5.2 CALCOLO DEGLI INDICI E DELLA CORRELAZIONE

I segnali provenienti dagli accelerometri e dai sensori di pressione sono stati registrati durante la simulazione di 5 giri di pista, durante i quali vengono sostanzialmente attraversate tutte le condizioni di funzionamento e possono essere ritenuti un ottimo riferimento per la costruzione di una correlazione robusta fra indici accelerometrici e di pressione. Infatti, il segnale raccolto dagli accelerometri è pesantemente influenzato da quello che è il rumore meccanico prodotto dal motore, e riuscire ad ottenere una buona correlazione in condizioni estremamente dinamiche, con la simulazione di cambiate e scalate, è sicuramente più difficile, ma anche più significativo per lo sviluppo di un sistema di controllo detonazione che possa essere robusto in termini di mancate diagnosi e falsi allarmi.

La presenza cicli detonanti lascia un'impronta caratteristica sulle frequenze presenti nel segnale, sia esso di pressione o accelerometrico, ed a grandi linee il range di frequenze caratteristico della detonazione va dai 5 kHz fino ai 50 kHz e dipende strettamente dai modi di vibrare della camera di combustione e della struttura del motore eccitati dal fenomeno. È necessario quindi filtrare il segnale passa alto per effettuare una diagnosi migliore del fenomeno, isolandone i sintomi da quelle che sono le naturali oscillazioni prodotte dal funzionamento del motore (fino alla 25ª-30ª armonica di ciclo → 3-4 kHz).

Confrontando un certo numero di cicli detonanti con cicli "normali" nelle medesime condizioni di funzionamento è stato scelto un filtro di **Butterworth del 4° ordine con frequenza di taglio posta a 4 kHz**. Come esempio sotto è riportato il confronto fra un ciclo detonante ed un suo omologo non detonante per quanto riguarda sia il segnale di pressione che accelerometrico, nel tempo ed in frequenza. È osservabile come per il segnale accelerometrico il rapporto rumore/segnale sia molto più alto e che il sensore vada in contro a saturazione: circa 400g corrispondenti ai 10V massimi acquisibili con il convertitore A/D di miracle.

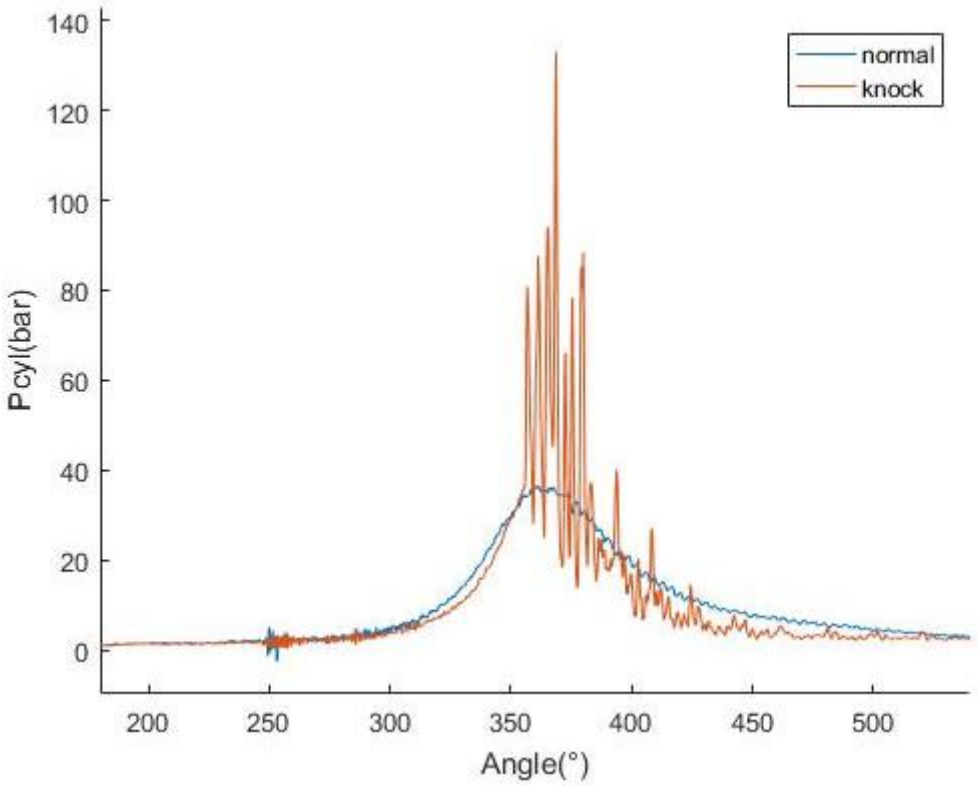


Figura 110: confronto sulle curve di pressione cilindro

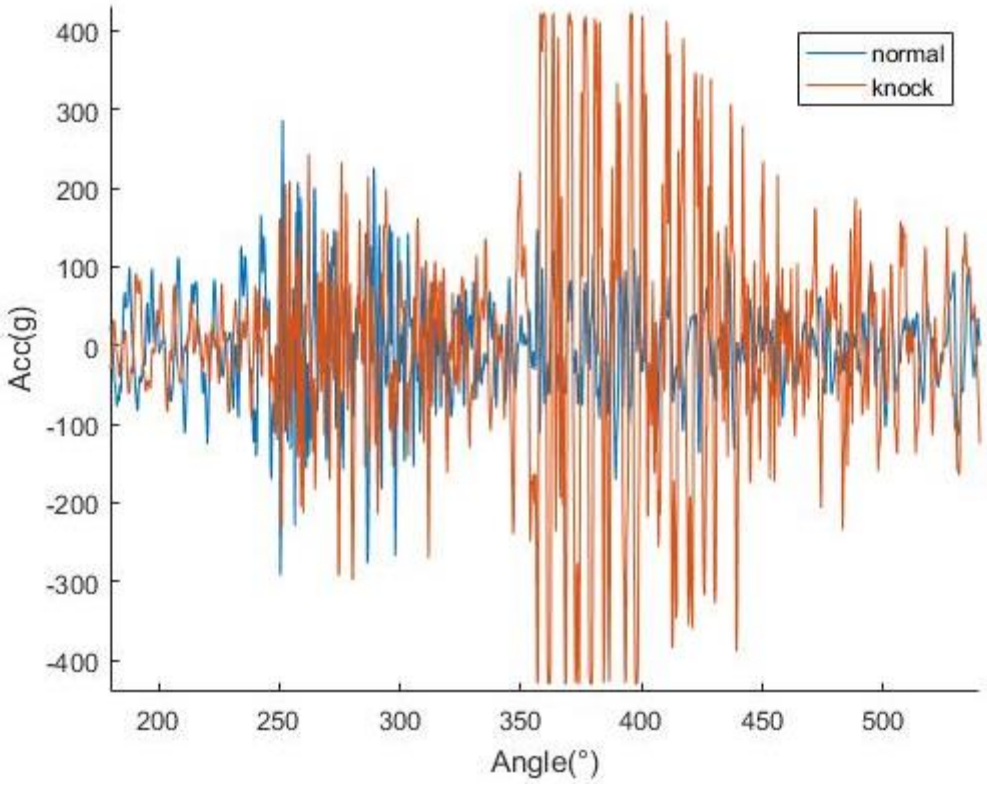


Figura 111: confronto del segnale accelerometrico

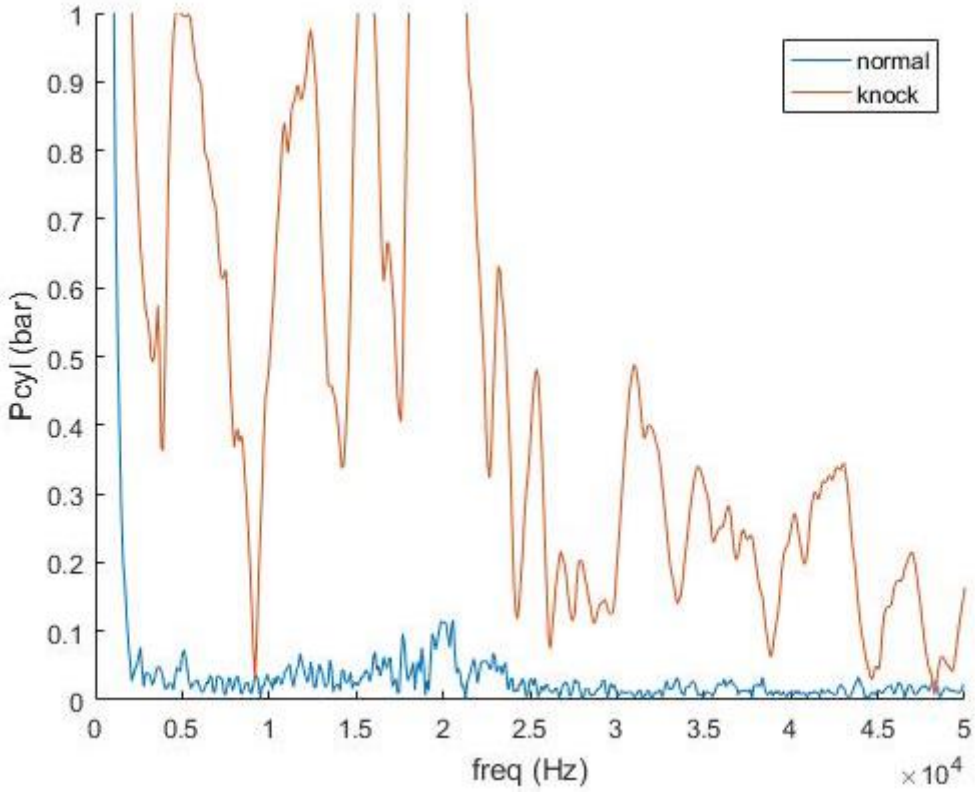


Figura 112: confronto dello spettro della pressione cilindro

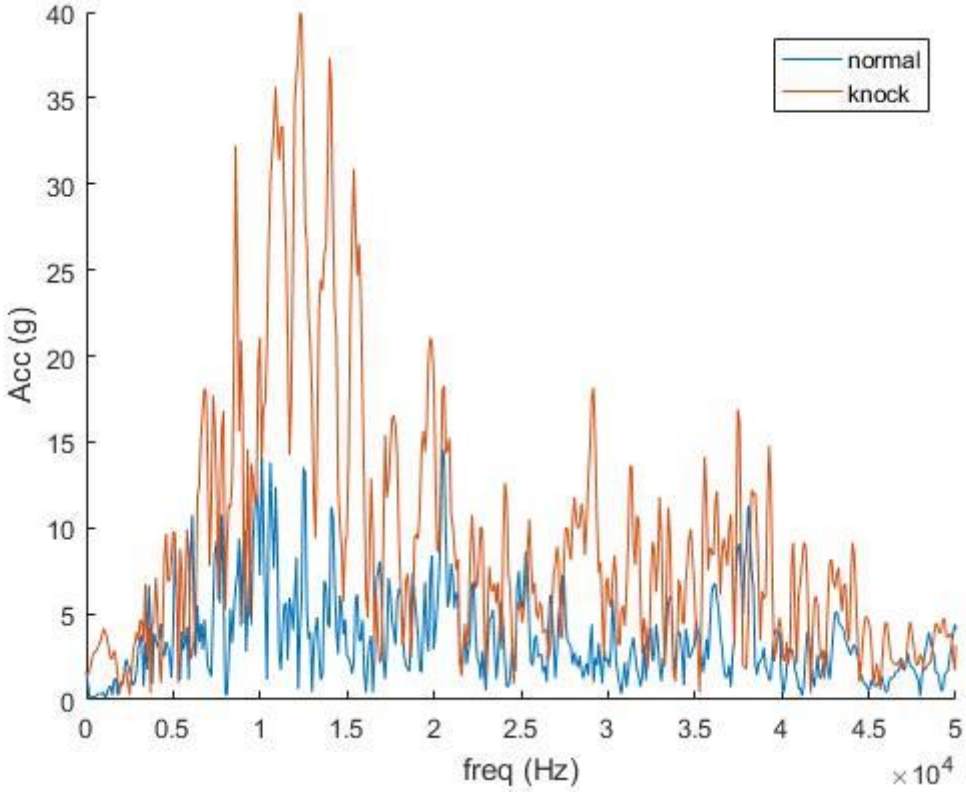


Figura 113: confronto dello spettro del segnale accelerometrico

A partire dai segnali grezzi sono stati presi in considerazione i seguenti indici sintetici ciclo-ciclo per valutare la presenza di detonazione:

MAPO (Max Amplitude Pressure oscillation)

Valuta l'ampiezza massima delle oscillazioni rilevate sul segnale filtrato passa alto. Si presta bene alla valutazione del segnale di pressione cilindro. Può ovviamente essere calcolato anche sul segnale accelerometrico.

$$\boldsymbol{MAPO = max(|Segnale_{filt}|)_{start\_window}^{end\_window}}$$

KINT (Knock Integral)

Valuta il valore medio del segnale filtrato passa alto. Si presta bene alla valutazione del segnale accelerometrico vista la sua natura più rumorosa.

$$\boldsymbol{KINT = \frac{1}{N} \sum_{start\_window}^{end\_window} |Segnale_{filt}|}$$

Per valutare la correlazione fra gli indici accelerometrici e quelli derivanti dalla pressione cilindro è stato usato l'indice di correlazione (Bravais-Pearson) così definito:

$$R_{xy} = \frac{\sigma_{xy}}{\sigma_x \sigma_y}$$

$$\sigma_{xy} = \frac{1}{N} \sum_{i=1}^{N} (x_i - \mu_x)(y_i - \mu_y)$$

$$\sigma_x = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \mu_x)^2}{N}}$$

I valori sono stati calcolati attraverso il software HeatIT-off di Alma Automotive a partire dalle registrazioni effettuate con OBI. Per il calcolo degli indici è stata scelta

una **finestra angolare fissa da -20° a +70°** rispetto al punto morto superiore della fase attiva.

Per distinguere gli indici calcolati sul segnale di pressione o accelerometrico si userà di seguito la seguente notazione:

- **MAPO**: oscillazione massima di **pressione**
- **KPINT**: indice integrale di **pressione**
- **MAAO**: oscillazione massima dell'**accelerazione**
- **KAINT**: indice integrale dell'**accelerazione**

Di seguito sono riportate le correlazioni ottenute per i due cilindri. Al fine della valutazione fra indici accelerometrici e di pressioni il grafico più significativo è sicuramente quello che riporta **MAPO** e **KAINT**

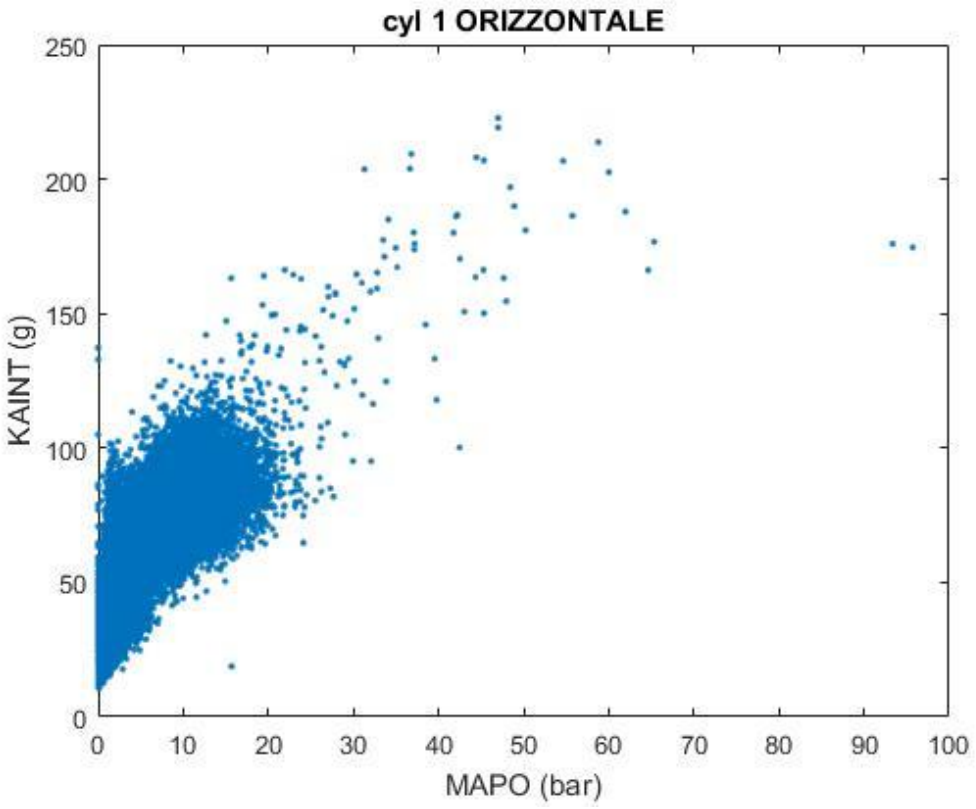


Figura 114: MAPO-KAINT cilindro 1 R= 0.8122

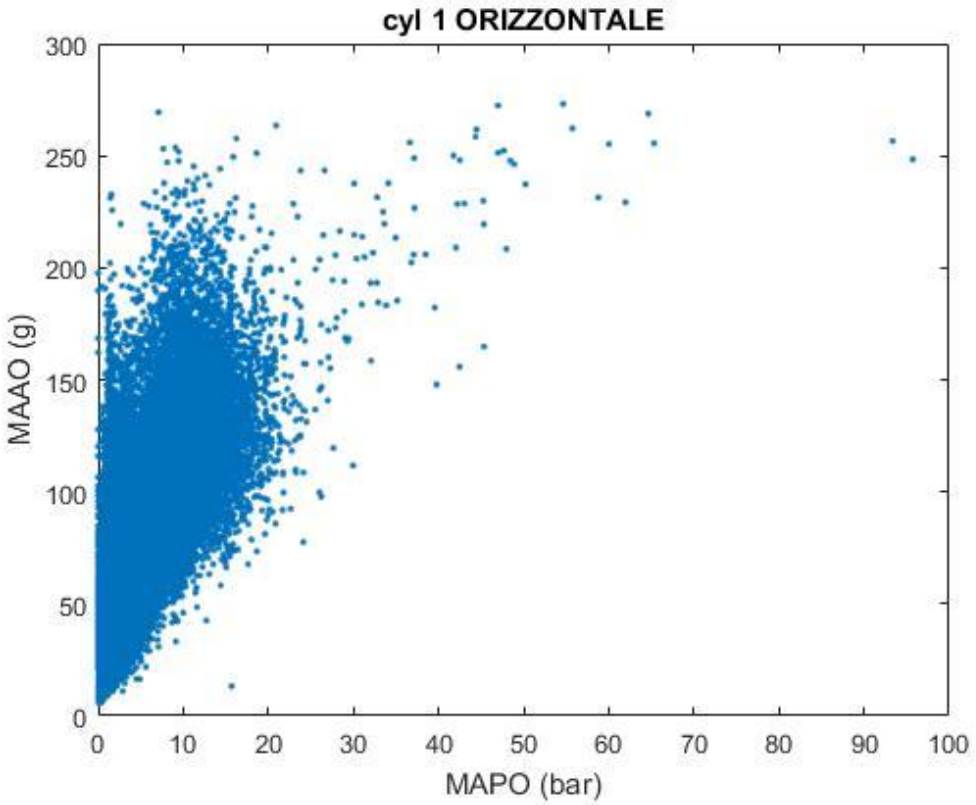


Figura 115: MAAO-MAPO cilindro 1 0.7430

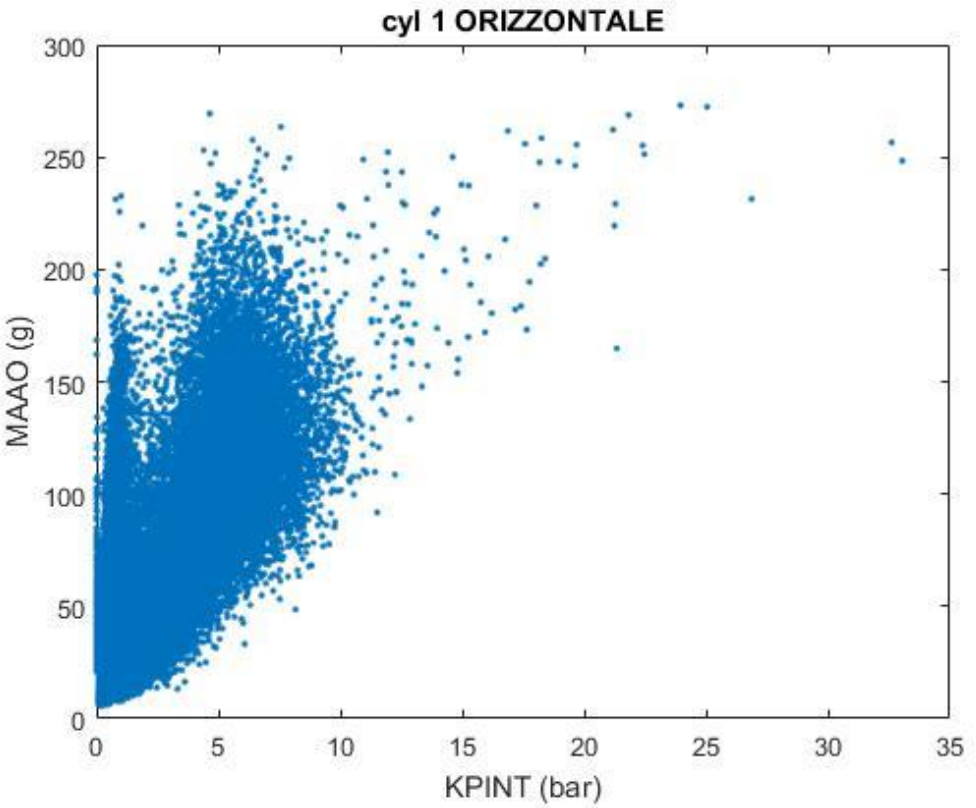


Figura 116: KPINT-MAAO cilindro 1 0.7118

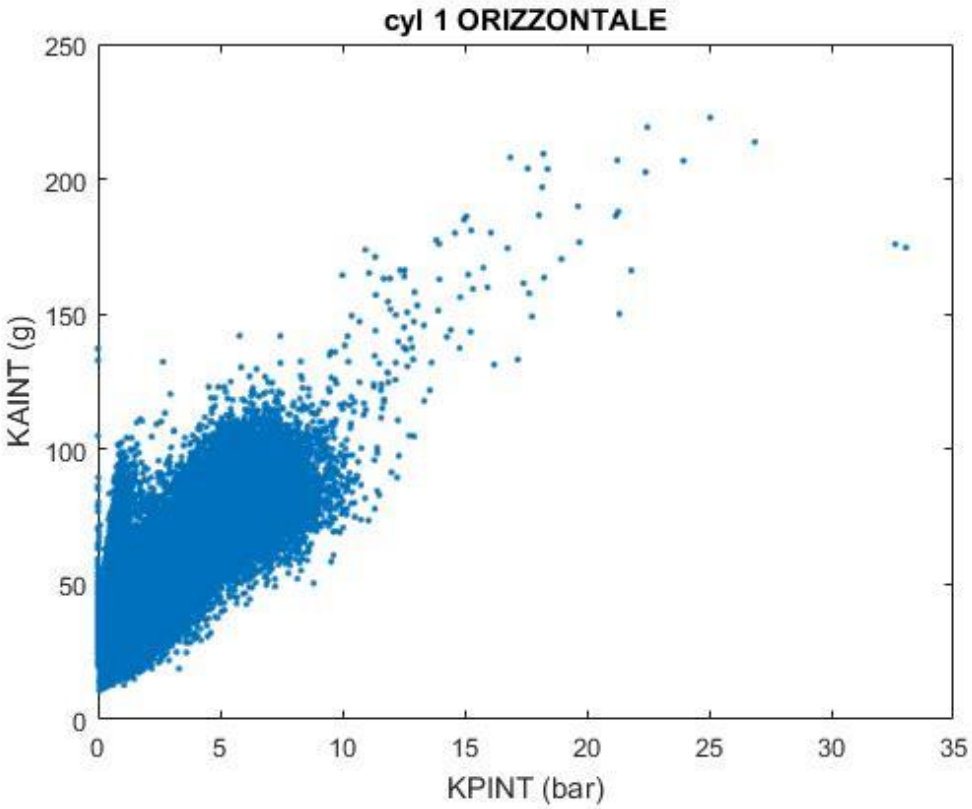


Figura 117: KPINT-KAINT cilindro 1 0.7960

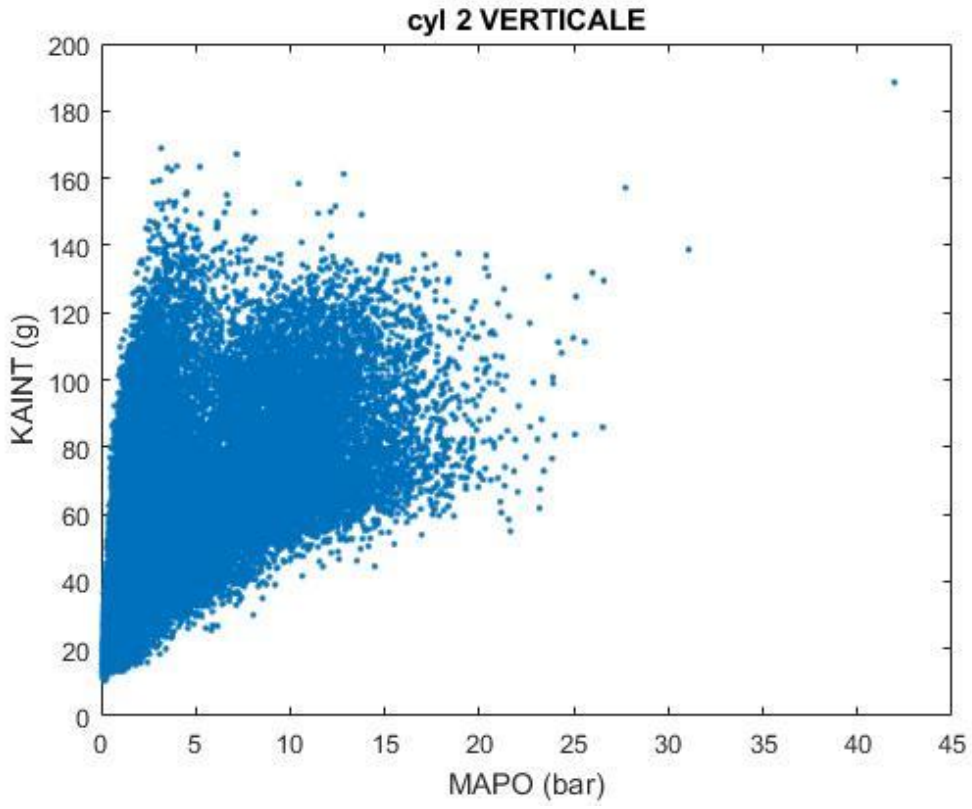


Figura 118: MAPO-KAINT cilindro 2 0.6305

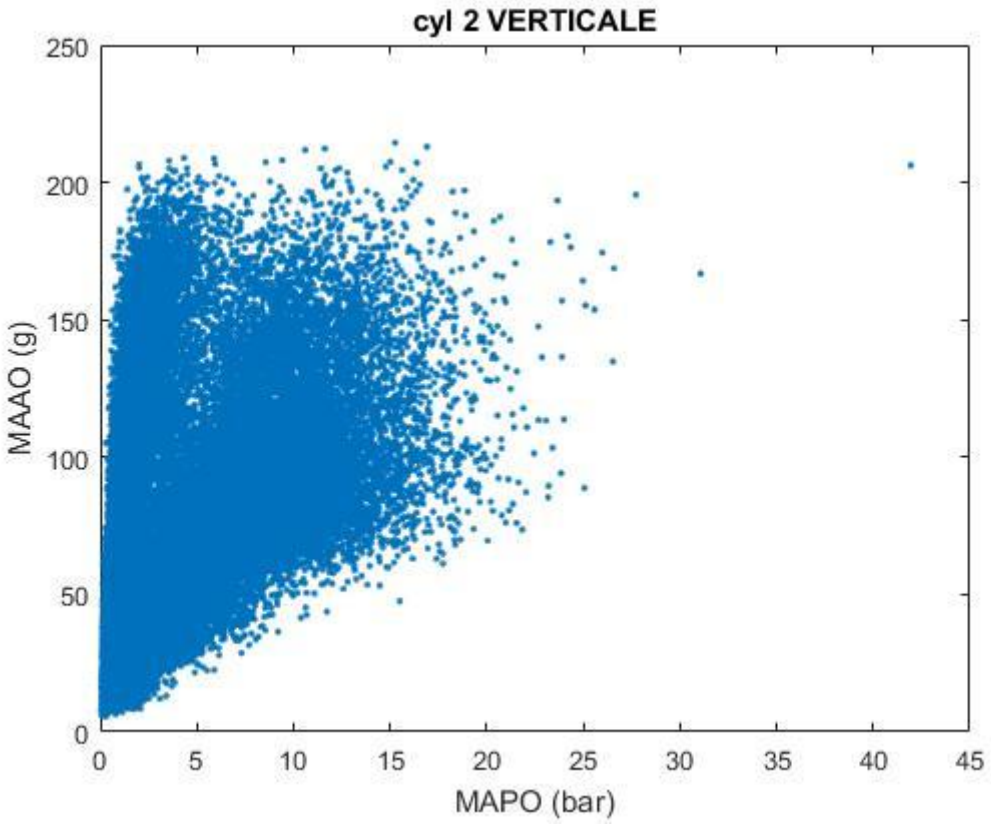


Figura 119: MAPO-MAAO cilindro 2 R= 0.5443

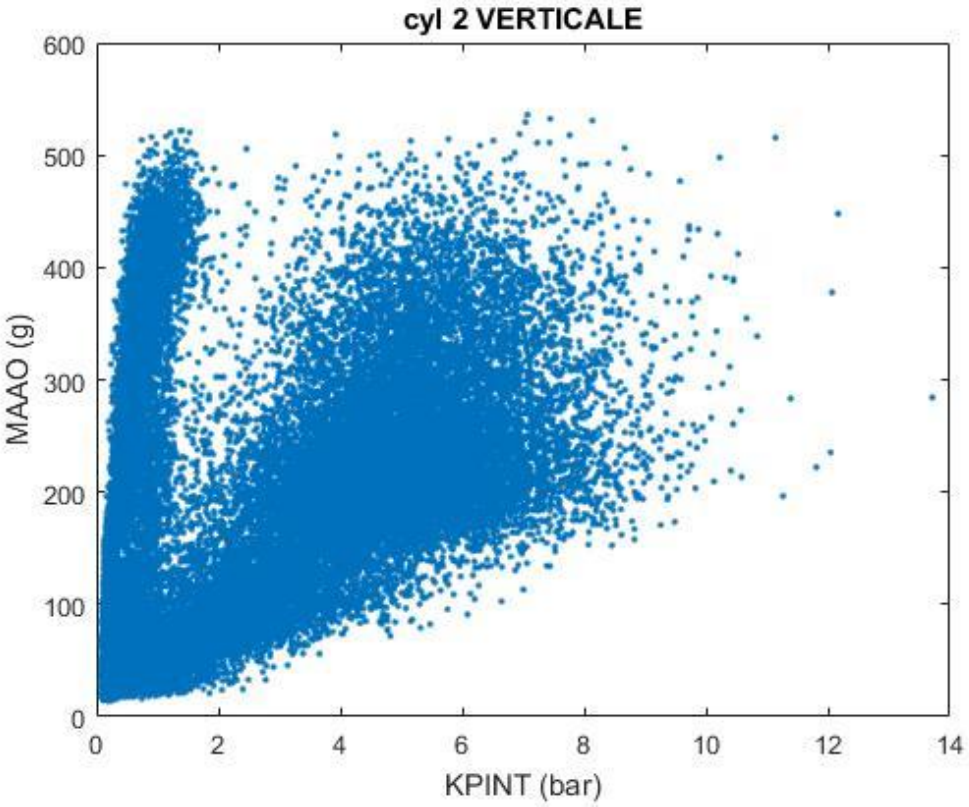


Figura 120: KPINT-MAAO cilindro 2 R=0.4466

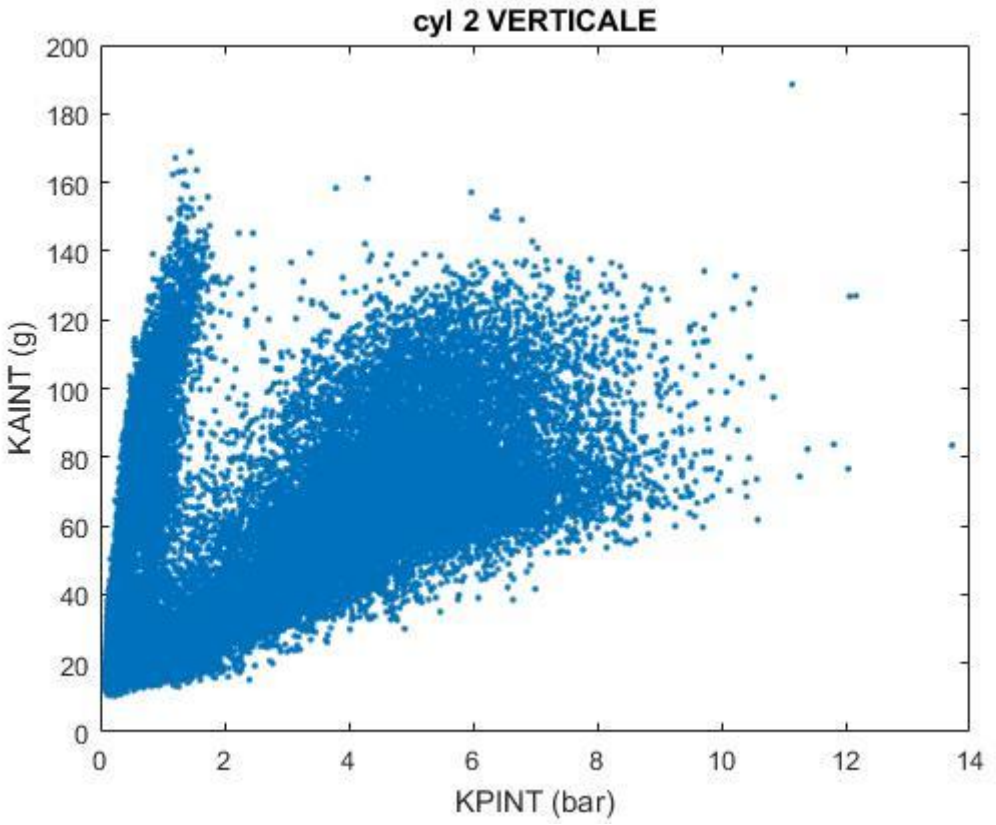


Figura 121: KPINT-KAINT cilindro 2 R=0.5449

È possibile fare le seguenti osservazioni:

- Le correlazioni migliori per entrambi i cilindri si ottengono confrontando il MAPO con il KAINT così come preventivato.
- Il cilindro 1 gode di una correlazione molto migliore del cilindro 2, probabilmente a causa del montaggio e della presenza di una più spiccata tendenza a detonare.
- È presente una specie di “ramo” laterale che mostra indici di pressione bassi e indici accelerometrici alti (falsi allarmi). Analizzando i dati a disposizione si è notato che questa distribuzione anomala è prodotta nelle fasi di scalata a causa probabilmente delle forti vibrazioni indotte durante le cambiate ed a causa delle risonanze sulla particolare linea di carico installata sul motore durante il test.

Soffermandosi in particolare sul grafico KPINT-MAAO del cilindro 2 dove quest’ultimo effetto è ben visibile:

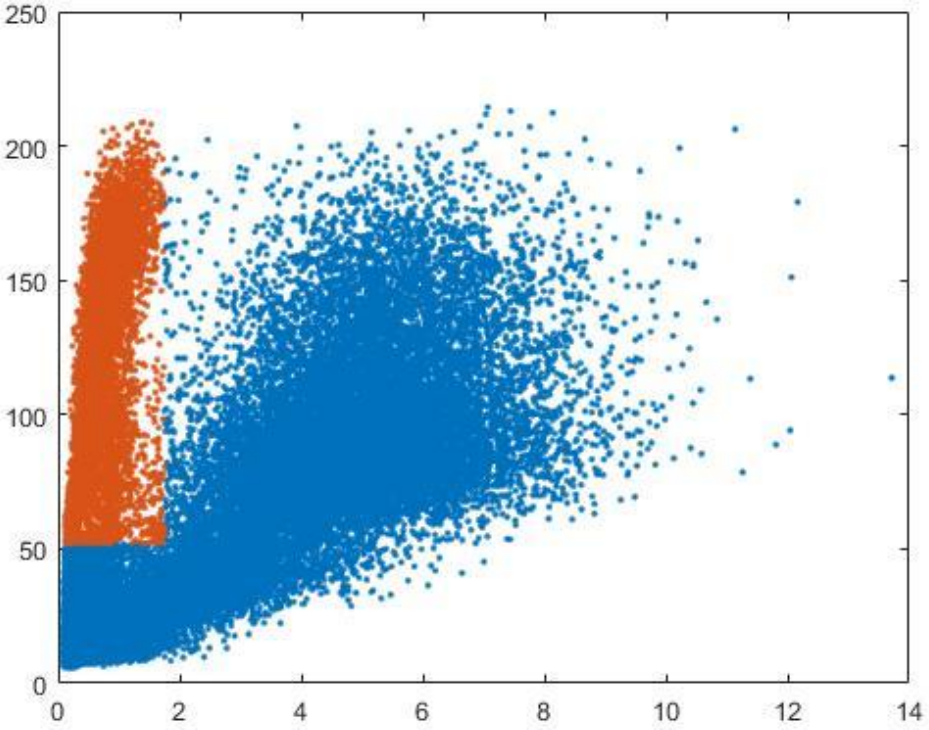


Figura 122: selezione dei cicli ritenuti "anomali" nella correlazione

Osservando sulla base degli RPM e della PMI quali cicli sono coinvolti:

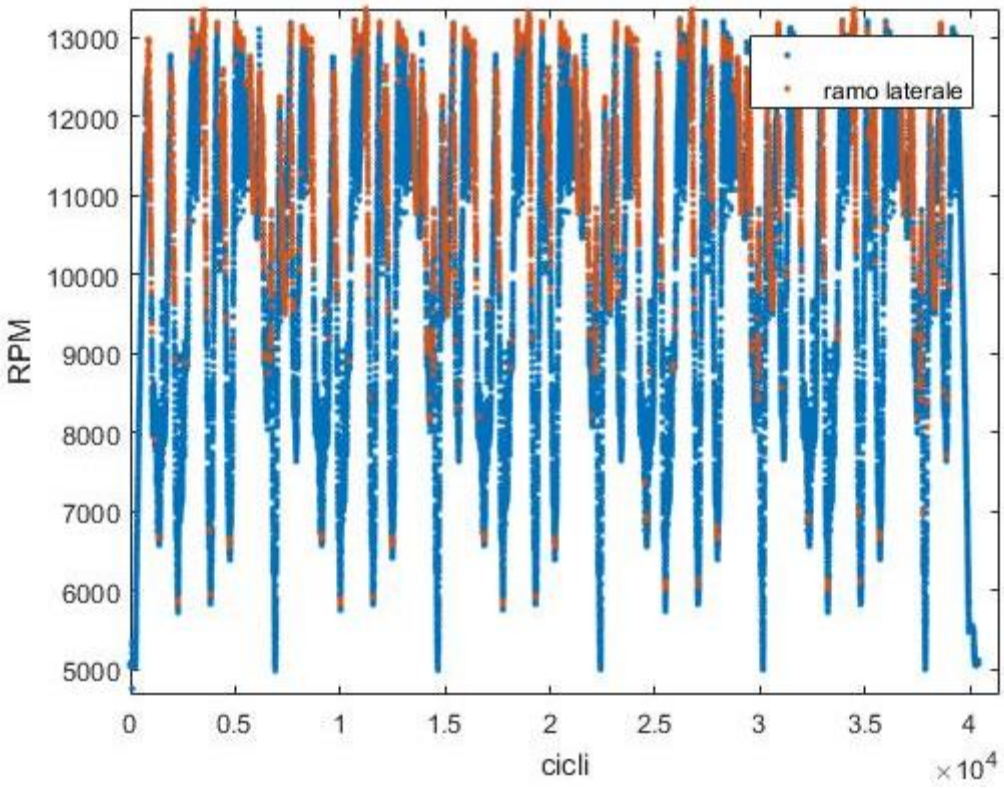
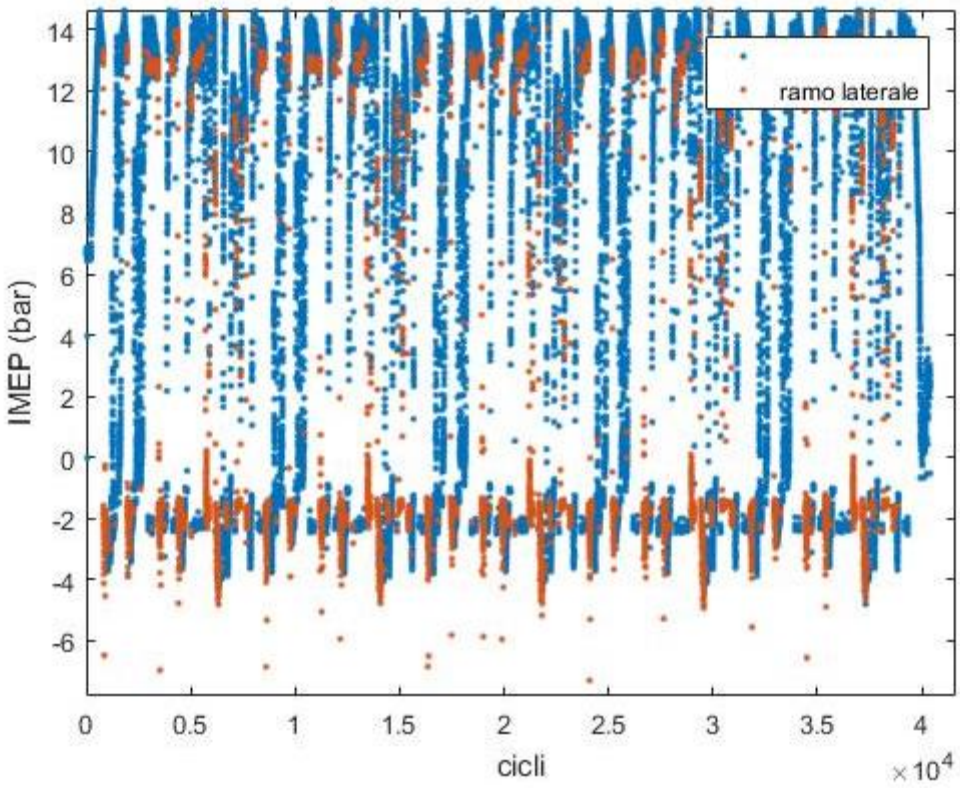


Figura 123: cicli del ramo laterale visualizzati sui giri



5.3 CONTROLLO KNOCK

Volendo implementare un controllo knock basato sull'indice integrale accelerometrico sarebbe dannoso in termini di falsi allarmi e mancate diagnosi, oltre che inutile, mantenere il controllo sempre attivo nell'arco del giro.

È possibile ad esempio immaginare di selezionare i cicli sui quali attivare il controllo knock ponendo le seguenti semplici condizioni d'intervento:

- RPM ciclo precedente – RPM ciclo attuale < 100
- IMEP > 8 bar

Con queste soglie è possibile selezionare solamente i cicli su sui è desiderabile avere attivo il controllo detonazione, cioè principalmente le fasi a pieno carico. Nella figura sottostante è mostrata la selezione effettuata nell'arco di un giro pista.

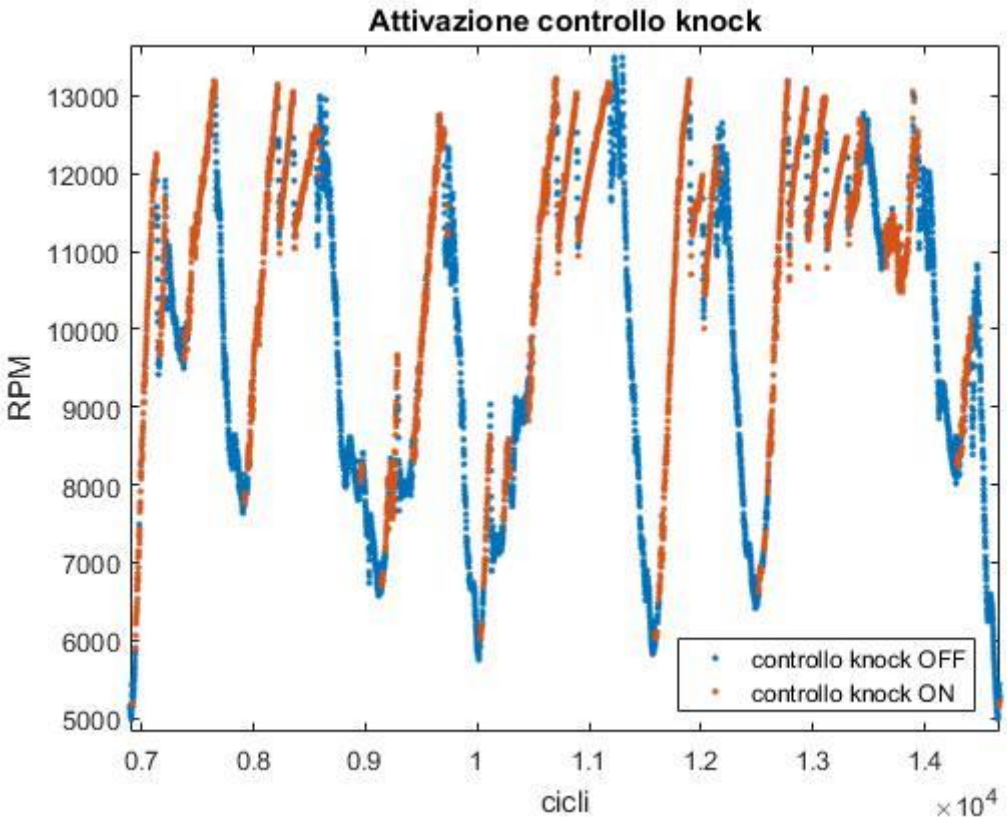


Figura 124: cicli sui quali è desiderabile avere attivo il controllo detonazione

Ovviamente questo è solamente un'ipotesi molto semplificata della selezione dei cicli e nel caso di implementazione real-time del controllo vanno previste logiche d'intervento più complesse e robuste, per evitare decurtazioni d'anticipo innescate da fattori da fattori esterni e di guida: cordoli, buche, cambiate, scalate, traction control possono causare diagnosi di detonazione errate a causa delle vibrazioni generate.

Il risultato che si ottiene sui cicli selezionati in termini di correlazione è leggermente inferiore sul cilindro 1, a causa dell'esclusione di cicli moto detonanti e ad alta correlazione causati dalla particolare configurazione di scarico installata sul motore durante la prova. Tuttavia la distribuzione è più pulita e per il cilindro 2 la correlazione aumenta.

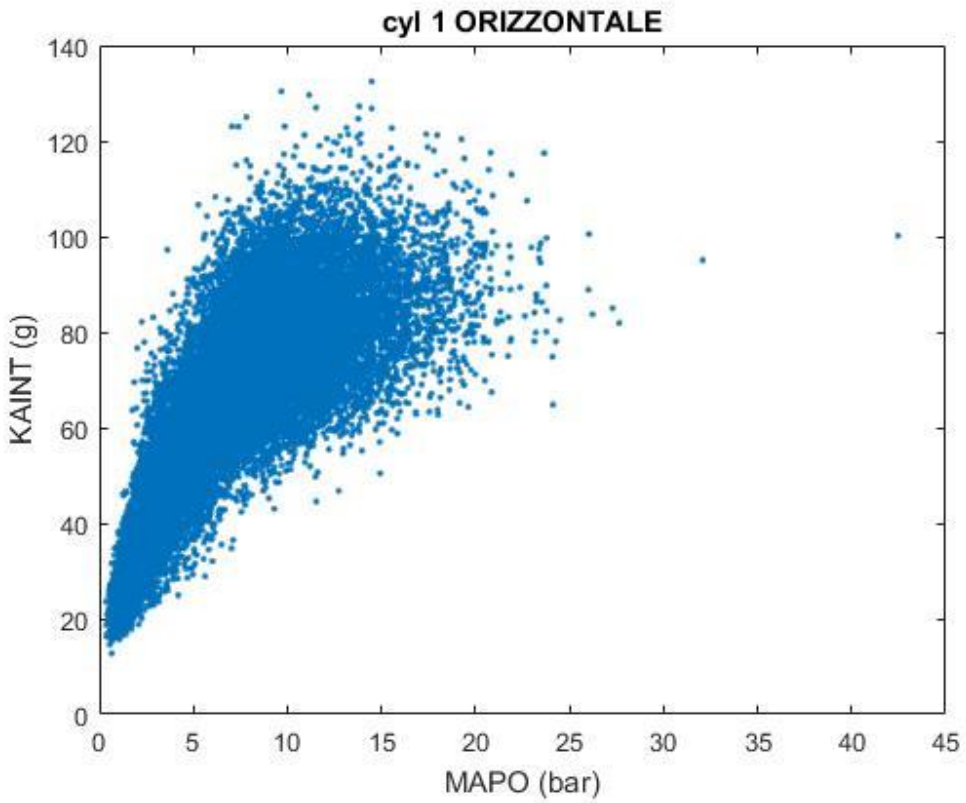


Figura 125: correlazione MAPO-KAINT sulla selezione per il cilindro 1 R=0.7665

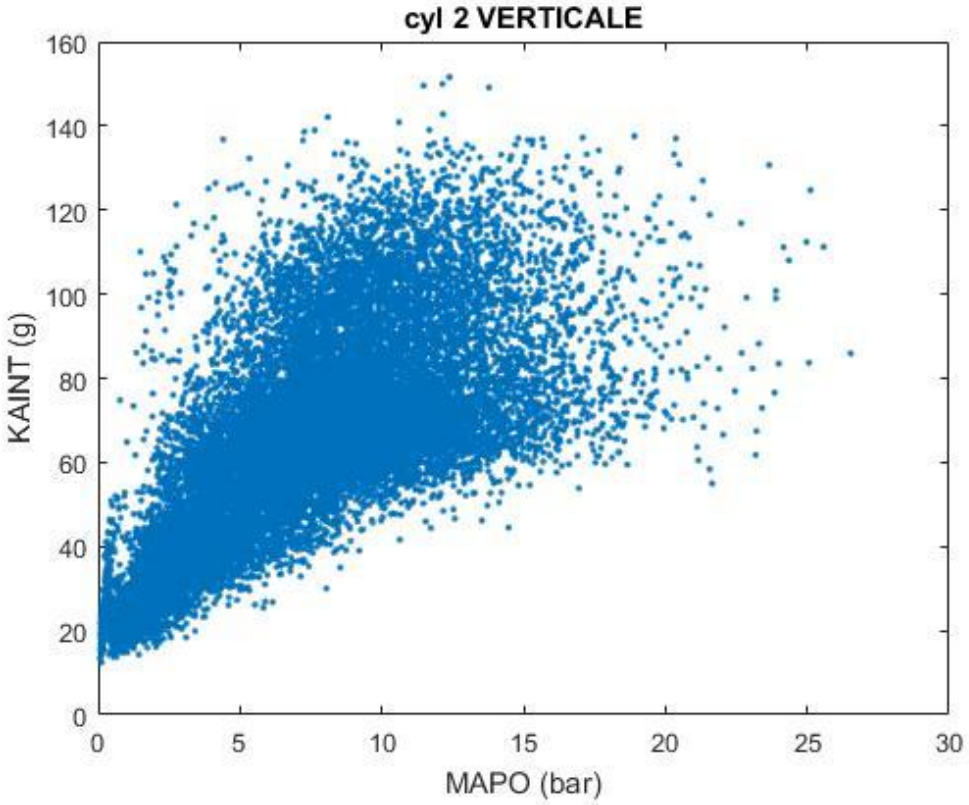


Figura 126: correlazione MAPO-KAINT sulla selezione per il cilindro 2 R=0.7125

Svolgendo inoltre un'analisi sull'andamento del KAINT accelerometrico in funzione di giri e carico è possibile iniziare a valutare le soglie d'allarme degli indici di detonazione mappate sulle condizioni di funzionamento, distinguendo quindi il trend dovuto al rumore meccanico da quello che è il comportamento detonante.

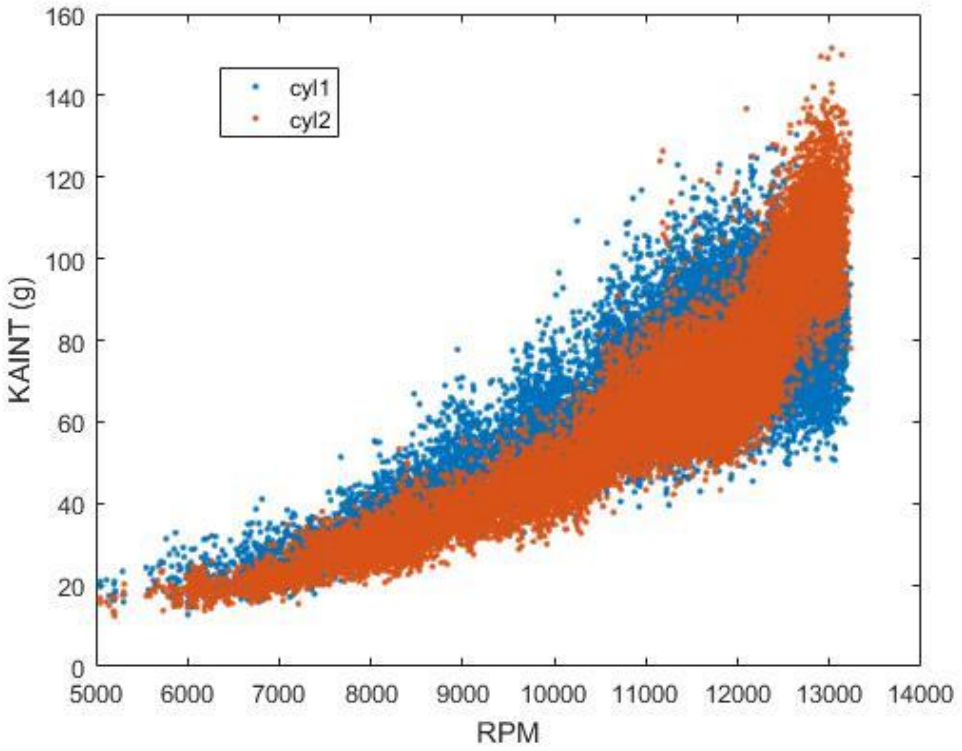


Figura 127: RPM – KAINT

La dipendenza dai giri è molto robusta e chiara, mentre è meno evidente la correlazione fra l'indice ed il carico.

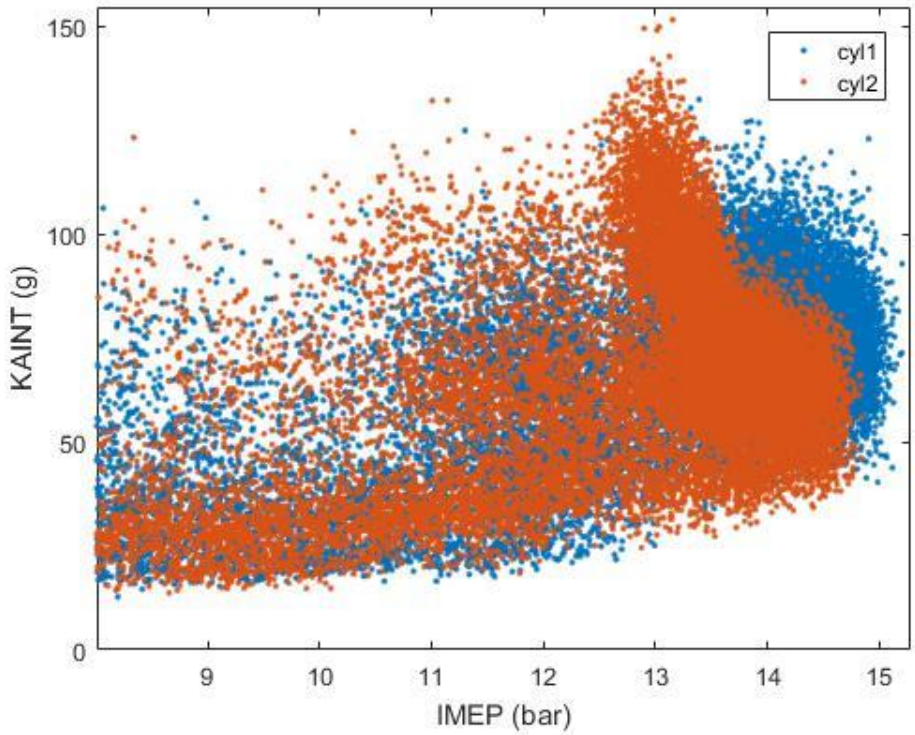


Figura 128: IMEP - KAINT

5.4 INDICI CON FINESTRA DINAMICA

La finestra angolare utilizzata finora è stata una finestra fissa compresa fra -20° e +70° attorno al PMS della fase attiva del ciclo. Questa finestra fissa si è dimostrata quella ottimale dopo alcune prove, che per brevità della trattazione non verranno riportate. Tuttavia è possibile implementare un algoritmo in grado eseguire il calcolo degli indici accelerometrici su una finestra dinamica, che a partire da quella fissa diventi più stretta e più centrata sul fenomeno stesso. La procedura di calcolo prevede i seguenti passaggi:

- Calcolo sulla finestra fissa -20/+70 degli indici indicativi della detonazione KAINT
- Individuazione dei campioni corrispondenti al 10° ed al 90° percentile di KAINT
- Ricalcolo dell'indice KAINT sulla finestra compresa fra il campione corrispondente al 10° percentile e quello corrispondente al 90°

La procedura di calcolo è stata implementata in Matlab ed eseguita su un set di dati ridotto proveniente sempre dalla stessa registrazione di 5 giri con cui sono state svolte le considerazioni precedenti. Per assicurarsi che la procedura non contenesse errori è stato verificato che anche tutti gli altri indici calcolati precedentemente con HeatIT-off combaciassero come valori.

Sul set di dati utilizzato la finestra dinamica produce un miglioramento della correlazione:

Cilindro 1:

R=0.7978 → R=0.8153

Cilindro 2:

R=0.6408 → R=0.7745

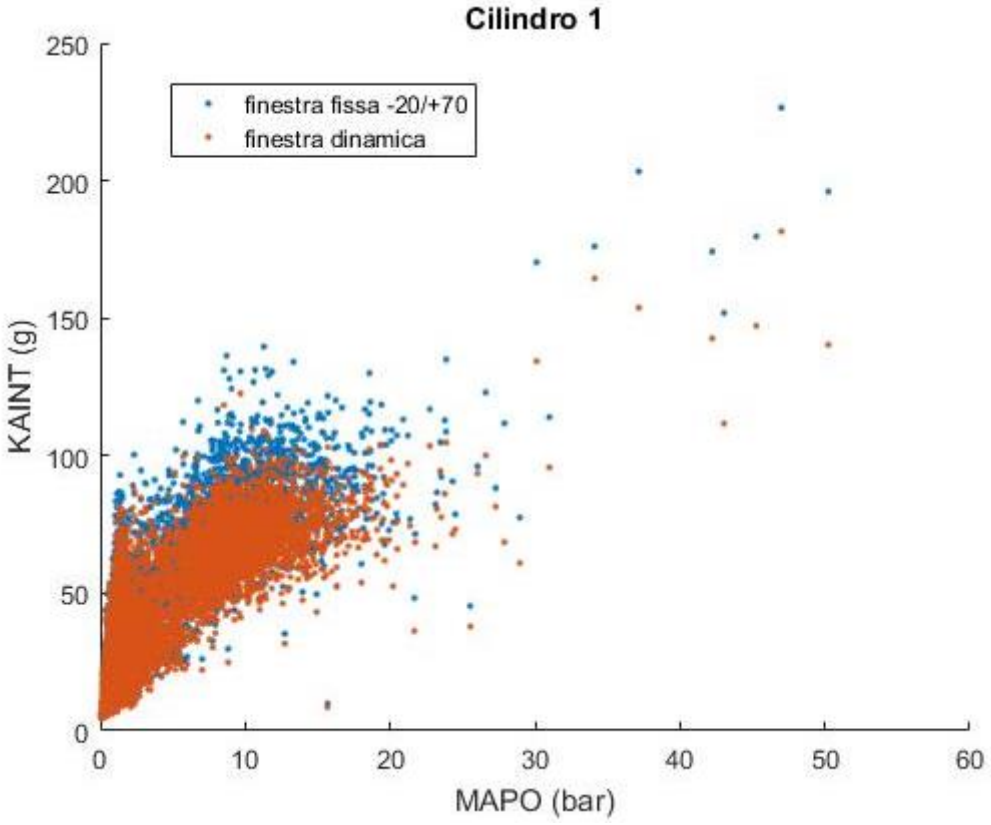


Figura 129: confronto sul cilindro 1 fra finestra dinamica e finestra fissa

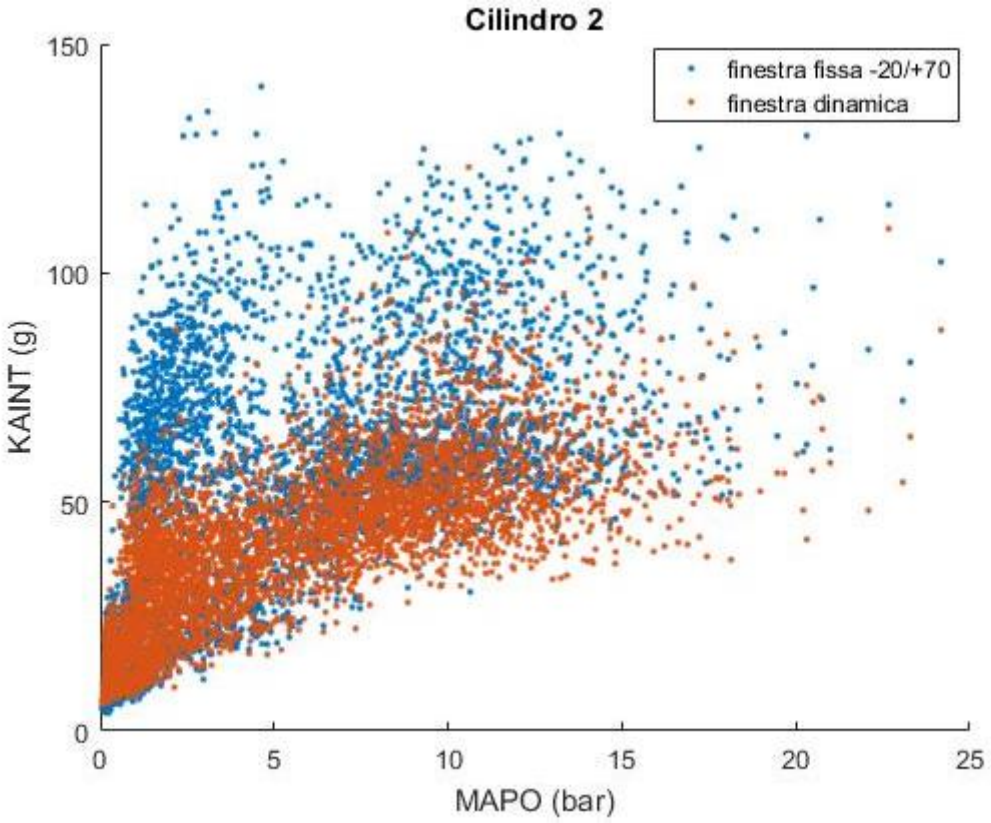


Figura 130: confronto sul cilindro 2 fra finestra dinamica e finestra fissa

Il miglioramento è significativo soprattutto nel caso del cilindro 2, caratterizzato in precedenza da una correlazione piuttosto bassa. L'implementazione di una procedura di calcolo di questo tipo su un sistema di indicating Real-Time andrebbe comunque valutata sui tempi di esecuzione necessari.

5.5 CONCLUSIONI

Da questa prima campagna sperimentale ed analisi è possibile concludere che l'implementazione di un controllo knock basato su accelerometri per applicazioni on-board è fattibile. Infatti si sono ottenute delle correlazioni ampiamente al di sopra dello 0.7 a fronte di un piazzamento dei sensori non pianificato con attenzione ed una scelta del tipo di sensore non particolarmente attenta. Vista la sensibilità degli indici accelerometrici al rumore meccanico ed al mezzo di trasmissione delle vibrazioni, focalizzando l'attenzione sulla scelta del sensore e del fissaggio è sicuramente possibile ottenere risultati migliori in termini di correlazione con i sensori di pressione.

Capitolo 6
ANALISI ROTAZIONALI GIUNTO ELASTICO

Il giunto elastico è un elemento molto sollecitato durante i test al banco e la sua rottura comporta lo stop delle prove oltre ad avere un costo piuttosto elevato. Per questo è stata svolta un'analisi mirata a caratterizzare il comportamento del giunto stesso e ad identificare eventuali risonanze incontrate nel range di funzionamento del motore. È infatti importante evitare la permanenza in tali zone per evitare surriscaldamenti dell'elemento in gomma che possono portare alla rottura anticipata del giunto elastico.

L'analisi svolta è principalmente di tipo rotazionale sulla velocità istantanea del motore ed è basata sull'impego di diagrammi di Campbell (RPM-frequenza) ottenuti applicando la trasformata di Fourier su piccoli buffer di registrazione. Le condizioni ideali per identificare la risposta del sistema è l'esecuzione di rampe di giri ampie a carico elevato, dalle quali sarebbe anche possibile mappare l'ampiezza delle oscillazioni in assenza di guasti ed utilizzare queste informazioni per fini diagnostici in seguito. Infatti in futuro l'idea è quella di implementare al banco un sistema di monitoraggio in grado di accorgersi dell'aumento delle oscillazioni di RPM corrispondenti ai principali ordini ed intervenire con uno stop prima della rottura definitiva.

Il primo set di dati analizzato è quello relativo ad una rampa in WOT da 5500 a 12500 giri. L'acquisizione è stata fatta dal PC di banco e l'analisi è stata concentrata sulla velocità di rotazione istantanea del motore rilevata dalla ECU e sotto-campionata a 500 kHz. La frequenza di campionamento così bassa comporta inevitabilmente la presenza di aliasing per il contenuto infrequenza oltre i 250 Hz, ben visibile nel diagramma successivo, ma che in larga misura non inficia la validità dell'analisi. L'STFT eseguita per ottenere la cascata di spettri è stata svolta su buffer di 500 elementi con un overlap di 375 elementi, e consente quindi una risoluzione spettrale

di 2 Hz. Per migliorare l'analisi sono inoltre state eliminate la componente media e quella lineare dovuta alla rampa crescente di giri.

Dal diagramma di Campbell seguente risultano evidenti le seguenti risonanze:

- Intorno ai 7000 rpm sull'ordine 1
- Intorno a 11500 rpm sull'ordine 1
- Intorno a 5750 rpm sull'ordine 1/2

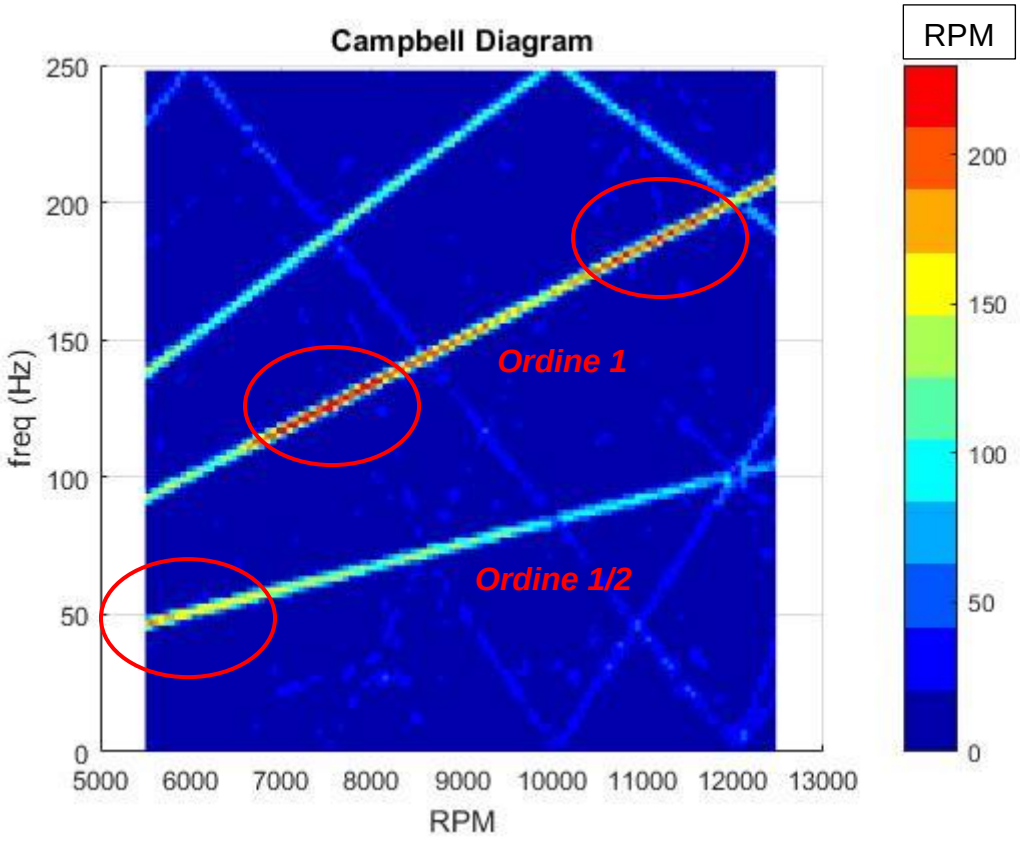


Figura 131: Campbell diagram 2D

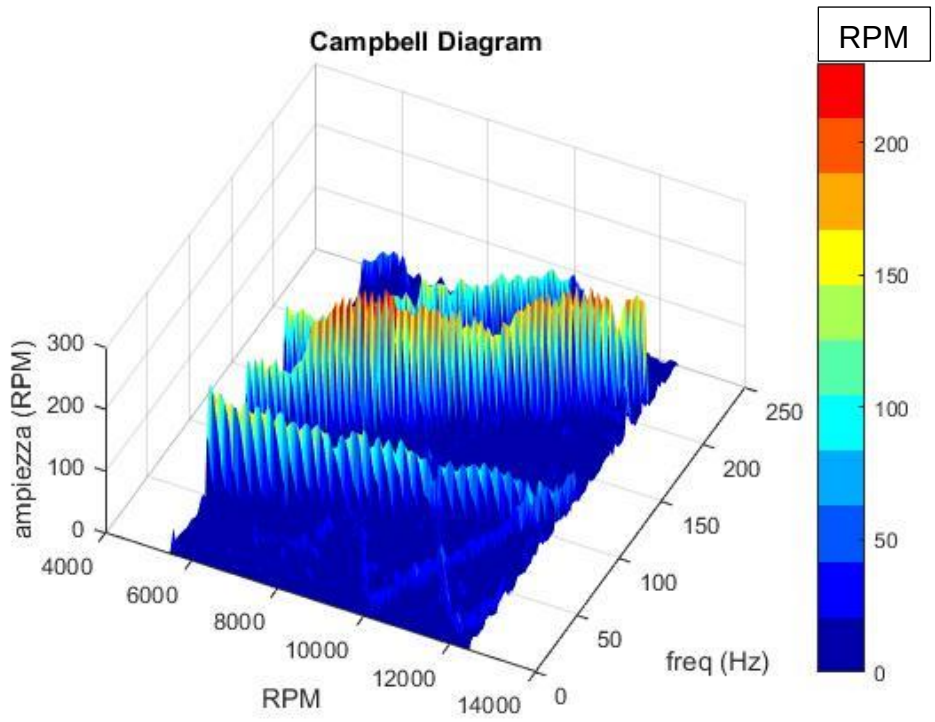


Figura 132: Campbell diagram 3D

Per quanto riguarda le oscillazioni di coppia prodotte sulla macchina elettrica è ben visibile il picco in corrispondenza della risonanza sull'ordine 1 introno a 7000 rpm.

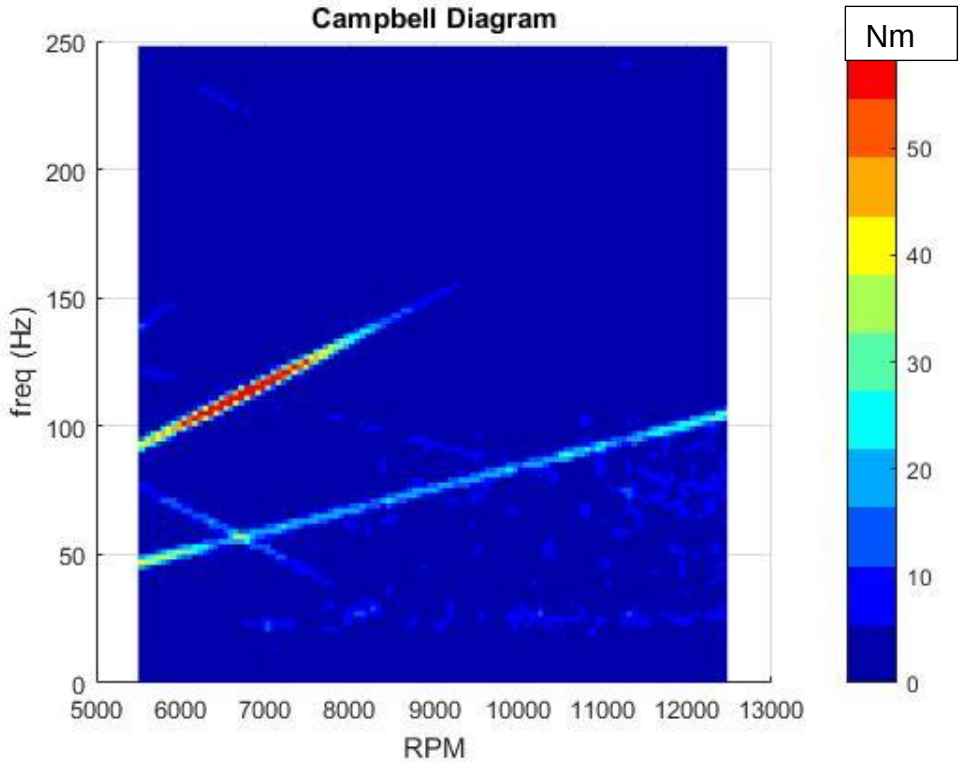


Figura 133: Campbell digram coppia frenante della macchina elettrica

Utilizzando invece i dati provenienti dal sistema OBI si possono ottenere ulteriori informazioni in quanto il segnale è campionato su base angolo attraverso una ruota fonica 24-2. Eseguendo quindi la stessa analisi STFT sui tempi dente convertiti in RPM istantanei si ottiene una serie di spettri degli ordini, dai quali si può risalire allo spettro in frequenza semplicemente moltiplicando per la velocità media corrispondente al buffer analizzato.

$$RPM = \frac{60}{dt\_tooth * \ n\_denti}$$

Il passaggio sulla cava della ruota fonica implica una caduta del valore di rpm rilevato, che va sostituito con il valore interpolato fra gli rpm del dente precedente e quelli del successivo. La cava viene identificata quando il tempo dente è almeno due volte superiore a quello del dente precedente (in corrispondenza della cava mancano 2 denti).

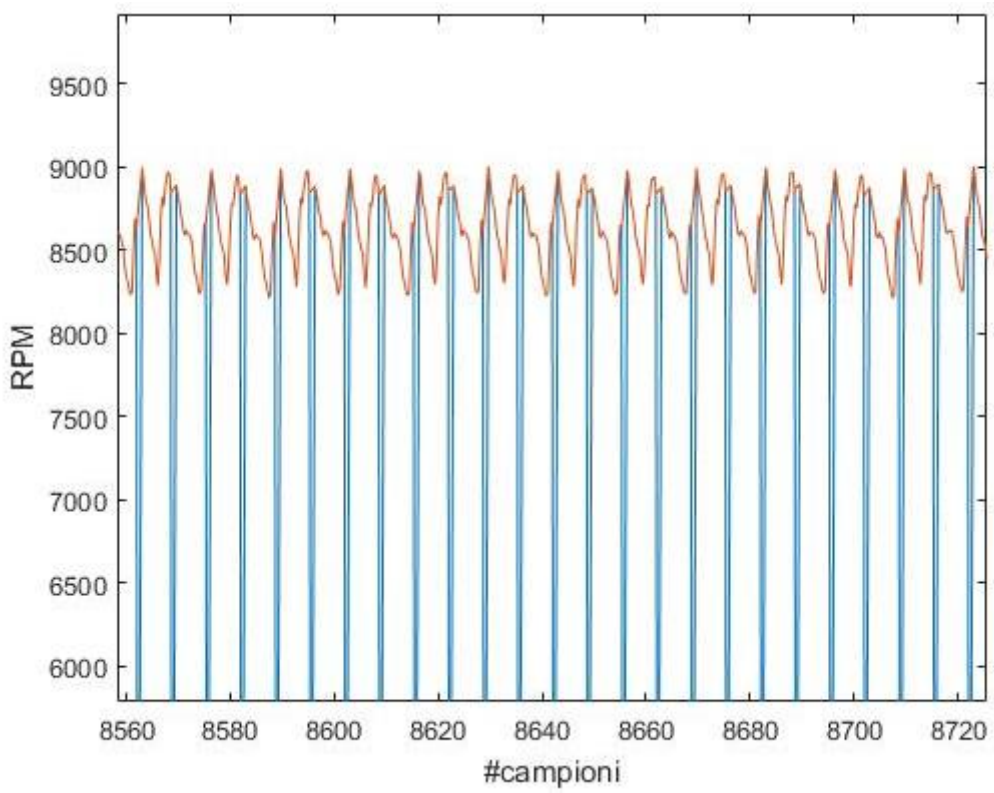


Figura 134: interpolazione sulla velocità istantanea per ricostruire il corretto valore in corrispondenza della cava della ruota fonica

Il set di dati analizzato non è lo stesso, ma proviene dalle rampe di salita effettuate durante una CdP, durante le quali le farfalle sono aperte linearmente con i giri, quindi l'effetto del carico è smorzato ai giri più bassi. Il buffer è di 40 campioni e l'overlap di 20, consentendo una risoluzione d'ordine pari a 0.025 (armoniche di ciclo).

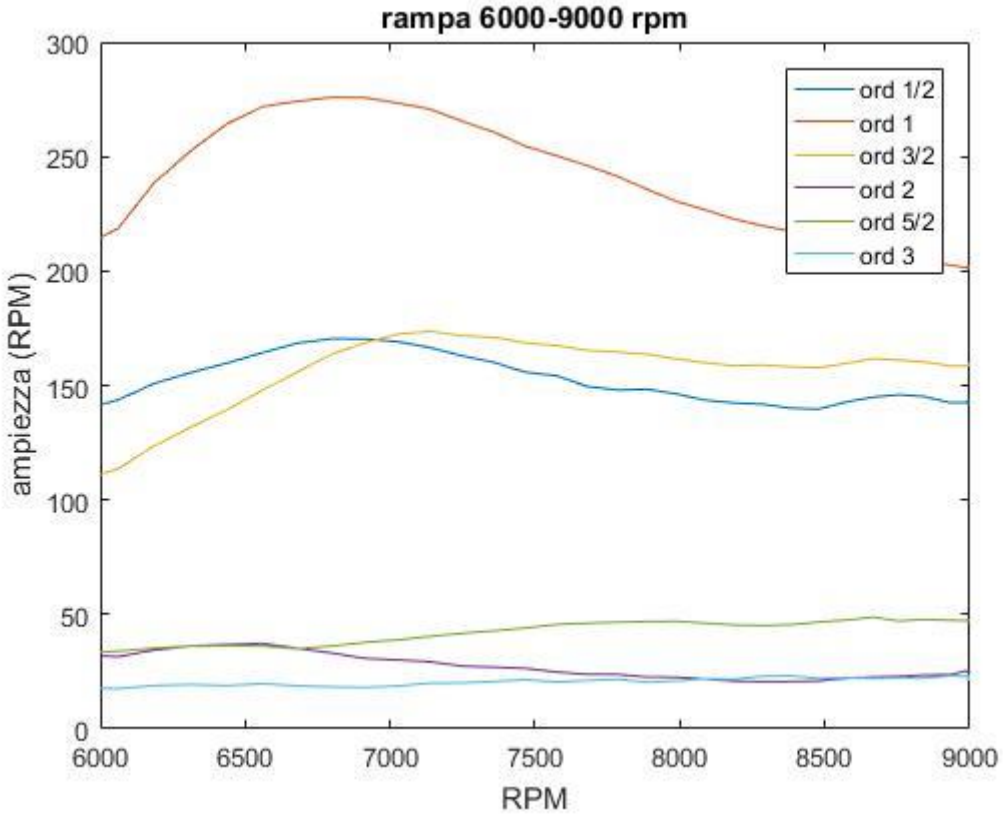


Figura 135: plot dei principali ordini in funzione dei giri

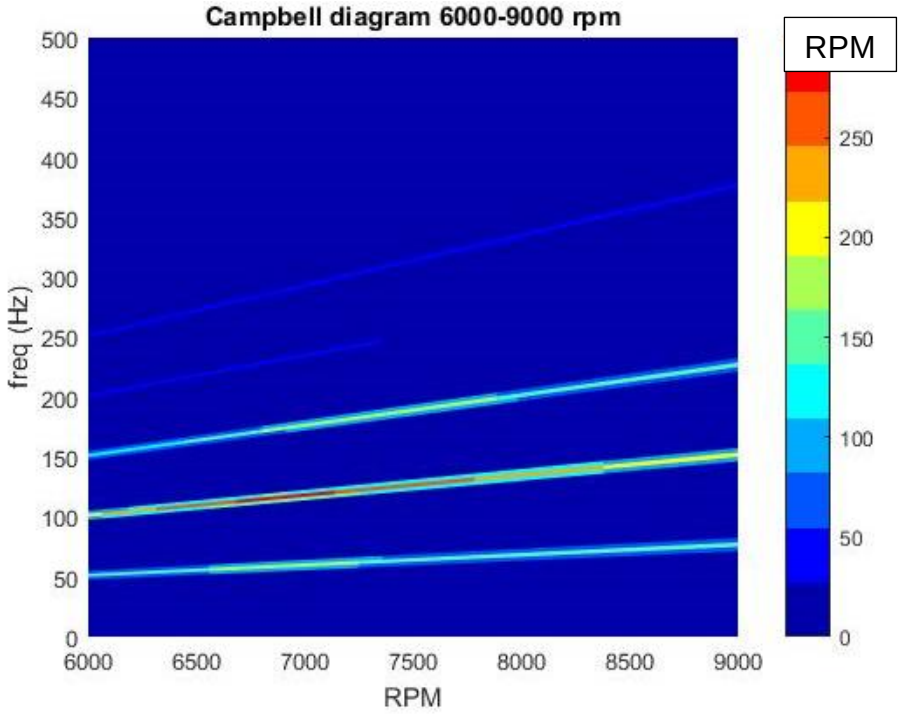


Figura 136: diagramma di Campbell corrispondente

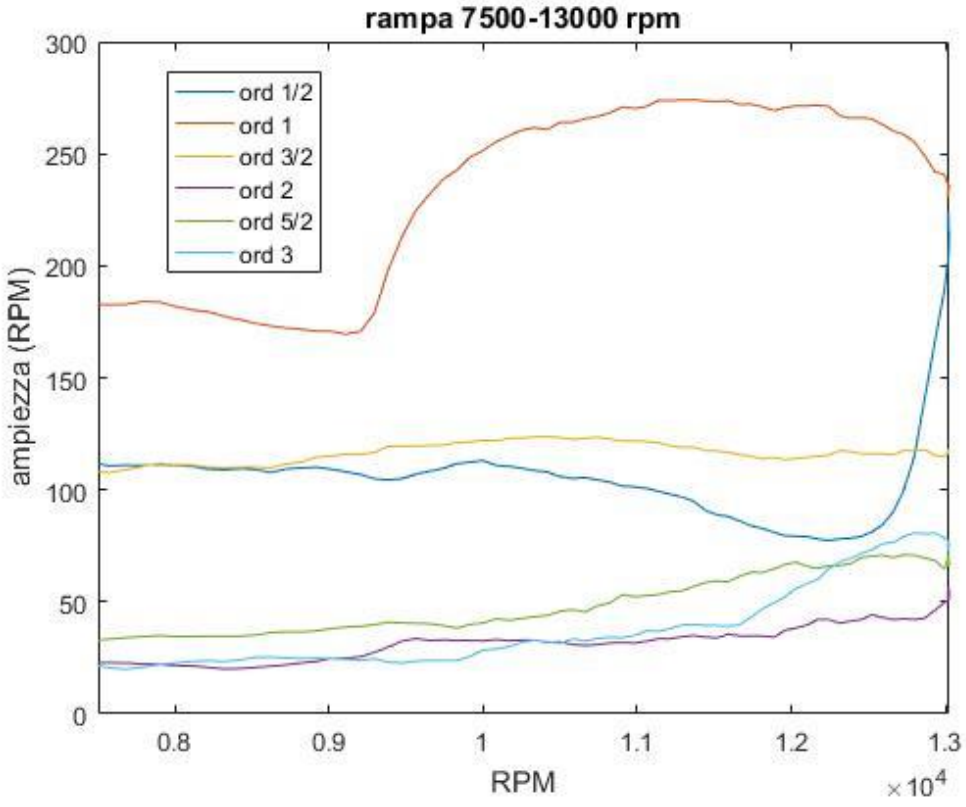


Figura 137: plot dei principali ordini in funzione dei giri

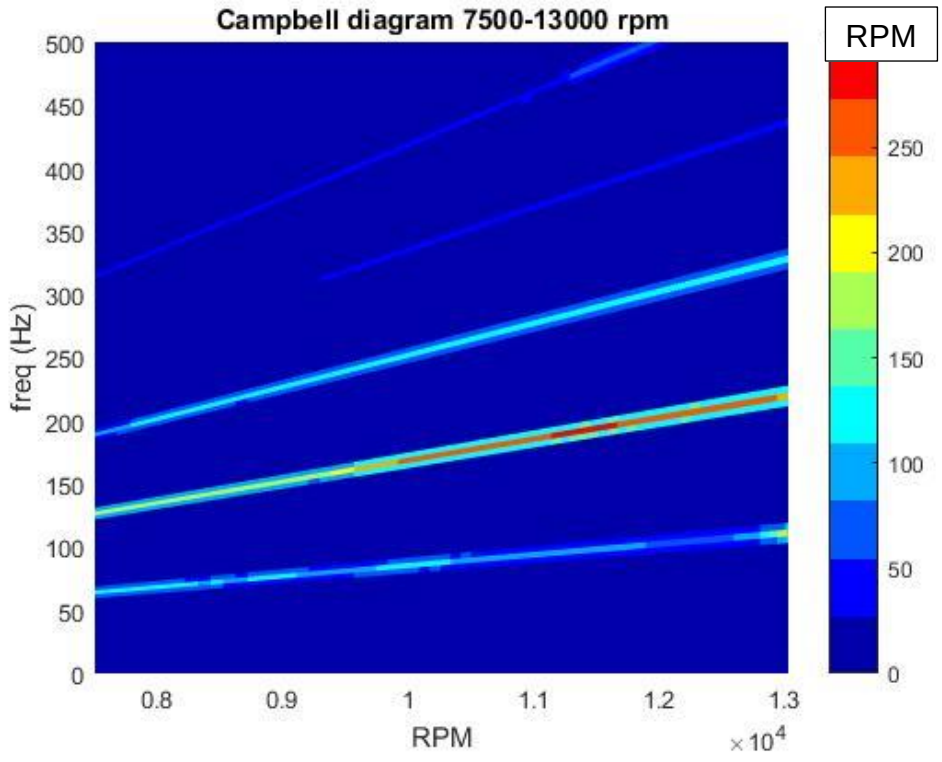


Figura 138: diagramma di Campbell corrispondente

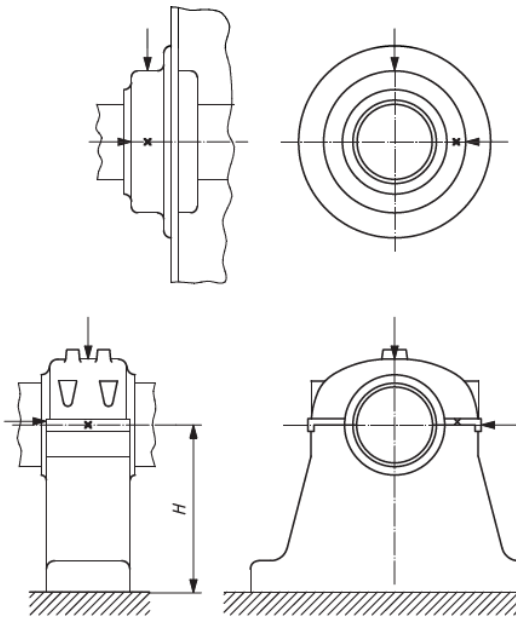
In conclusione sono confermate le seguenti zone di risonanza:

- Intorno ai 110-120 Hz (intorno ai 6500-7000 rpm sull'ordine 1 ed oltre i 13000 rpm per l'ordine ½)
- Intorno a 190 Hz (circa 11500 rpm sull'ordine 1)

Saranno ovviamente presenti i rispettivi picchi anche a regimi inferiori per gli ordini più alti, cosa che può essere visibile durante gli avviamenti, unica condizione in cui il motore rimane al di sotto dei 5000 rpm.

Capitolo 7
ANALISI DELLE VIBRAZIONI SULLA MACCHINA ELETTRICA

Con l'obiettivo di valutare l'accettabilità delle vibrazioni subite dalla macchina elettrica è stata condotta una prova sperimentale di 5 giri simulati con accelerometro installato. La normativa di riferimento per la valutazione delle vibrazioni su macchine rotanti è la ISO 20816 - 2016 dal titolo: "Mechanical vibration — Measurement and evaluation of machine vibration", che sostituisce la precedente IO 1086. In particolare la parte 3 è stata quella maggiormente di nostro interesse, in quanto è dedicata alle macchine industriali con potenza superiore a 15 kW e velocità nominale fra i 120 e i 15.000 giri/min, definizione entro la quale ricade la macchina del banco prova: Oswald QDI22.3-2FI con potenza nominale pari a 221 kW e regime massimo di 10000 rpm. Il posizionamento degli accelerometri è prescritto in corrispondenza degli alloggiamenti dei cuscinetti, con orientamento radiale rispetto agli stessi, e la macchina ha delle filetta ture M8 predisposte per questa applicazione.



NOTE Measurements to be made at the bearing housing or, if not accessible, then at positions as close as possible to the bearing housings which provide adequate sensitivity to the machine dynamic forces.

Figura 139: ISO 20816 posizionamento accelerometri

Foto piazzamento accelerometro.

La normativa ISO 20816 indica da tabella un limite tollerabile massimo di 7,1 mm/s di velocità RMS, per una macchina elettrica come quella utilizzata. Va comunque specificato che i limiti in questione sono definiti per un funzionamento in stazionario, mentre le condizioni di funzionamento d'interesse sono estremamente dinamiche.

Table A.2 — Classification of vibration severity zones for machines of Group 2: Medium-sized machines with rated power above 15 kW up to and including 300 kW; electrical machines with shaft height 160 mm $\leqslant H <$ 315 mm

Support class	Zone boundary	r.m.s. displacement µm	r.m.s. velocity mm/s
Rigid	A/B	22	1,4
	B/C	45	2,8
	C/D	71	4,5
Flexible	A/B	37	2,3
	B/C	71	4,5
	C/D	113	7,1

NOTE 1 These values apply to radial vibration measurements on all bearings, bearing pedestals, or housings of machines and to axial vibration measurements on thrust bearings under steady-state operating conditions at rated speed or within the specified speed range. They do not apply when the machine is undergoing a transient condition (i.e. changing speed or load).

Figura 140: tabella normativa ISO 20816

In normativa è altresì specificato che è possibile valutare lo stato operativo della macchina considerando lo scostamento della misura attuale da un riferimento precedente di un certo valore di velocità RMS.

L'accelerometro utilizzato è stato un Monitran 1185-CQ25 con uscita in corrente 4-20 mA proporzionale alla velocità RMS. L'acquisizione è avvenuta attraverso un VI Labview su PC host ed un NI cDAQ 9191 con scheda d'acquisizione NI 9203.

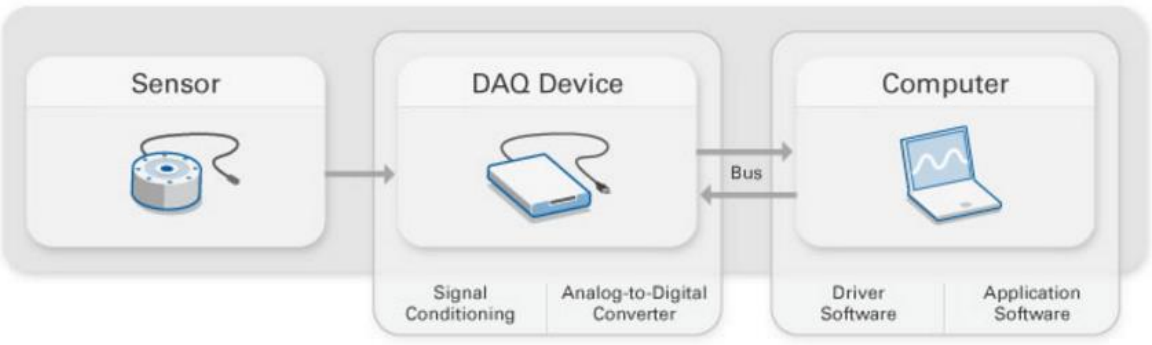
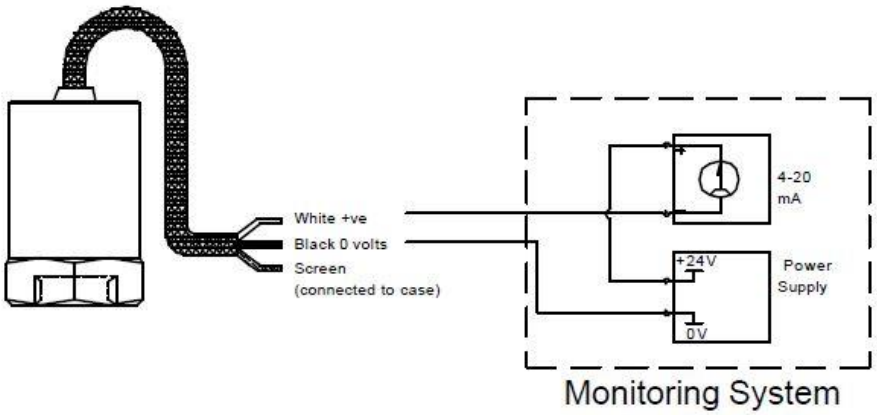


Figura 141: setup d'acquisizione

System Connection



0-25 mm/sec RMS

Technical Specification

Output Current	4-20mA DC proportional to RMS velocity (mm/s)
Velocity Range	See table below
Frequency Response	2 Hz to 1 kHz ±10 %
Dynamic Range	50 g peak
Mounted Resonance	5 kHz min
Isolation	Base isolated
Operating Temp Range	-25 to 90 ° C
Temperature Sensitivity	0.08 %/° C
Transverse Sensitivity	Less than 5%
Supply Voltage	12-32 Volts DC (for 4-20mA)

Figura 142: datasheet accelerometro Monitran 1185CQ-25

NI 9203 Datasheet

8-Channel, ±20 mA, 16-Bit Analog Input Module

- 8 channels, 200 kS/s current input
- ±20 mA, 0 mA to 20 mA programmable input ranges; 16-bit resolution
- NIST-traceable calibration
- Screw-terminal or spring-terminal connectivity
- 250 V_{rms}, CAT II bank isolation
- -40 °C to 70 °C operating range, 5 g vibration, 50 g shock

Figura 143: datasheet scheda d'acquisizione NI 9203

Durante la prova di 5 giri registrata, nonostante il livello di vibrazione non sia particolarmente preoccupante considerando la dinamicità della prova, si riscontra un andamento crescente del livello RMS in corrispondenza degli stessi eventi per giri successivi.

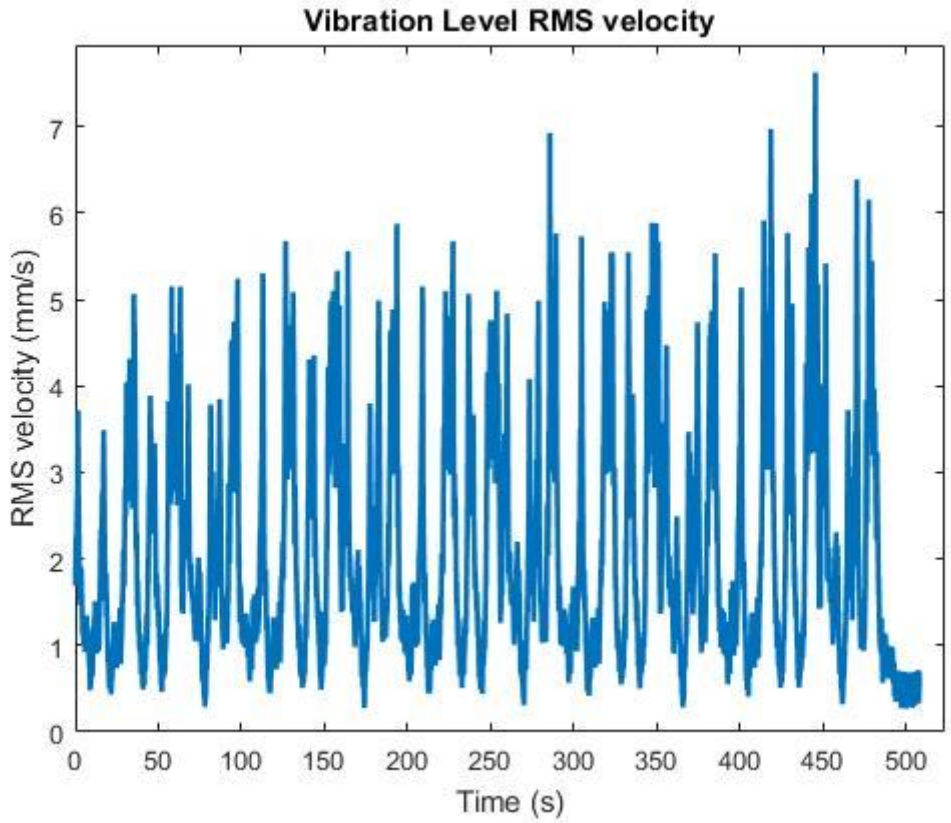


Figura 144: valore di velocità RMS rilevato durante la registrazione di 5 giri

Questa osservazione può essere interessante perché potrebbe essere il sintomo di un qualche cambiamento nello stato di un componente messo sotto stress durante la prova, ad esempio potrebbe essere causato dall'aumento di temperatura della parte in gomma del giunto elastico posto fra motore e macchina elettrica. La tematica necessita di approfondimento.

APPENDICE A:
CODICE DI SIMULAZIONE CONDITION MONITORING

COSTRUZIONE DELLA MAPPA

```
[fn,pn]=uigetfile('*.mat','Select File','multiselect','on');
% if fn==0,return;end
if ~iscell(fn)
    if fn==0,return;end
    fn={fn};
end

%%%%%%% Per mappare utilizzare 1 solo file con i primi giri %%%%%%%%

%% DATI CAMPIONAMENTO E INIZIALIZZAZIONE MAPPA %%
Fs=200000 %Hz f_campionamento
buff= 200/1000*Fs %%% selezionare il buffer della registrazione da
analizzare
over= 66/1000*Fs %%%overlap campioni analizzati

tipo_mappa='200_fitta_over_cond';

RPM_grid=linspace(5000,13500,(13500-5000)/250+1);
TWGRDM_grid=[0 2 5 8 12 18 25 35 45 55 65 75 85 95]

weights=zeros(length(TWGRDM_grid),length(RPM_grid));
MAP=zeros(length(TWGRDM_grid),length(RPM_grid),buff/2);

for iFile=1:length(fn)

%% CARICAMENTO DATI %%

[pathstr,nome_file,ext] = fileparts([pn,fn{iFile}]);
load([pn,fn{iFile}]);


%%

N_tot=length(bas);
N=buff;
wind=hanning(N)'/mean(hanning(N));

start=13;
fine=13;%%% secondo a cui si vuole far iniziare l'analisi
inf=-buff+over+1+start*Fs;

tic

for h=1:(floor((N_tot-buff-start*Fs-fine*Fs)/buff)*(1/(1-over/buff)))
```

```
inf=inf+buff-over;
sup=inf+buff-1;
range=[inf:1:sup];

X=bas(range);
%X_t=detrend(X_tt,0); %%% eliminazione componente media
%X=detrend(X_t,1); %%% eliminazione componente lineare

FTX=fft(X.*wind,N); tipo='ampiezza';

S=(abs(FTX))/N;  tipo='ampiezza';
S_p=(abs(FTX).^2)/N; tipo_p='potenza';

f=(0:Fs/N:Fs-Fs/N);

%S=S(1:N/2)*2; S(1)=0; %elimino la seconda meta' delle stime emoltiplico *2
tranne sulla componente media
S=S_p(1:N/2)*2; S_p(1)=0;
f=f(1:N/2); %elimino la seconda meta' dele frequenze

%% VARIABILI DA SALVARE

time_spectrum(h)=time(range(1));%%%tempo corrispondente di ogni fft
spectrum(h,:)=S; %%% varaibile che contiene lo spettrogramma


%% RIEMPIMENTO MAPPA %%
RPM_mean(h)=mean(RPM(range)); %%%giri medi sul buffer
TWGRDM_mean(h)=mean(TWGRDM_sync(range)); %%%farfalla media sul buffer

[~,idx_RPM]=min(abs(RPM_grid-RPM_mean(h)));
[~,idx_TWGRDM]=min(abs(TWGRDM_grid-TWGRDM_mean(h)));

coeff_RPM=(RPM_grid(idx_RPM)-RPM_mean(h))/RPM_grid(idx_RPM);
coeff_TWGRDM=(TWGRDM_grid(idx_TWGRDM)-
TWGRDM_mean(h))/TWGRDM_grid(idx_TWGRDM);

if ~isfinite(coeff_RPM)==1
    coeff_RPM=0;
end
if ~isfinite(coeff_TWGRDM)==1
    coeff_TWGRDM=0;
end

corr=1+coeff_RPM+coeff_TWGRDM;

S=S.*corr;

plot(h,RPM_mean(h),'b.'),hold on
if max(RPM(range))-min(RPM(range))<100 && max(TWGRDM_sync(range))-
min(TWGRDM_sync(range))<10

    n=weights(idx_TWGRDM,idx_RPM);

MAP(idx_TWGRDM,idx_RPM,:)=(reshape(MAP(idx_TWGRDM,idx_RPM,:),1,buff/2)*n+S)
./(n+1);
```

```
    weights(idx_TWGRDM,idx_RPM)=weights(idx_TWGRDM,idx_RPM)+1;
    plot(h,RPM_mean(h),'r.')
end

% figure,hold on,
% plot(S)
% plot(reshape(MAP(idx_TWGRDM,idx_RPM,:),1,5000))
% RPM_mean(h)
% TWGRDM_mean(h)
% low
% pause
% close

%%
% %%%plot singola fft
% figure(2)
% subplot(2,1,1)
% plot(t,X_t),hold on
% subplot(2,1,1)
% plot(t,X)
% subplot(2,1,2)
% plot(f_FT,S)
% pause
% close 2

end

save(['E:\analisi_vibrazioni\',nome_file,
'_MAP_',tipo_mappa,'.mat'],'MAP','TWGRDM_grid','RPM_grid','weights','time_s
pectrum','spectrum','f','RPM_mean','TWGRDM_mean')

clearvars time_spectrum spectrum RPM_mean TWGRDM_mean
end
```

COSTRUZIONE DELLE TOLLERANZE

```
clear all
close all
clc

[fn,pn]=uigetfile('*data.mat','Select File','multiselect','on');
% if fn==0,return;end
if ~iscell(fn)
    if fn==0,return;end
    fn={fn};
end

%%%%%%% Per mappare utilizzare 1 solo file con i primi giri %%%%%%%%

%% DATI CAMPIONAMENTO E INIZIALIZZAZIONE MAPPA %%
Fs=200000 %Hz f_campionamento
buff= 50/1000*Fs %%% selezionare il buffer della registrazione da
analizzare
over= 0/1000*Fs %%%overlap campioni analizzati

tipo_mappa='50_fitta_over';
```

```
%%%% caricare file MAPPA %%%%

load(['E:\analisi_vibrazioni\170724_161917_170724_SBK41_11_01_D_100_data_MA
P_',tipo_mappa,'.mat'],'MAP','weights','RPM_grid','TWGRDM_grid')

Tol=zeros(length(TWGRDM_grid),length(RPM_grid));
weights_tol=zeros(length(TWGRDM_grid),length(RPM_grid));
for iFile=1:length(fn)

%% CARICAMENTO DATI %%

[pathstr,nome_file,ext] = fileparts([pn,fn{iFile}]);
load([pn,fn{iFile}]);

%%

N_tot=length(bas);
N=buff;
wind=hanning(N)'/mean(hanning(N));

start=13;
fine=13;%%% secondo a cui si vuole far iniziare l'analisi
inf=-buff+over+1+start*Fs;

tic
for h=1:(floor((N_tot-buff-start*Fs-fine*Fs)/buff)*(1/(1-over/buff)))

inf=inf+buff-over;
sup=inf+buff-1;
range=[inf:1:sup];

X=bas(range);
%X_t=detrend(X_tt,0); %%% eliminazione componente media
%X=detrend(X_t,1); %%% eliminazione componente lineare

FTX=fft(X.*wind,N); tipo='ampiezza';

S=(abs(FTX))/N;  tipo='ampiezza';
S_p=(abs(FTX).^2)/N; tipo_p='potenza';

f_FT=(0:Fs/N:Fs-Fs/N);

%S=S(1:N/2)*2; S(1)=0; %elimino la seconda meta' delle stime emoltiplico *2
(per avere la stessa potenza/ampiezza sullo spettro a singolo lato)
S=S_p(1:N/2)*2; S_p(1)=0;
f_FT=f_FT(1:N/2); %elimino la seconda meta' dele frequenze

%% TRENDINDEX %%

RPM_mean(h)=mean(RPM(range)); %%%giri medi sul buffer
TWGRDM_mean(h)=mean(TWGRDM_sync(range)); %%%farfalla media sul buffer

%clearvars -except MAP RPM_gri TWGRDM_grid weights Trend_Index S f_FT pn fn

[~,idx_RPM]=min(abs(RPM_grid-RPM_mean(h)));
[~,idx_TWGRDM]=min(abs(TWGRDM_grid-TWGRDM_mean(h)));
```

```
coeff_RPM=-(RPM_grid(idx_RPM)-RPM_mean(h))/RPM_grid(idx_RPM);
coeff_TWGRDM=-(TWGRDM_grid(idx_TWGRDM)-
TWGRDM_mean(h))/TWGRDM_grid(idx_TWGRDM);

if ~isfinite(coeff_RPM)==1
    coeff_RPM=0;
end
if ~isfinite(coeff_TWGRDM)==1
    coeff_TWGRDM=0;
end

corr=1+coeff_RPM+coeff_TWGRDM;

Trend_Index_40k_r(h)=sum(abs((S(2:0.4*buff/2)./reshape(MAP(idx_TWGRDM,idx_R
PM,2:0.4*buff/2),1,length(2:0.4*buff/2)))-1))/length(2:0.4*buff/2); %indice
calcolato come rapporto: banda fino a 40kHz
Trend_Index_5k_r(h)=sum(abs((S(2:0.05*buff/2)./reshape(MAP(idx_TWGRDM,idx_R
PM,2:0.05*buff/2),1,length(2:0.05*buff/2)))-1))/length(2:0.05*buff/2);
%indice calcolato come rapporto: banda fino a 5kHz
Trend_Index_2k_r(h)=sum(abs((S(2:0.02*buff/2)./reshape(MAP(idx_TWGRDM,idx_R
PM,2:0.02*buff/2),1,length(2:0.02*buff/2)))-1))/length(2:0.02*buff/2);
%indice calcolato come rapporto: banda fino a 2kHz (10a armonica a
12000rpm)

%% MAPPA DELLE TOLLERANZE %%

    n=weights_tol(idx_TWGRDM,idx_RPM);

Tol_40k_r(idx_TWGRDM,idx_RPM)=((Tol(idx_TWGRDM,idx_RPM)*n+Trend_Index_40k_r
(h))/(n+1));

Tol_5k_r(idx_TWGRDM,idx_RPM)=((Tol(idx_TWGRDM,idx_RPM)*n+Trend_Index_5k_r(h
))/(n+1));

Tol_2k_r(idx_TWGRDM,idx_RPM)=((Tol(idx_TWGRDM,idx_RPM)*n+Trend_Index_2k_r(h
))/(n+1));
    weights_tol(idx_TWGRDM,idx_RPM)=weights_tol(idx_TWGRDM,idx_RPM)+1;

%%
% %%%plot singola fft
% figure(2)
% subplot(2,1,1)
% plot(t,X_t),hold on
% subplot(2,1,1)
% plot(t,X)
% subplot(2,1,2)
% plot(f_FT,S)
% pause
% close 2

end

save(['E:\analisi_vibrazioni\',nome_file, '_TOL_', tipo_mappa
,'.mat'],'RPM_mean','TWGRDM_mean','Tol_40k_r','Tol_5k_r','Tol_2k_r','weight
s_tol')
%,'Trend_Index','Trend_Index_2_1000','Trend_Index_2_500','Trend_Index_2_250
','Trend_Index_2_125','Trend_Index_2_50','Trend_Index_20_250'

end
```

CALCOLO DEL TRENDINDEX

```
clear all
close all
clc

[fn,pn]=uigetfile('*data.mat','Select File','multiselect','on');
% if fn==0,return;end
if ~iscell(fn)
    if fn==0,return;end
    fn={fn};
end

%%%%%%% Per mappare utilizzare 1 solo file con i primi giri %%%%%%%%

%% DATI CAMPIONAMENTO E INIZIALIZZAZIONE MAPPA %%
Fs=200000 %Hz f_campionamento
buff= 50/1000*Fs %%% selezionare il buffer della registrazione da
analizzare
over= 0/1000*Fs %%%overlap campioni analizzati

%%%% caricare file MAPPA %%%%

tipo_mappa='50_fitta_over';

load(['E:\analisi_vibrazioni\170724_161917_170724_SBK41_11_01_D_100_data_MA
P_',tipo_mappa,'.mat'],'MAP','RPM_grid','TWGRDM_grid')
load(['E:\analisi_vibrazioni\170724_161917_170724_SBK41_11_01_D_100_data_TO
L_',tipo_mappa,'.mat'],'Tol_40k_r','Tol_5k_r','Tol_2k_r')

RPM_mean_tot=[];
TWGRDM_mean_tot=[];
RMS_tot=[];

Trend_Index_40k_r_tolg_tot=[];
Trend_Index_5k_r_tolg_tot=[];
Trend_Index_2k_r_tolg_tot=[];

Trend_Index_40k_r_tol_tot=[];
Trend_Index_5k_r_tol_tot=[];
Trend_Index_2k_r_tol_tot=[];

Trend_Index_40k_r_tot=[];
Trend_Index_5k_r_tot=[];
Trend_Index_2k_r_tot=[];

for iFile=1:length(fn)

%% CARICAMENTO DATI %%

clearvars RPM TWGRDM_sync bas time RPM_mean TWGRDM_mean RMS
Trend_Index_40k_r Trend_Index_5k_r Trend_Index_2k_r Trend_Index_40k_r_tolg
```

```
Trend_Index_5k_r_tolg Trend_Index_2k_r_tolg Trend_Index_40k_r_tol
Trend_Index_5k_r_tol Trend_Index_2k_r_tol

[pathstr,nome_file,ext] = fileparts([pn,fn{iFile}]);
load([pn,fn{iFile}])
[pn,fn{iFile}]


%%

N_tot=length(bas);
N=buff;
wind=hanning(N)'/mean(hanning(N));

an_40k=5:0.4*buff/2;
an_5k=5:0.05*buff/2;
an_2k=5:0.02*buff/2;

start=0;
fine=0;%%% secondo a cui si vuole far iniziare l'analisi
inf=-buff+over+1;
sup=start*Fs;
tic
for h=1:(floor((N_tot-buff-start*Fs-fine*Fs)/buff)*(1/(1-over/buff)))

inf=inf+buff-over;
sup=inf+buff-1;
range=[inf:1:sup];

X=bas(range);
%X_t=detrend(X_tt,0); %%% eliminazione componente media
%X=detrend(X_t,1); %%% eliminazione componente lineare

FTX=fft(X.*wind,N); tipo='ampiezza';

S=(abs(FTX))/N;  tipo='ampiezza';
%S_p=(abs(FTX).^2)/N; tipo_p='potenza';

f_FT=(0:Fs/N:Fs-Fs/N);

S=S(1:N/2)*2; S(1)=0; %elimino la seconda meta' delle stime emoltiplico *2
(per avere la stessa potenza/ampiezza sullo spettro a singolo lato)
%S=S_p(1:N/2)*2; S_p(1)=0;
f_FT=f_FT(1:N/2); %elimino la seconda meta' dele frequenze

%% TRENDINDEX %%

RPM_mean(h)=mean(RPM(range)); %%%giri medi sul buffer
TWGRDM_mean(h)=mean(TWGRDM_sync(range)); %%%farfalla media sul buffer

%clearvars -except MAP RPM_gri TWGRDM_grid weights Trend_Index S f_FT pn fn

[~,idx_RPM]=min(abs(RPM_grid-RPM_mean(h)));
[~,idx_TWGRDM]=min(abs(TWGRDM_grid-TWGRDM_mean(h)));

coeff_RPM=-(RPM_grid(idx_RPM)-RPM_mean(h))/RPM_grid(idx_RPM);
```

```
coeff_TWGRDM=-(TWGRDM_grid(idx_TWGRDM)-
TWGRDM_mean(h))/TWGRDM_grid(idx_TWGRDM);

if ~isfinite(coeff_RPM)==1
    coeff_RPM=0;
end
if ~isfinite(coeff_TWGRDM)==1
    coeff_TWGRDM=0;
end

corr=1+coeff_RPM+coeff_TWGRDM;

%tol_exceed=abs(S-
reshape(MAP(idx_TWGRDM,idx_RPM,:),1,N/2))>Tol(idx_TWGRDM,idx_RPM); %% 1 se
la tolleranza è ecceduta
%Trend_Index_tol(h)=sum(tol_exceed.*(abs(S-
reshape(MAP(idx_TWGRDM,idx_RPM,:),1,N/2))-Tol(idx_TWGRDM,idx_RPM)));

RMS(h)=rms(X);

Trend_Index_40k_r(h)=sum(abs((S(an_40k)./reshape(MAP(idx_TWGRDM,idx_RPM,an_
40k),1,length(an_40k)))-1))/length(an_40k); %indice calcolato come
rapporto: banda fino a 40kHz
Trend_Index_5k_r(h)=sum(abs((S(an_5k)./reshape(MAP(idx_TWGRDM,idx_RPM,an_5k
),1,length(an_5k)))-1))/length(an_5k); %indice calcolato come rapporto:
banda fino a 5kHz
Trend_Index_2k_r(h)=sum(abs((S(an_2k)./reshape(MAP(idx_TWGRDM,idx_RPM,an_2k
),1,length(an_2k)))-1))/length(an_2k); %indice calcolato come rapporto:
banda fino a 2kHz (10a armonica a 12000rpm)

%tolleranza globale
Trend_Index_40k_r_tolg(h)=(sum(abs((S(an_40k)./reshape(MAP(idx_TWGRDM,idx_R
PM,an_40k),1,length(an_40k)))-1))-Tol_40k_r(idx_TWGRDM,idx_RPM));
Trend_Index_5k_r_tolg(h)=(sum(abs((S(an_5k)./reshape(MAP(idx_TWGRDM,idx_RPM
,an_5k),1,length(an_5k)))-1))-Tol_5k_r(idx_TWGRDM,idx_RPM));
Trend_Index_2k_r_tolg(h)=(sum(abs((S(an_2k)./reshape(MAP(idx_TWGRDM,idx_RPM
,an_2k),1,length(an_2k)))-1))-Tol_2k_r(idx_TWGRDM,idx_RPM));

%tolleranza sulle componenti spettrali
Trend_Index_40k_r_tol(h)=sum(abs((S(an_40k)./reshape(MAP(idx_TWGRDM,idx_RPM
,an_40k),1,length(an_40k)))-
1).*(abs((S(an_40k)./reshape(MAP(idx_TWGRDM,idx_RPM,an_40k),1,length(an_40k
)))-1)>(Tol_40k_r(idx_TWGRDM,idx_RPM))/length(an_40k)));
Trend_Index_5k_r_tol(h)=sum(abs((S(an_5k)./reshape(MAP(idx_TWGRDM,idx_RPM,a
n_5k),1,length(an_5k)))-
1).*(abs((S(an_5k)./reshape(MAP(idx_TWGRDM,idx_RPM,an_5k),1,length(an_5k)))
-1)>(Tol_5k_r(idx_TWGRDM,idx_RPM))/length(an_5k)));
Trend_Index_2k_r_tol(h)=sum(abs((S(an_2k)./reshape(MAP(idx_TWGRDM,idx_RPM,a
n_2k),1,length(an_2k)))-
1).*(abs((S(an_2k)./reshape(MAP(idx_TWGRDM,idx_RPM,an_2k),1,length(an_2k)))
-1)>(Tol_2k_r(idx_TWGRDM,idx_RPM))/length(an_2k)));

spectrum(h,:)=S;
idx(h,1)=idx_TWGRDM;
idx(h,2)=idx_RPM;

Trend_Index_40k(h)=sum(abs((S(an_40k)-
reshape(MAP(idx_TWGRDM,idx_RPM,an_40k),1,length(an_40k)))-1)); %indice
calcolato come somma: banda fino a 40kHz
```

```
Trend_Index_5k(h)=sum(abs((S(an_5k)-
reshape(MAP(idx_TWGRDM,idx_RPM,an_5k),1,length(an_5k)))-1)); %indice
calcolato come somma: banda fino a 5kHz
Trend_Index_2k(h)=sum(abs((S(an_2k)-
reshape(MAP(idx_TWGRDM,idx_RPM,an_2k),1,length(an_2k)))-1)); %indice
calcolato come somma: banda fino a 2kHz

% if isfinite(Trend_Index_5k_r(h)) && Trend_Index_5k_r(h)>50
% %% PLOT
%
% %figure(1)
%
%plot(abs(S(an_5k)./reshape(MAP(idx_TWGRDM,idx_RPM,an_5k),1,length(an_5k)))
)
%
% figure(2),hold on
% plot(f_FT,S),plot(f_FT,reshape(MAP(idx_TWGRDM,idx_RPM,:),1,buff/2))
% title(['RPM: ',num2str(RPM_grid(idx_RPM)),'-',num2str(RPM_mean),' TWGRDM:
',num2str(TWGRDM_grid(idx_TWGRDM)),'-',num2str(TWGRDM_mean)])
% xlabel('freq (Hz)')
% ylabel('ampiezza (g)')
% axis([0 1200 0 max(S(5:0.012*buff/2))+0.005])
% %plot(tol_exceed.*(abs(S-reshape(MAP(idx_TWGRDM,idx_RPM,:),1,N/2))-
Tol(idx_TWGRDM,idx_RPM)))
% pause
% %close 1
% close 2
% end
end

RPM_mean_tot=[RPM_mean_tot RPM_mean];
TWGRDM_mean_tot=[TWGRDM_mean_tot TWGRDM_mean];

RMS_tot=[RMS_tot RMS];

Trend_Index_40k_r_tot=[Trend_Index_40k_r_tot Trend_Index_40k_r];
Trend_Index_5k_r_tot=[Trend_Index_5k_r_tot Trend_Index_5k_r];
Trend_Index_2k_r_tot=[Trend_Index_2k_r_tot Trend_Index_2k_r];

Trend_Index_40k_r_tol_tot=[Trend_Index_40k_r_tol_tot
Trend_Index_40k_r_tol];
Trend_Index_5k_r_tol_tot=[Trend_Index_5k_r_tol_tot Trend_Index_5k_r_tol];
Trend_Index_2k_r_tol_tot=[Trend_Index_2k_r_tol_tot Trend_Index_2k_r_tol];

Trend_Index_40k_r_tolg_tot=[Trend_Index_40k_r_tolg_tot
Trend_Index_40k_r_tolg];
Trend_Index_5k_r_tolg_tot=[Trend_Index_5k_r_tolg_tot
Trend_Index_5k_r_tolg];
Trend_Index_2k_r_tolg_tot=[Trend_Index_2k_r_tolg_tot
Trend_Index_2k_r_tolg];

end

save(['E:\analisi_vibrazioni\INDEX_',tipo_mappa,'.mat'],'RPM_mean_tot','TWG
RDM_mean_tot','RMS_tot','Trend_Index_40k_r_tolg_tot','Trend_Index_5k_r_tolg
_tot','Trend_Index_2k_r_tolg_tot','Trend_Index_40k_r_tol_tot','Trend_Index_
5k_r_tol_tot','Trend_Index_2k_r_tol_tot','Trend_Index_40k_r_tot','Trend_Ind
ex_5k_r_tot','Trend_Index_2k_r_tot')
```

APPENDICE B:
CODICE DI CALCOLO DEGLI INDICI DI DETONAZIONE

```
close all
clc
%%
%%%caricare il file .mat creato con Fenice
%%%deve contenere:
%%%Time_Domain P_cyl + Acc
%%%Cycles_Domain rpm, MAPO, KINT

%%% solo cyl_1(OR)--> è il più detonante

%%

%%% filtro
Wn=4000/100000;%/(CyclesDomain.RPM.data(i)/60*3600); %%% calcolo
dell'ordine corrispondente alla f di taglio desiderata sul ciclo
[b,a] = butter(4,Wn,'high');

%Cycles_An.knock.indexes=1:length(Time_Group_RAW_RelativeAngleValue_Cyl2);

trova=find(Time_Group_RAW_RelativeAngleValue_Cyl2(1:end-1)>-
20&Time_Group_RAW_RelativeAngleValue_Cyl2(1:end-1)<70);
trova_ini=trova(find(Time_Group_RAW_RelativeAngleValue_Cyl2(trova-
1)+20<0));
trova_end=trova(find(Time_Group_RAW_RelativeAngleValue_Cyl2(trova+1)-
70>0));
trova_end=trova_end(1:end);

plot(Time_Group_RAW_RelativeAngleValue_Cyl2(trova_ini))%,'.')
figure
plot(Time_Group_RAW_RelativeAngleValue_Cyl2(trova_end))%,'.')

Time_Group_RAW_PcylRaw02_filt=filter(b,a,Time_Group_RAW_PcylRaw02);
Time_Group_RAW_Other04_filt=filter(b,a,Time_Group_RAW_Other04);

for i=1:length(trova_ini)

    wind=trova_ini(i):trova_end(i);

    knock_time.P=Time_Group_RAW_PcylRaw02_filt(wind);
    knock_time.A=Time_Group_RAW_Other04_filt(wind);

    knock_time.Pfilt=knock_time.P;%filter(b,a,knock_time.P);
    knock_time.Afilt=knock_time.A;%filter(b,a,knock_time.A);

    %%% ricalcolo degli indici
    knock_time.MAPO_an(i)= max(abs(knock_time.Pfilt));
    knock_time.MAAO_an(i)= max(abs(knock_time.Afilt));
```

```
    cumulative_P=cumsum(abs(knock_time.Pfilt))./length(wind);
    cumulative_A=cumsum(abs(knock_time.Afilt))./length(wind);

    knock_time.KPINT_an(i)=cumulative_P(end);
    knock_time.KAINT_an(i)=cumulative_A(end);

    %%% finestra dinamica e ricalcolo KINT accelerometrico


wind_dyn=[wind(find(cumulative_A>0.1*knock_time.KAINT_an(i),1)):wind(find(c
umulative_A>0.9*knock_time.KAINT_an(i),1))];


knock_time.KPINT_an_dyn(i)=sum(abs(Time_Group_RAW_PcylRaw01(wind_dyn)))/len
gth(wind_dyn);

knock_time.KAINT_an_dyn(i)=sum(abs(Time_Group_RAW_Other05(wind_dyn)))/lengt
h(wind_dyn);

end

figure, hold on
plot(knock_time.MAPO_an,knock_time.KAINT_an,'.')
plot(knock_time.MAPO_an,knock_time.KAINT_an_dyn,'.')
legend('finestra fissa -20/+70','finestra dinamica')

corrcoef(knock_time.MAPO_an,knock_time.KAINT_an_dyn)
corrcoef(knock_time.MAPO_an,knock_time.KAINT_an)
```

APPENDICE C:
CODICE DI CALCOLO PER L'ANALISI ROTAZIONALE

7.1 FILE DI BANCO CAMPIONATI SU BASE TEMPO

```
clear all
close all
clc

%%%% analisi canali ad 1kHz registrati da banco

addpath 'C:\Program Files\MATLAB\R2016b\extern function &
lib\uigetvariables'
addpath 'E:\SBK41_11\170721_091300_SBK41_11_rodaggio+cdp'
%addpath 'E:\SBK31_09\20171107_5lap_knockacc+psens'

load 'Rampa_WOT_02_C_SBK30_09.mat'



%%
%%%%%%%%VARIABILE DA ANALIZZARE%%%%%%%%
Fs=500 %Hz
buff= 500/1000*Fs %%% selezionare il buffer della registrazione da
analizzare
over= 376/1000*Fs %%%overlap campioni analizzati

[VAR, VARNAME] = uigetvariables('Seleziona la variabile da analizzare')
%VAR{1}=%rpm-rpm_dyn;
%VARNAME{1}=%'rpm-rpm_dyn';

lap=15337:22861;
time=Time;
%%

t_tot=time(lap)-min(time(lap));
x_tot=VAR{1}(lap);
var_an=VARNAME{1};
figure
subplot(2,1,1)
plot(t_tot,x_tot)
axis([0 max(t_tot) min(x_tot) max(x_tot)])

N_tot=length(time(lap));
N=buff;
wind=hanning(N)/mean(hanning(N));

inf=-buff+over+1;
tic
for h=1:(floor((N_tot-buff)/buff)*(1/(1-over/buff)))
```

```
inf=inf+buff-over;
sup=inf+buff-1;
range=[inf:1:sup];

t=t_tot(range);
X_tt=x_tot(range);

rpm_m(h)=mean(RPMIST(lap(range)));
X_t=detrend(X_tt,0); %%% eliminazione componente media
X=detrend(X_t,1); %%% eliminazione componente lineare
N=buff;%length(RPM_Time_Domain.Tooth_Time.data(range)/1000);

FTX=fft(X.*wind,N); tipo='ampiezza';

S=abs(FTX)/N;  tipo='ampiezza';
%S=(abs(FTX).^2)/N; tipo='potenza';

f_FT=(0:Fs/N:Fs-Fs/N);

S=S(1:N/2)*2; S(1)=0; %elimino la seconda meta' dello spettro e moltiplico
*2 esclusa la componente media
f_FT=f_FT(1:N/2); %elimino la seconda meta' dele frequenze

time_spectrum(h)=over*h/Fs; %tempo corrispondente di ogni fft
for hh=1:length(S)

    spectrum(hh,h)=S(hh);

end

%%
% plot singola fft
% figure(2)
% subplot(2,1,1)
% plot(t,X_t),hold on
% subplot(2,1,1)
% plot(t,X)
% subplot(2,1,2)
% plot(f_FT,S)
% pause
% close 2

end
toc
%%
%%%colormap
colormap('jet')
x = [0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1]*60;

Nx = length(x);
clim = [min(x) max(x)];
dx = min(diff(x));
y = clim(1):dx:clim(2);
for k=1:Nx-1, y(y>x(k) & y<=x(k+1)) = x(k+1); end % NEW
cmap = colormap(jet(Nx));
cmap2 = [...
interp1(x(:),cmap(:,1),y(:)) ...
```

```
interp1(x(:),cmap(:,2),y(:)) ...
interp1(x(:),cmap(:,3),y(:)) ...
];

subplot(2,1,2)
imagesc(time_spectrum,f_FT,spectrum)
set(gca,'Ydir','Normal')
title('Spettrogramma')
xlabel('time')
ylabel('freq (Hz)')
zlabel(tipo)
colormap(cmap2)
caxis(clim)
%axis([0 max(time_spectrum) 0 50])

figure
surf(rpm_m,f_FT,spectrum,'LineStyle','none')
title('Campbell Diagram')
xlabel('RPM')
ylabel('freq (Hz)')
zlabel(tipo)
colormap(cmap2)
caxis(clim)
colorbar
```

7.2 FILE DI OBI CAMPIONATI SU BASE ANGOLO

```
clear all
close all
clc

%%%%% analisi basata sui tempi dente registrati con OBI

%%% la cartella da cui caricare il file
addpath 'C:\Program Files\MATLAB\R2016b\extern function &
lib\uigetvariables'
addpath 'E:\SBK41_11\170721_091300_SBK41_11_rodaggio+cdp'
addpath 'E:\SBK31_09\20171107_5lap_knockacc+psens'

%load '6000-9000.mat'
load '7500-13000.mat'
%load '170721_174552_SBK41_11_02P_avviamento'
%load 'rpm+copp.mat'

%%
%%%%%%%%VARIABILE DA ANALIZZARE%%%%%%%%
Fs=22 %denti/giro (24-2)
buff= 40*Fs %%% selezionare il buffer della registrazione da analizzare
over= 1/2*buff %%%overlap campioni analizzati
%[VAR, VARNAME] = uigetvariables('Seleziona la variabile da analizzare')
VAR{1}=rpm_obi;
VARNAME{1}='rpm_obi';

time=time_obi;
lap=1:length(time); %an_elements_obi;
%%
```

```
t_tot=time(lap)-min(time(lap));
x_tot=VAR{1}(lap);
var_an=VARNAME{1};
figure(1)
%subplot(2,1,1)
plot(t_tot,x_tot)
axis([0 max(t_tot) min(x_tot) max(x_tot)])

N_tot=length(time(lap));
N=buff;
wind=hanning(N)/mean(hanning(N));

inf=-buff+over+1;
tic
for h=1:(floor((N_tot-buff)/buff)*(1/(1-over/buff)))

inf=inf+buff-over;
sup=inf+buff-1;
range=[inf:1:sup];

t=t_tot(range);
X_tt=x_tot(range);

% figure(2)
% plot(t,X_tt)

%%% sostituzione degli Inf
X_max=max(X_tt(isfinite(X_tt)));
for i=1:length(X_tt)
   if X_tt(i)==Inf
   X_tt(i)=X_max;
   end
end
%%% interpolazione cava
for i=2:length(X_tt)-1
   if X_tt(i+1)/X_tt(i)>2
       X_tt(i)=interp1([t(i-1) t(i+1)],[X_tt(i-1)
X_tt(i+1)],t(i));%(X_tt(i-1)+X_tt(i+1))/2;
   end
end

% %%%% plot rpm_ist
% % X_tt=filter(b,a,X_tt);
%  hold on
%  plot(t,X_tt)
%  pause
%  close(2)

rpm_m(h)=mean(X_tt);
X_t=detrend(X_tt,0); %%% eliminazione componente media
X=detrend(X_t,1); %%% eliminazione componente lineare
N=buff;%length(RPM_Time_Domain.Tooth_Time.data(range)/1000);
FTX=fft(X.*wind',N);

S=(abs(FTX))/N;  tipo='ampiezza';
%S=(abs(FTX).^2)/N; tipo='potenza';

f_FT=(0:Fs/N:Fs-Fs/N);
```

```
S=S(1:N/2)*2; S(1)=0; %elimino la seconda meta' delle stime emoltiplico *2
(per avere la stessa potenza/ampiezza sullo spettro a singolo lato)
f_FT=f_FT(1:N/2); %elimino la seconda meta' degliordini
freq(:,h)=f_FT*rpm_m(h)/60; %conversione a frequenza

time_spectrum(h)=t_tot(inf); %tempo corrispondente di ogni fft
for hh=1:length(S)

    spectrum(hh,h)=S(hh); %%ordini

end

[rpm_sort, kk]=sort(rpm_m);
spectrum_sort=spectrum(:,kk);
freq_sort=freq(:,kk);
%% plot singola fft
% figure(3)
%  subplot(2,1,1)
% % plot(t,X_t),hold on
% % subplot(2,1,1)
% plot(t,X)
% subplot(2,1,2)
% plot(f_FT,S)
% pause
% close(3)

end
toc

%%
Ris_ord=Fs/N; disp(['risoluzione spettro degli ordini = '
num2str(Ris_ord)]);

% RPM_sort=zeros(size(freq_sort));
% for p=1:length(rpm_sort)
% RPM_sort(p,:)=rpm_sort(p);
% end

%%
%%%colormap
colormap('jet')
%x = [0 2 5 7.5 15 30 60 125 250 500]./5;
x = [0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1].*300;
Nx = length(x);
clim = [min(x) max(x)];
dx = min(diff(x));
y = clim(1):dx:clim(2);
for k=1:Nx-1, y(y>x(k) & y<=x(k+1)) = x(k+1); end % NEW
cmap = colormap(jet(Nx));
cmap2 = [...
interp1(x(:),cmap(:,1),y(:)) ...
interp1(x(:),cmap(:,2),y(:)) ...
interp1(x(:),cmap(:,3),y(:)) ...
];

% subplot(2,1,2)
% imagesc(time_spectrum,f_FT,spectrum)
% set(gca,'Ydir','Normal')
```

```
% title('Spettrogramma')
% xlabel('time')
% ylabel('freq (Hz)')
% zlabel(tipo)
% colormap(cmap2)
% caxis(clim)

figure
surf(rpm_m,freq,spectrum,'LineStyle','none')
title('Campbell diagram 7500-13000 rpm')
xlabel('RPM')
ylabel('freq (Hz)')
zlabel(tipo)
colormap(cmap2)
caxis(clim)
colorbar
axis([7500 max(rpm_m) 0 500])

order_idx=[21 41 61 81 101 121]; %%% selezione ordini 1/2 1 3/2 2 5/2 3
figure
plot(rpm_m,spectrum(order_idx,:))
axis([7500 max(rpm_m) 0 300])
title('rampa 7500-13000 rpm')
xlabel('RPM')
ylabel('ampiezza (RPM)')
legend('ord 1/2', 'ord 1', 'ord 3/2', 'ord 2', 'ord 5/2', 'ord 3')
```

BIBLIOGRAFIA

[1] G. C. R. L.Barelli, «Diagnosis of internal combustion engine through vibration and acoustic pressure non - intrusive measurements,» *Applied Thermal Engineering,* 2008.

[2] P. ,. M. M.Pinelli, «Gas Turbine Health State Determination: Methodology Approach and Field Application,» *International Journal of Rotating Machinery,* 2012.

[3] S. d. C. Ribeiro, *Implementation of an Engine Condition Monitoring tool for Airbus Aircraft,* Instituto Superior Tecnico, Lisbona, Portugal, May 2015.

[4] H. Tienhaara, «Guidelines to engine dynamics and vibration,» *Wärtsilä Corporation,* 2004.

[5] M. G., *Metodi diagnostici per ingranaggi ad elevate prestazioni con misure di vibrazione ed acustiche- Confronto accelerometri- microfono,* Università di Pisa, Tesi di laurea vecchio ordinamento, October 2004.

[6] BS ISO, *20816-1:2016, Mechanical vibration —Measurement and evaluation of machine vibration, part 1: General guidelines.*

[7] H. a. H. H. Konstantin-Hansen, «Envelope and cepstrum analyses for machinery fault identification,» *Sound and Vibration 44.5,* 2010.

[8] G. O. A. S.Delvecchio, «Condition monitoring in diesel engines for cold test apllications. Part1: vibration analysis for pass/fail decision.,» in *Proceedings of COMADEM 2007, International Congress on Condition Monitoring and Diagnostics Engineering Management*, Portugal, 13-15 June 2007.

[9] V. P. P. V.Macian, «Analytical approach to wear rate determination for internal combustion engine condition monitoring based on oil analysis,» *Tribology International,* 36.10 (2003): 771-776.

[10] R.Murukesan, «Lube Oil Condition Monitoring System: - An Alternative Methodology,» *SAE Technical Paper,* n. 2008-01-2757, 2008.

[11] H. C. &. P. A.G., «Harris' Shock And Vibration Handbook 5th Ed,» McGraw Hill, 2002, p. capitolo 16.

[12] J. B. ALENA BILOŠOVÁ, *Vibration Diagnostics,* Ostrava, 2012.

[13] Reilhofer KG, «delta-Analyzer v2,» [Online]. Available: httpps://www.rhf.de.

[14] Wintec, *Metodi, Tecnologie e Strumenti per l'Analisi Vibroacustica di Macchine Rotanti.*

[15] S. S., *Emchancing Automotive Embedded System with FPGAs,* Nanyang Technological University, April 2016.

[16] Alma Automotive, «Miracle2 - Micro Rapid Controller & Logging Enviroment,» [Online]. Available: http://www.alma-automotive.it.

[17] Alma Automotive, «OBI-M2 Compact and Rugged Combustion Analysis System for use on Vehicles, Motorbikes and Test Benches,» [Online]. Available: http://www.alma-automotive.it.

RINGRAZIAMENTI

A conclusione di questo lavoro mi sembra doveroso lasciare per iscritto un ringraziamento al professor Enrico Corti per le opportunità offertemi e per la fiducia mostratami nello svolgimento delle attività, durante le quali ha ricoperto un fondamentale ruolo di riferimento e supervisione, pur concedendomi un'ampia ed apprezzata autonomia.

In egual misura mi sento di ringraziare Paolo e Manuel, che seguendomi regolarmente nello sviluppo del software hanno reso possibile questo elaborato. Senza ciò che mi hanno insegnato e senza le loro indicazioni questo lavoro non sarebbe stato possibile ed in particolare vorrei ringraziarli per l'accoglienza che mi hanno riservato.

Inoltre vorrei ringraziare tutti i ragazzi di Alma Automotive, a partire da Alessandro e Lino che mi hanno accolto in sala prove, che in un modo o nell'altro mi hanno incontrato, aiutato o che semplicemente hanno scambiato due parole con me durante le giornate lavorative. Infine un pensiero ed un ringraziamento per l'aiuto va ai dottorandi Filippo e Michele ed anche a Stefano e Giacomo con cui ho condiviso molte giornate in hangar.

Per famigliari e amici i ringraziamenti sono scontati e verranno fatti nei luoghi e nei modi opportuni, ma al termine di questo elaborato trovo giusto ricordare per iscritto soprattutto chi ha reso possibile attivamente la sua realizzazione, anche se il formalismo di un ringraziamento su carta non rende mai sufficiente giustizia.