POLITECNICO DI TORINO

Corso di Laurea Magistrale in

Ingegneria Civile

Tesi di Laurea Magistrale

Monitoraggio strutturale di un viadotto in calcestruzzo armato tramite inclinometri:
analisi dei dati con un algoritmo scritto in Python

Relatore

Prof. Gabriele Bertagnoli

Candidato

Cecilia Antognelli

Anno Accademico 2019/2020

Inevitabilmente,
ai miei genitori.

Indice

1. INTRODUZIONE 9

1.1. Viadotto in esame 9

1.2. Strumentazione 10

1.2.1. Sensore MEMS 12

2. ACQUISIZIONE DEI DATI 15

2.1. Prova di carico 15

2.1.1. Carichi di prova 15

2.1.2. Punti di lettura e strumenti di misurazione 17

2.1.3. Modalità di prova 17

2.1.4. Risultati 18

2.2. Allineamento dei dati antecedenti e successivi alla prova di carico 23

2.3. Creazione di un valore di zero a ponte scarico 24

3. ANALISI DEI DATI 27

3.1. Analisi dei dati grezzi 27

3.2. Coefficienti di correlazione dei dati grezzi 31

3.2.1. Coefficiente di correlazione 31

3.2.2. Correlazione tra rotazioni raw e temperature 32

3.3. Calcolo della deriva termica istantanea 35

3.4. Analisi dei dati corretti 37

3.5. Coefficienti di correlazione dei dati corretti 38

3.6. Calcolo della deriva termica differita 43

3.7. Analisi dei dati doppiamente corretti 45

3.8. Coefficienti di correlazione per dati doppiamente corretti 46

3.9. Coefficienti di correlazione tra sensori 50

3.10. Analisi nel dominio della frequenza 57

3.10.1. Trasformata di Fourier 57

3.10.2. Densità Spettrale di Energia (ESD) 61

3.11. Grafici ottenuti 62

4. ALGORITMO IN PYTHON 69

4.1. Introduzione 69

4.2. Moduli aggiuntivi 69

4.3. Moduli aggiuntivi personalizzati 71

4.4. File di input e parametri scelti dall'utente 73

4.5. File di output generati dal programma 79

4.6. Operazioni preliminari 83

4.6.1. Blocco (1): apertura file di input e acquisizione dei parametri scelti dall'utente 83

4.7. Prima parte: pulizia da anomalie (plateaux e picchi) 91

4.7.1. Blocco (2): Acquisizione dei dati 91

4.7.2. Blocco (3): Operazioni di pulizia dei dati raw dai plateaux 91

4.7.3. Blocco (4): Operazioni di pulizia dei dati raw dai picchi 92

4.7.4. Blocco (5): Creazione delle directory di salvataggio 92

4.7.5. Blocco (6): Grafici 92

4.7.6. Blocco (7): Creazione file di output con dati puliti 92

4.8. Seconda parte: analisi dei dati e svincolamento da temperatura (o umidità) 93

4.8.1. Blocco (8): Definizione dei data-set e acquisizione dei parametri scelti dall'utente 93

4.8.2. Blocco (9): Calcoli sulle rotazioni raw 104

4.8.3. Blocco (10): Calcoli su temperatura e umidità 107

4.8.4. Blocco (11): Correlazioni tra le rotazioni raw e la temperatura 112

4.8.5. Blocco (12): Calcolo della deriva istantanea 114

4.8.6. Blocco (13): Calcoli sulle rotazioni corrette dalla deriva istantanea 117

4.8.7. Blocco (14): Correlazioni tra le rotazioni corrette e la temperatura 118

4.8.8. Blocco (15): Calcolo della deriva differita 120

4.8.9. Blocco (16): Calcoli sulle rotazioni corrette dalla deriva differita 124

4.8.10. Blocco (17): Correlazioni tra le rotazioni doppiamente corrette e la temperatura 125

4.8.11. Blocco (18): Grafici 126

4.8.12. Blocco (19): Analisi nel dominio della frequenza 133

4.8.13. Blocco (20): Salvataggio del file Excel dei risultati 138

4.8.14. Blocco (21): Salvataggio dati per la cross-correlazione tra sensori 139

4.8.15. Blocco (22): Creazione di vettori privi di nan 151

4.8.16. Blocco (23): Grafico coefficienti correlazione – ritardi 151

4.8.17. Blocco (24): Cross-correlazione tra sensori 154

4.9. Cenni su sensori di tipo A 170

5. CONCLUSIONI 171

Indice delle Figure 175

Indice delle Tabelle 179

Ringraziamenti 181

1. INTRODUZIONE

L'impianto infrastrutturale italiano, ed in particolar modo la rete autostradale, sono stati costruiti per gran parte nel XX secolo. Negli ultimi anni, fomentato da alcuni tragici avvenimenti, si è diffuso il comune sentimento che la rete infrastrutturale italiana non sia adeguata alle mutate esigenze di traffico e logistica, oltre che indebolita dal tempo e dallo stress causato da agenti esterni; questo fa sì che sia considerata inaffidabile e traballante in termini di sicurezza. Questa condizione rende chiara la necessità, ora più che mai, di interventi di monitoraggio e di manutenzione precisi e sistematici.

L'utilizzo di un sistema di monitoraggio strutturale consente di analizzare le prestazioni strutturali di una costruzione, con continuità nel tempo. Da questo deriva una migliore conoscenza del comportamento reale della struttura, evidenziando le eventuali carenze in termini di resistenza e la presenza di danni.

Lo scopo finale di queste operazioni è una valutazione della sicurezza complessiva della struttura; citando le Norme Tecniche per le Costruzioni (NTC) 2018, "la valutazione della sicurezza, argomentata con apposita relazione, deve permettere di stabilire se:

- l'uso della costruzione possa continuare senza interventi;
- l'uso debba essere modificato (declassamento, cambio di destinazione e/o imposizione di limitazioni e/o cautele nell'uso);
- sia necessario aumentare la sicurezza strutturale, mediante interventi."[1]

1.1. Viadotto in esame

Sono stati analizzati i dati provenienti dai clinometri posizionati sulla ventitreesima campata di un viadotto autostradale.

L'impalcato è costituito da sei travi; in particolare sono stati installati due sensori per ciascuna delle quattro travi centrali, uno sul lato nord della campata e uno sul lato sud.

In *Figura 1* si mostra la disposizione dei sensori usati; i clinometri sono rappresentati in blu, mentre gli estensimetri (i cui dati non vengono analizzati in questo documento) sono rappresentati in rosso. Ciascun inclinometro viene indicato con un codice del tipo EL-C23-T2-N, in cui il primo gruppo di lettere specifica il tipo di strumento (i clinometri sono

[1] Decreto ministeriale 17 gennaio 2018, "Aggiornamento delle «Norme tecniche per le costruzioni»" (pubblicato nel Supplemento Ordinario n. 8 alla Gazzetta Ufficiale n. 42 del 20 febbraio 2018)

indicati con EL, mentre il codice identificativo degli estensimetri inizia con SG), il secondo e il terzo gruppo determinano rispettivamente il numero della campata e il numero della trave, mentre il quarto e ultimo gruppo specifica il lato della campata su cui è posizionato (nord o sud).

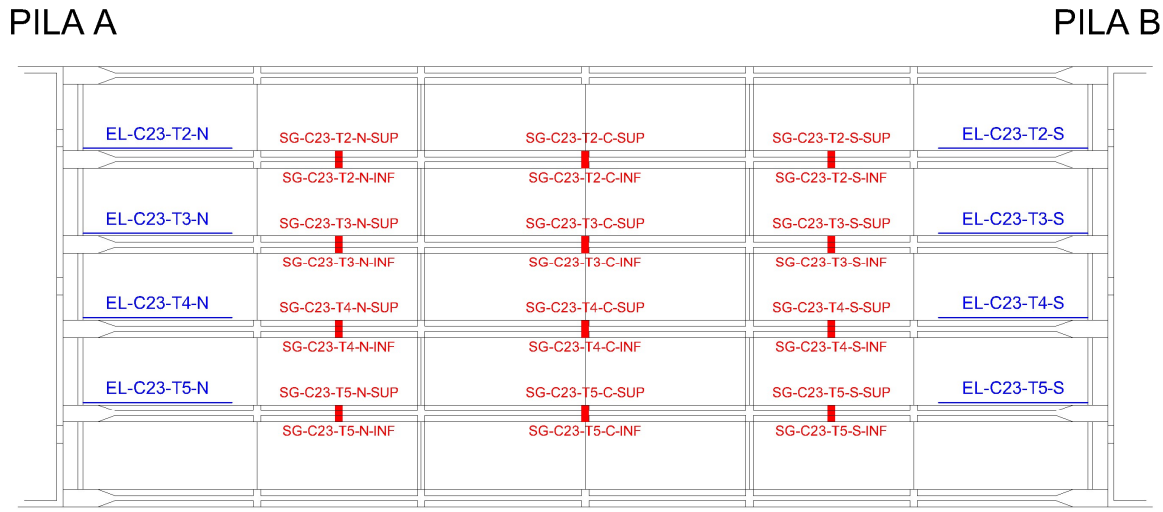


Figura 1: Disposizione dei sensori

1.2. Strumentazione

Il clinometro o inclinometro (più raramente, tiltmetro) è lo strumento utilizzato per la misura di angoli di inclinazione (in inglese, *tilt*), di pendenza, di elevazione o di depressione rispetto all'orizzontale.

Con questo termine, in genere, si intende il sistema composto da un tubo e da una sonda inclinometrica, utilizzato per misurare gli spostamenti profondi orizzontali del suolo, ma ne esistono altre tipologie, come ad esempio gli inclinometri da parete.

I sensori installati per il monitoraggio del viadotto in esame sono, per la precisione, inclinometri a barra.
Questi possono essere utilizzati singolarmente per monitorare l'inclinazione relativa tra due punti (come nel caso in esame), oppure possono essere installati in sequenza per controllare, ad esempio, i cedimenti differenziali di una struttura.

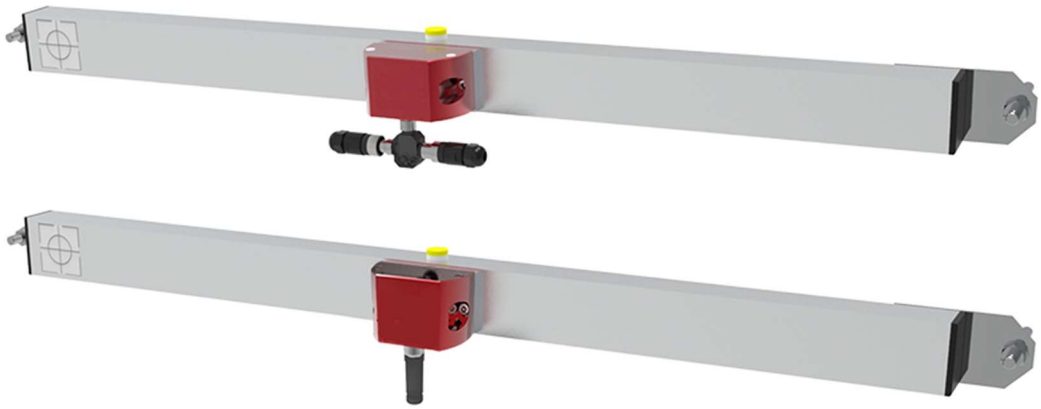


Figura 2: Inclinometri a barra

Sistemi inclinometrici a barra (chiamati "*Tilt Beam sensors*") sono costituiti da un inclinometro MEMS (Micro Electro-Mechanical Systems) posizionato su una barra di alluminio avente una ben definita *"gauge length"*, indicata come L in *Figura 3*, in generale pari a 1, 2 o 3 metri, che rappresenta la distanza tra i due punti di riferimento tra cui si vuole misurare l'inclinazione.

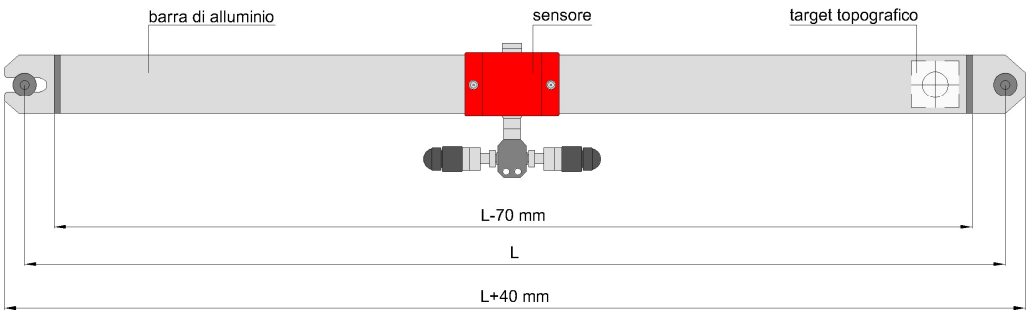


Figura 3: Elementi e dimensioni dell'inclinometro a barra

Essi possono essere analogici o digitali, e il sensore può essere monoassiale o biassiale, cioè può misurare angoli di inclinazione in una o due direzioni perpendicolari tra loro.

1.2.1. Sensore MEMS

L'elemento centrale del sensore MEMS è un accelerometro, il quale misura la forza di reazione vincolare con cui un supporto fisso agisce su una massa di prova per mantenerla legata a sé nel movimento. Nella sua configurazione più elementare esso è costituito da una massa di prova collegata alla struttura di supporto tramite una molla (o un qualche elemento elastico). Quando l'oggetto su cui è montato l'accelerometro è sottoposto a una variazione di velocità, la massa, avente propria inerzia, tende a conservare il proprio stato, e perciò la molla che la collega al supporto viene deformata, ed imprime sulla massa sospesa una forza tale che anche la massa abbia stessa accelerazione del supporto; la deformazione della molla, ovvero lo spostamento relativo tra massa di prova e supporto fisso, è proporzionale alla forza elastica esercitata, la quale a sua volta è proporzionale all'accelerazione subita. Dunque, nota la distanza tra il supporto e la massa di prova, è nota l'accelerazione che si vuole valutare.

I sensori usati in questo caso sono accelerometri capacitivi, i quali sfruttano come principio per valutare lo spostamento della massa, la variazione della capacità elettrica di un condensatore al variare della distanza tra le sue armature; una struttura solidale al dispositivo e la massa di prova costituiscono queste armature. Al variare della distanza, la capacità elettrica varia anch'essa; misurando quest'ultima, è noto lo spostamento.

Il sensore converte poi l'accelerazione del corpo in un segnale elettrico.

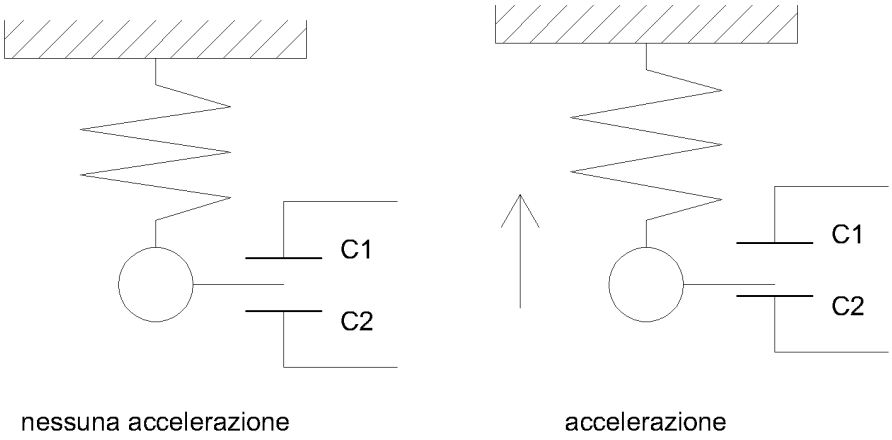


Figura 4: Funzionamento di un accelerometro capacitivo

Misure accelerometriche vengono sfruttate, poi, per valutare angoli di inclinazione. Immaginiamo un accelerometro mono-assiale, in grado, cioè, di misurare l'accelerazione lungo un solo asse, e supponiamo che sia posto inizialmente sul piano dell'orizzonte (piano perpendicolare alla direzione del vettore che rappresenta la forza di gravità), ovvero immaginiamo che sia posto su un piano soggetto al campo di gravità. In questo stato, l'accelerometro misura un'accelerazione pari a 1g sull'unico asse di misura a disposizione. Ipotizziamo ora di ruotare l'accelerometro di un determinato angolo α rispetto al piano dell'orizzonte.

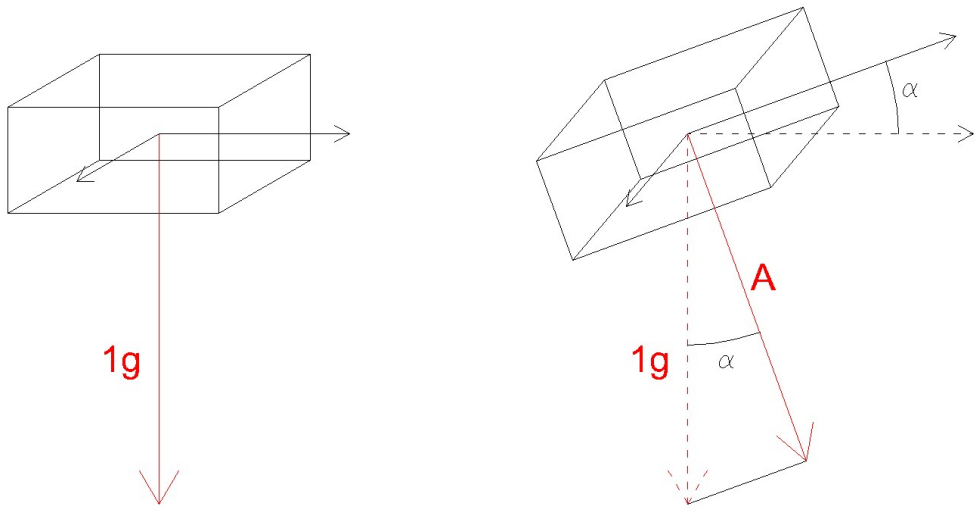


Figura 5: Uso di un accelerometro per valutare inclinazioni

L'accelerazione che ora viene misurata dall'accelerometro è solo la componente dell'accelerazione di gravità parallela all'asse su cui il sensore effettua misurazioni, nella *Figura 5* pari ad A; considerando le regole della trigonometria, si può affermare che:

$$A = 1g \cdot \sin(\alpha)$$

Invertendo tale formula, si ottiene facilmente il valore dell'angolo α:

$$\alpha = \arcsin\left(\frac{A}{1g}\right)$$

Infine, questi sensori MEMS sono dotati di un sensore di temperatura a termistore per monitorare le variazioni termiche durante le letture.

2. ACQUISIZIONE DEI DATI

L'acquisizione dei dati è iniziata il 01/01/2019, e sono disponibili misurazioni fino al 04/06/2019.

La frequenza di campionamento è stata variabile nel tempo; nello specifico, l'intervallo di tempo compreso tra un'acquisizione e la successiva, per diverse date, viene riportato di seguito:

- Dal 01/01/2019 al 18/01/2019 5 minuti;
- Dal 18/01/2019 al 31/01/2019 15 minuti;
- Dal 31/01/2019 al 01/02/2019 60 minuti;
- Dal 01/01/2019 al 18/02/2019 30 minuti;
- Dal 18/02/2019 al 04/06/2019 20 minuti.

Si considerano quindi validi per l'analisi che si intende svolgere solo i dati relativi al periodo con frequenza di campionamento costante, pari a 20 minuti, ovvero l'intervallo di tempo compreso tra il 18/02 e il 04/06.

2.1. Prova di carico

Nella notte tra il 13 e il 14 marzo 2019 è stata eseguita una prova di carico sull'impalcato, che ha consentito di tarare la strumentazione.

2.1.1. Carichi di prova

Sono state eseguite due prove di carico sulla campata in esame.

Per la prima prova sono stati usati carichi modesti applicati con 8 autovetture di peso pari a 1.6 ton ciascuna. Per la seconda prova si è sollecitato l'impalcato con carichi pesanti, costituiti da 4 automezzi di peso pari a 34 ton ciascuno. La disposizione degli automezzi relativa a ciascuna prova di carico è mostrata di seguito. In entrambi i casi l'eccentricità del carico (ovvero l'asse dei veicoli) rispetto al centro dell'impalcato è pari a 6.80m.

Prima di iniziare le prove gli automezzi sono stati numerati e pesati in modo tale da poter risalire, a posteriori, all'effettivo carico applicato sull'impalcato e alla sua reale disposizione.

La prova di carico con i mezzi pesanti ha lo scopo di riprodurre le sollecitazioni di servizio "Transitabilità con restrizioni" e di tarare il sistema di monitoraggio installato sul viadotto dal DT7. La prova con mezzi leggeri ha lo scopo di validare le soglie di Allerta ed Allarme previsto nel progetto del monitoraggio.

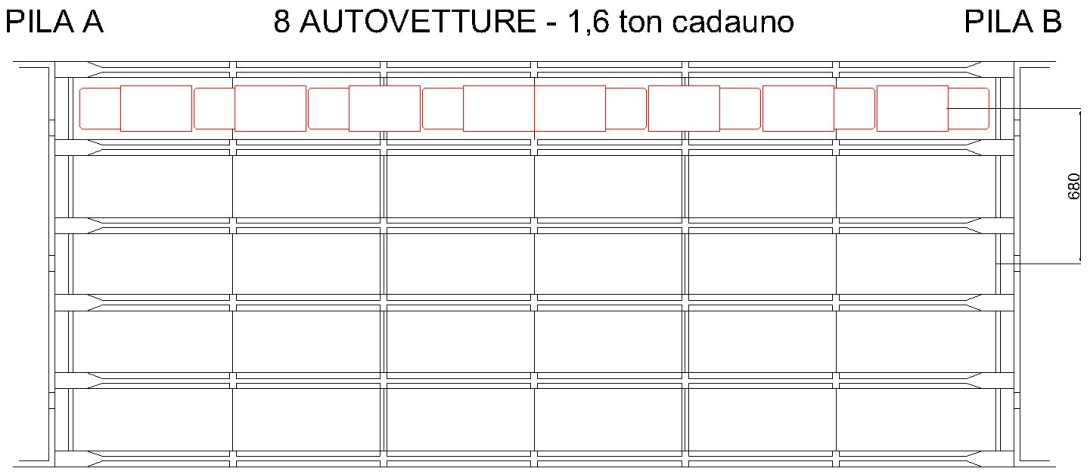


Figura 6: Disposizione dei mezzi per prova di carico con mezzi leggeri

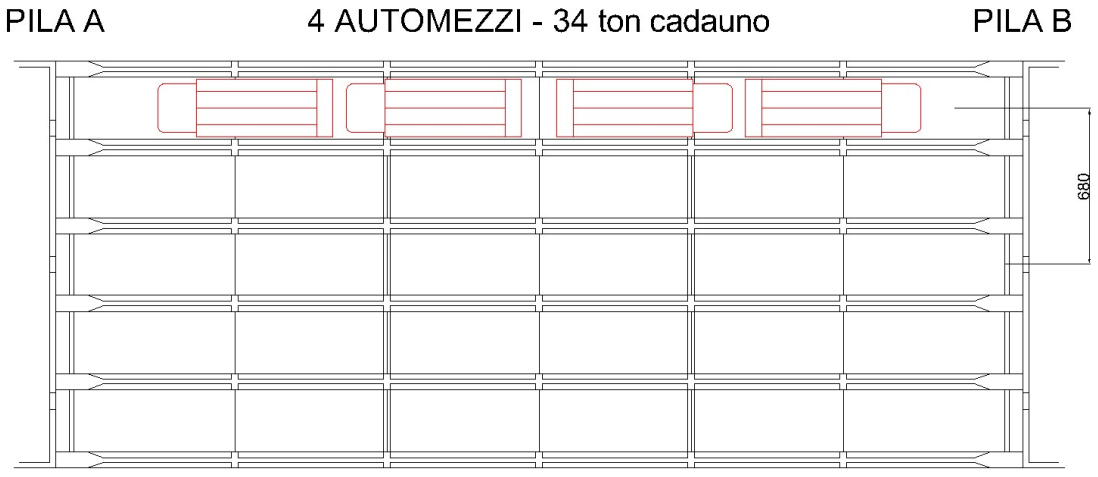


Figura 7: Disposizione dei mezzi per prova di carico con mezzi pesanti

2.1.2. Punti di lettura e strumenti di misurazione

Sono stati monitorati gli spostamenti verticali in undici punti di controllo mediante livellazione topografica di precisione dall'estradosso dell'impalcato. I punti di lettura sono disposti in due file di cinque punti ciascuna lungo i due bordi dell'impalcato, cui si aggiunge un punto isolato approssimativamente al centro dell'impalcato medesimo. Dei cinque punti di lettura di ogni fila, due sono posti in corrispondenza degli appoggi (x=0, x=L), due vengono posizionati in corrispondenza dei quarti della campata (x=L/4, x=3L/4), ed uno in mezzeria (x=L/2). Vengono inoltre eseguite le letture dei sensori installati per il monitoraggio dell'impalcato nelle fasi specifiche di carico da associare alle frecce lette topograficamente.

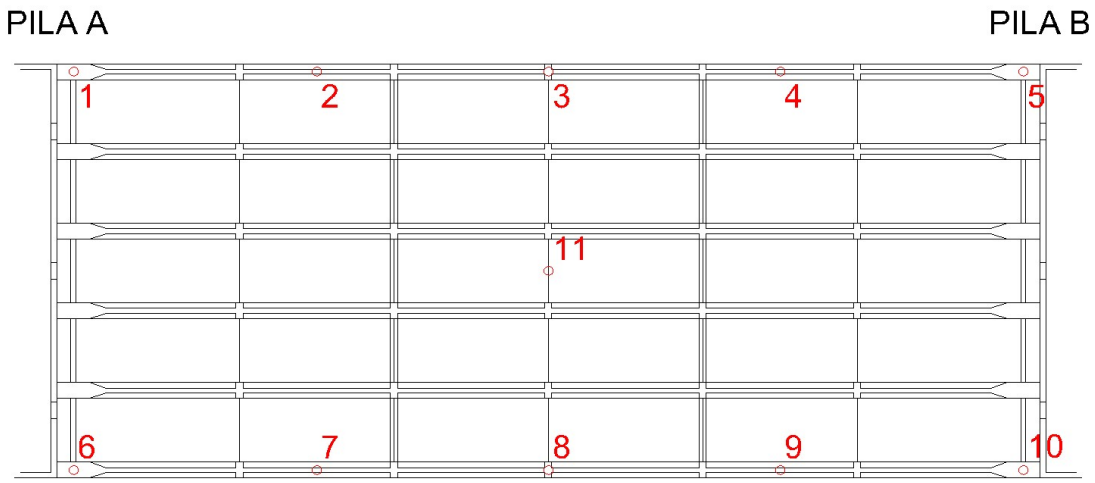


Figura 8: Disposizione dei capisaldi per la lettura degli spostamenti verticali

2.1.3. Modalità di prova

La prova di carico per mezzi leggeri è articolata in due diverse fasi di carico (fase 1 e fase 2), il cui scopo è quello di generare i massimi effetti sull'impalcato ed in particolare sulle travi di bordo, considerando che il carico viene applicato con la massima eccentricità possibile.

Sono usate complessivamente otto autovetture da 1.6 ton ciascuna; la fase 1 prevede l'introduzione di quattro di queste, mentre per la fase 2 vengono posizionate tutte.

Nella prima fase di carico i quattro automezzi sono allineati in colonna, disposti sulla corsia più esterna in corrispondenza della mezzeria dell'impalcato con posto di guida in posizioni opposte (spalla a spalla). Nella fase 2 altre quattro autovetture vengono allineate sulla stessa colonna di bordo, accodandosi ai precedenti due mezzi col medesimo verso rispetto a quello che lo precede.

Terminata la prima fase di carico, si effettuano le letture su ognuno degli undici punti di stazione, a distanza di 10 minuti l'una dall'altra, per un totale di 30 minuti.
Al termine della seconda fase di carico, le letture sono ripetute in maniera analoga.

Il processo di scarico dell'impalcato avviene analogamente alla fase di carico. Al termine di questo vengono effettuate letture per 60 minuti.

La prova con mezzi pesanti si svolge secondo le medesime modalità e procedure previste per la prova con mezzi leggeri. L'unica variante è costituita dal fatto che nella fase 1 i veicoli pesanti sono due (1+1 spalla a spalla) e nella fase 2 sono quattro (2+2).

2.1.4. Risultati

Si ottengono i valori di spostamento verticale corrispondenti ai punti di lettura sopra indicati, gli allungamenti misurati dagli estensimetri e le rotazioni misurate dagli inclinometri, oltre alla temperatura. Per facilitarne la lettura, si considerano le differenze di lettura rispetto a quella relativa al ponte scarico.
Si riportano di seguito i grafici ottenuti.

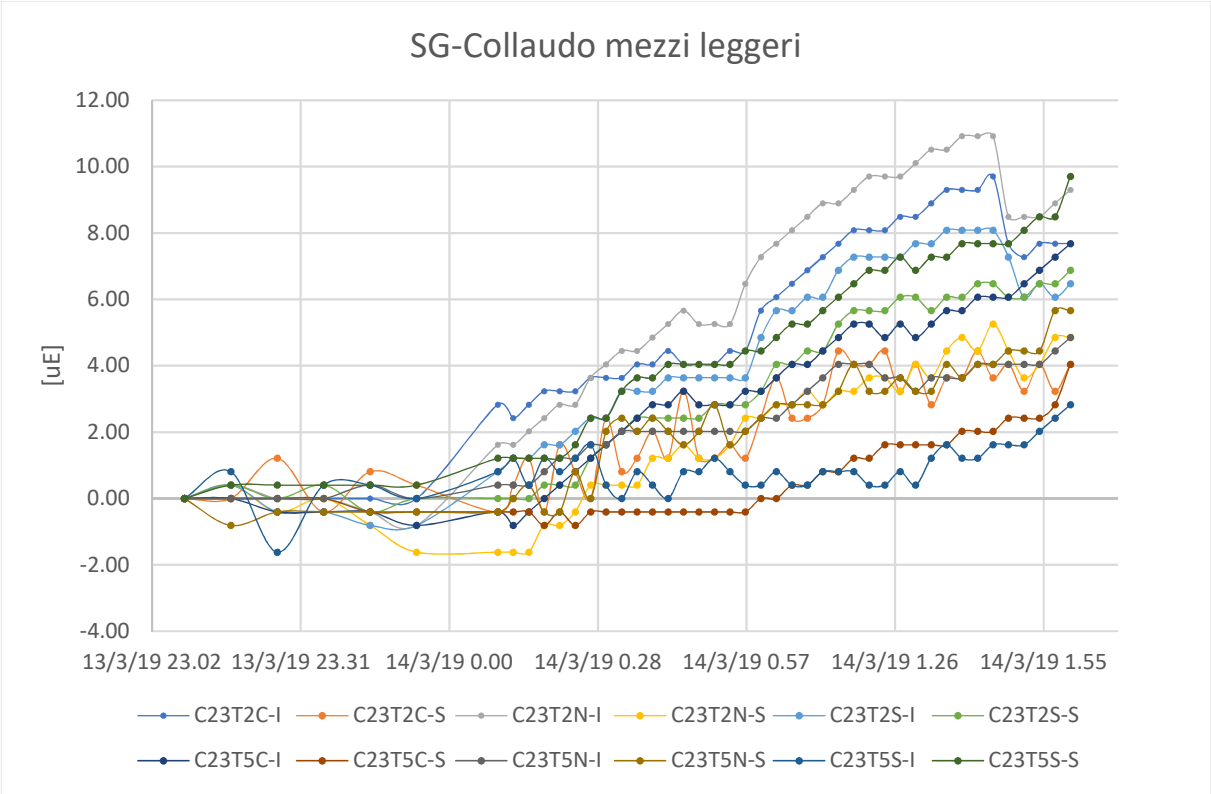


Figura 9: Variazioni di estensione dovute alla fase 1 – mezzi leggeri (με)

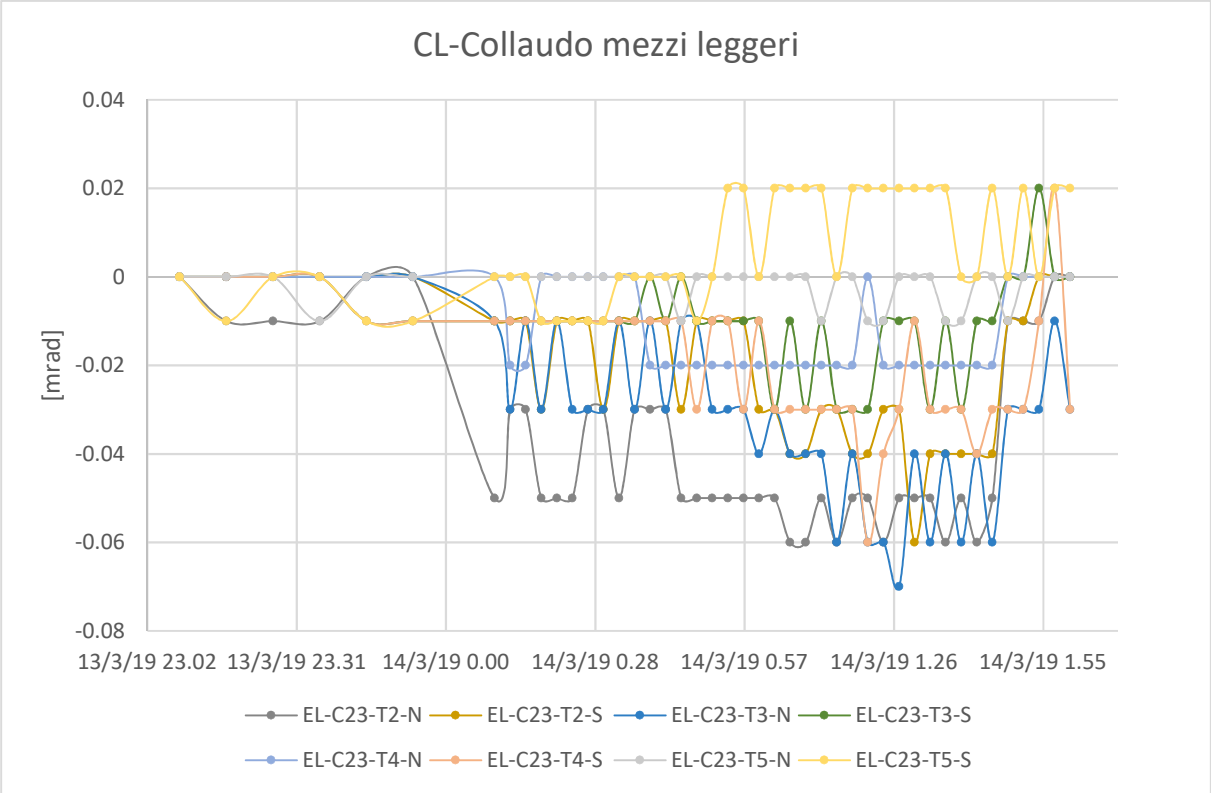


Figura 10: Variazioni di rotazione dovute alla fase 1 – mezzi leggeri (mrad)

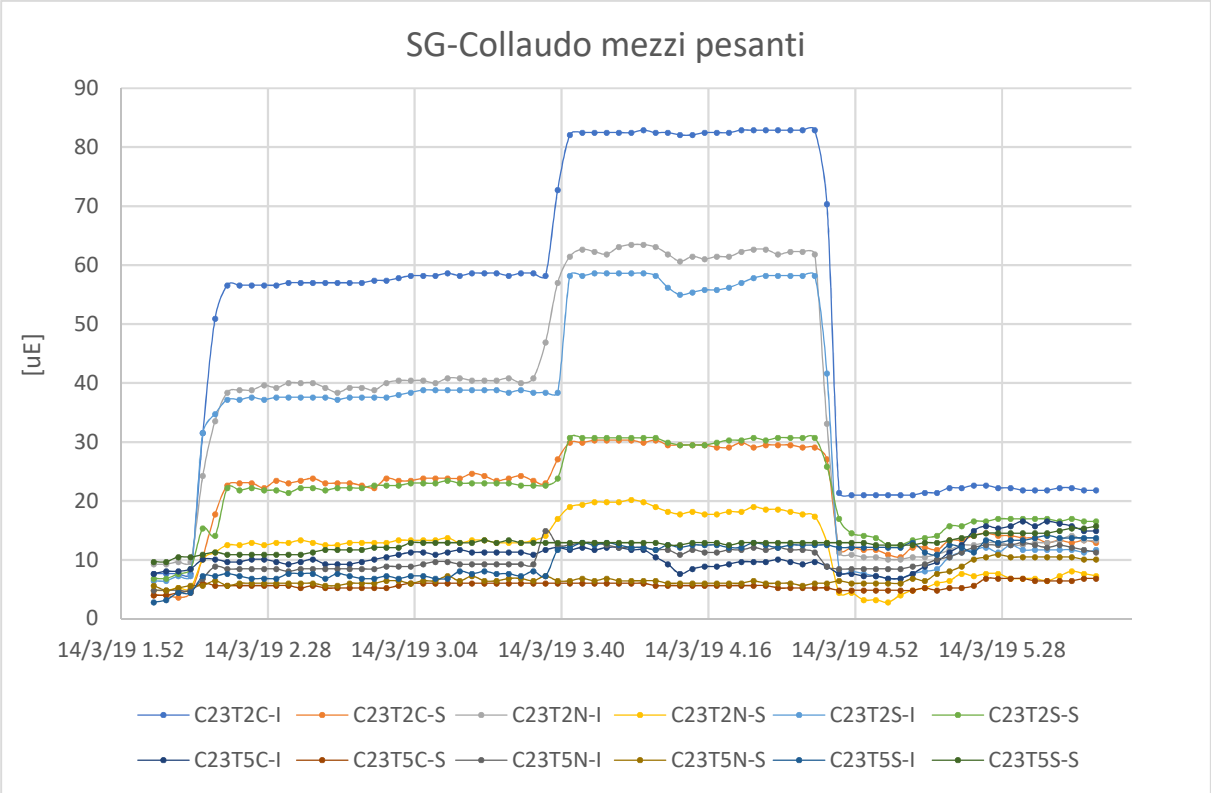


Figura 11: Variazioni di estensione dovute alla fase 2 – mezzi pesanti (με)

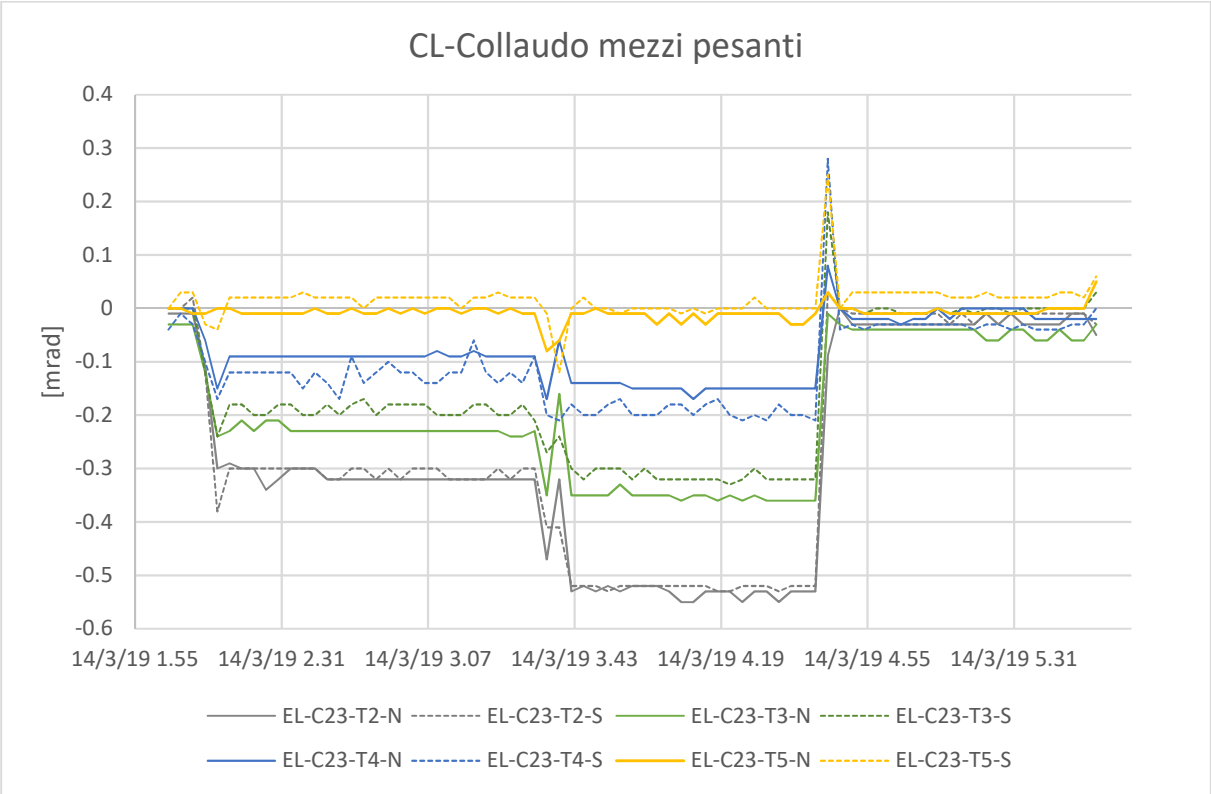


Figura 12: Variazioni di rotazione dovute alla fase 2 – mezzi pesanti (mrad)

Infine, si mostrano due grafici in cui vengono esplicitate le fasi della prova di carico.

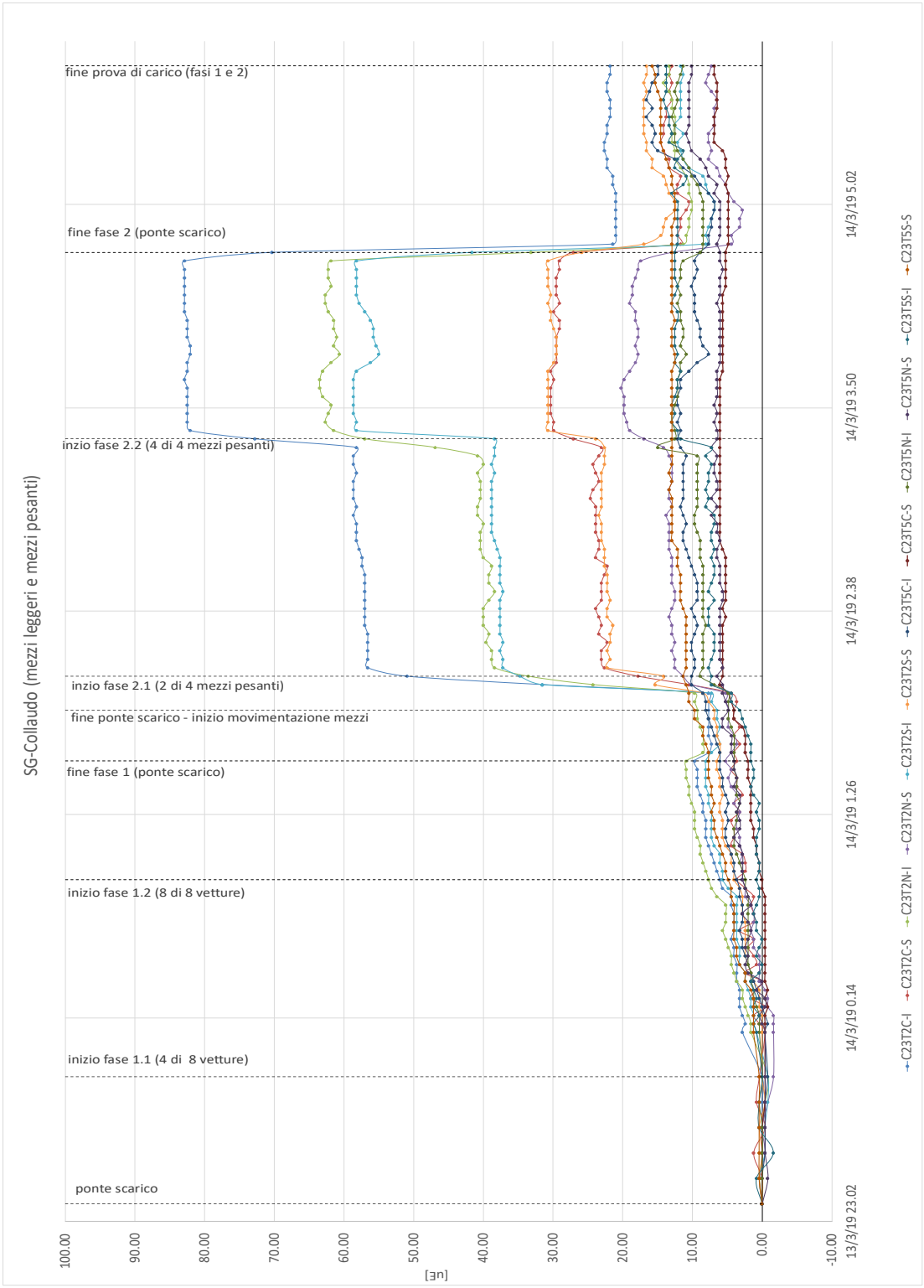


Figura 13: Variazioni di estensione dovute alle fasi 1 e 2 – mezzi leggeri e pesanti (με)

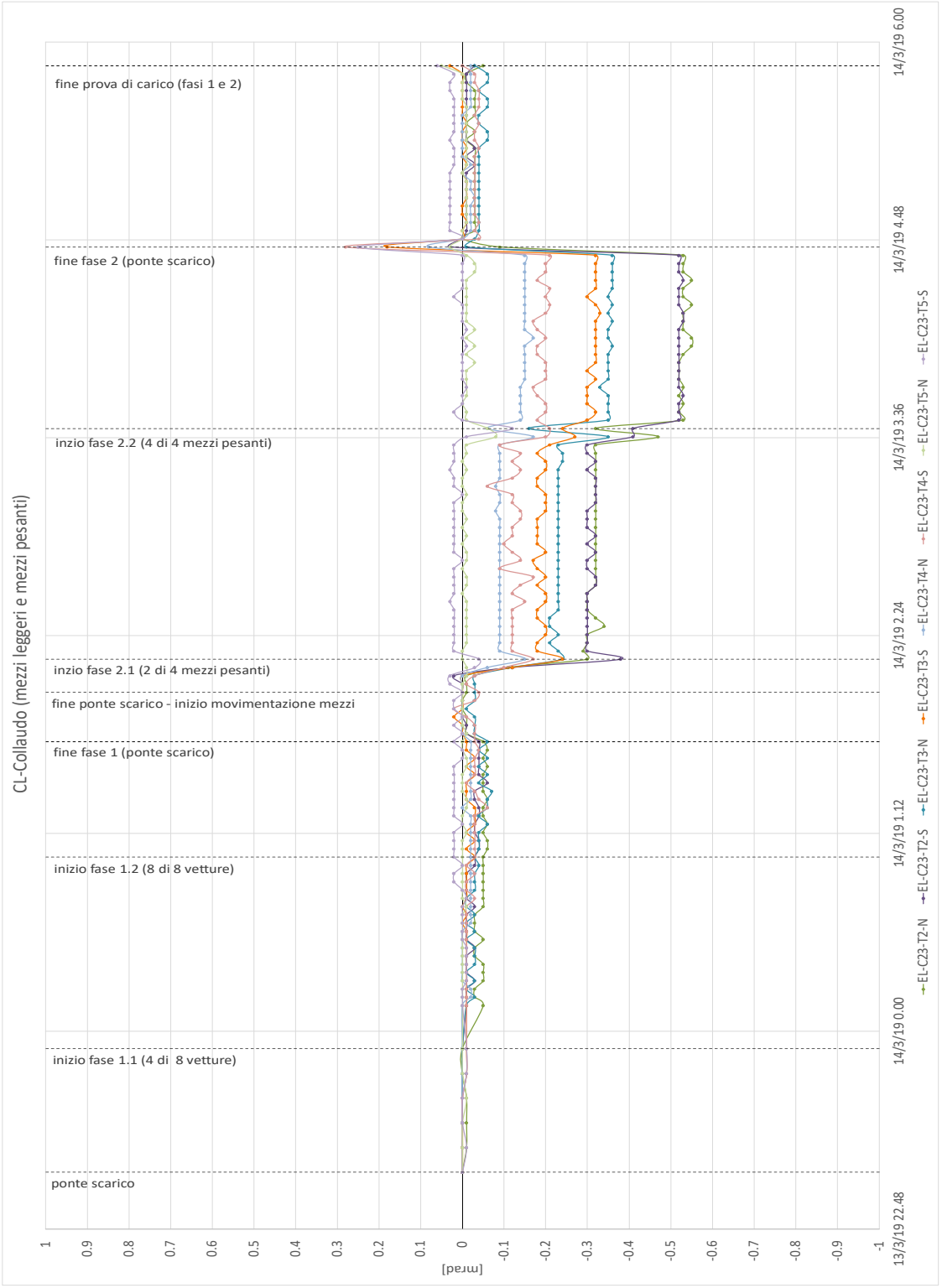


Figura 14: Variazioni di rotazioni dovute alle fasi 1 e 2 – mezzi leggeri e pesanti (mrad)

2.2. Allineamento dei dati antecedenti e successivi alla prova di carico

Il giorno 14/03 è stata operata una correzione dei dati tramite traslazione rispetto ad un nuovo valore di zero delle rotazioni ottenuto durante le prove di collaudo a ponte scarico. Tutti i dati acquisiti in precedenza (dal 18/02 al 13/03) sono stati quindi allineati sul nuovo riferimento secondo la formula seguente:

$$\theta_{new}(t) = \theta_{old}(t) + (\theta_{new,14/03\ 11.58} - \theta_{old,14/03\ 11.58})$$

Dove:

- $\theta_{new}(t)$ è la rotazione misurata dopo l'allineamento sul nuovo riferimento;
- $\theta_{old}(t)$ è la rotazione misurata prima dell'allineamento sul nuovo riferimento;
- $\theta_{new,14/03\ 11.58}$ è la rotazione misurata dopo l'allineamento sul nuovo riferimento, il 14/03 alle 11.58;
- $\theta_{old,14/03\ 11.58}$ è la rotazione misurata prima dell'allineamento sul nuovo riferimento, il 14/03 alle 11.58.

Si riporta, come esempio, il grafico che rappresenta la serie di rotazioni 'new' e quella di rotazioni 'old' relative al sensore installato sulla trave numero 2, nel lato nord della campata (EL-C23-T2-N). Si può notare chiaramente come la serie di valori 'old', raffigurata in grigio, sia stata matematicamente manipolata tramite l'operazione descritta sopra per ottenere la serie di rotazioni 'new', rappresentata con il colore giallo, allineata sul nuovo riferimento. Per misurazioni relative a istanti successivi, invece, l'allineamento sul nuovo riferimento è già insito nella misura, per cui la serie di rotazioni 'new' è esattamente quella misurata, e viene presentata in arancione.

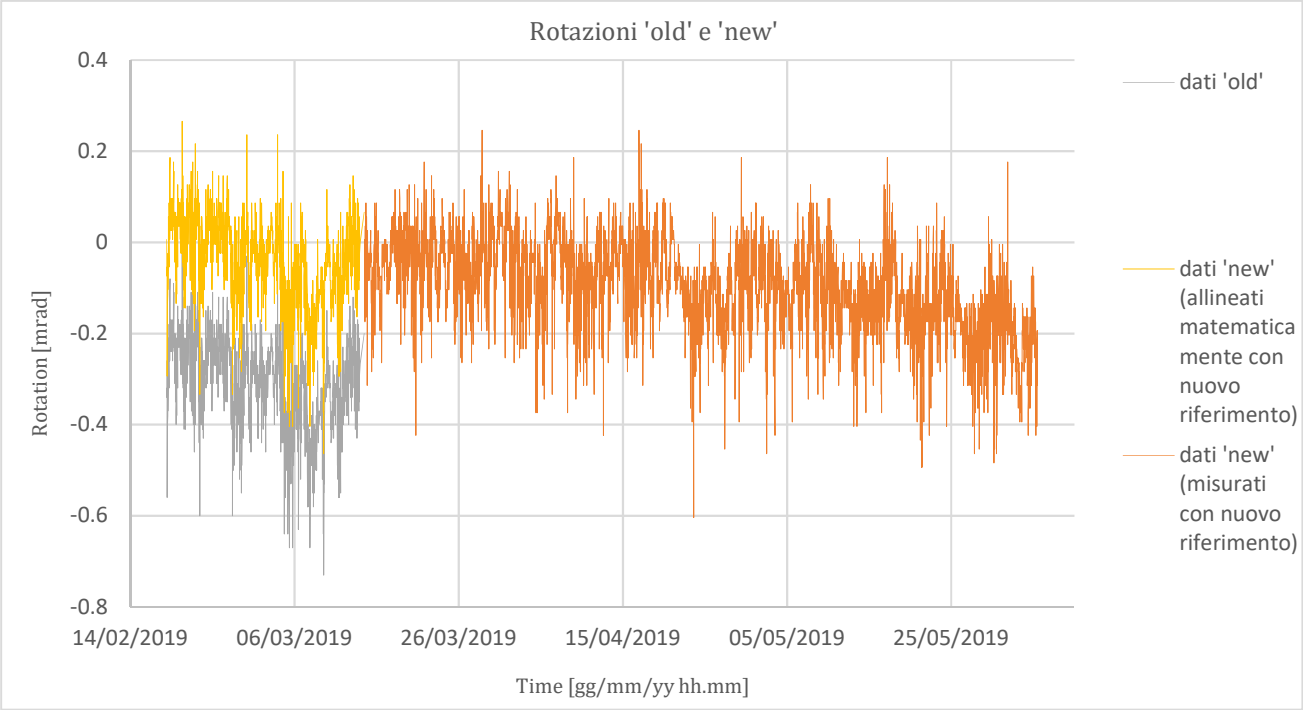


Figura 15: Grafico delle rotazioni 'old' e 'new' per il sensore EL_C23-T2-N

2.3. Creazione di un valore di zero a ponte scarico

La serie così ricavata viene quindi depurata del valore di rotazione ottenuto a ponte scarico, definito durante la prova di collaudo.

Si ottiene così la serie di rotazioni grezze ('raw') che è il data-set di partenza su cui si effettueranno le operazioni desiderate. La formula usata è la seguente:

$$\theta_{raw}(t) = \theta(t) - \theta(t_0)$$

Dove:

- $\theta_{raw}(t)$ è la rotazione grezza ('raw') all'istante t;
- $\theta(t)$ è la rotazione misurata dal sensore all'istante t;
- $\theta(t_0)$ è la rotazione misurata a ponte scarico.

Il valore di rotazione a ponte scarico $\theta(t_0)$ viene ricavato durante le operazioni relative alla prova di carico.

La serie di dati ottenuta per ciascun sensore nel periodo in oggetto consta di 7594 misure di rotazione e relative temperature, distribuite regolarmente a intervalli di 20 minuti.
Si è scelto come verso positivo della rotazione quello mostrato in *Figura 12*, avendo assunto negative le rotazioni che comportano un'inflessione verso il basso dell'impalcato.

Figura 12: verso positivo delle rotazioni

3. ANALISI DEI DATI

3.1. Analisi dei dati grezzi

Sono state effettuate delle operazioni di tipo statistico sulla serie di rotazioni acquisite dal sensore (rotazioni 'raw'), quali il calcolo del valor medio della serie, lo scarto quadratico medio, la media mobile centrata calcolate su finestre temporali di 2 ore e 20 minuti, 3 ore e 3 ore e 40 minuti (considerando un tempo di campionamento di 20 minuti, queste sono rispettivamente relative a 7, 9 e 11 misure), la differenza tra il segnale originale e la media mobile su 180 minuti, e lo scarto quadratico medio di questa serie di differenze.
Si riportano di seguito le formulazioni usate:

$$\bar{x} = \frac{\sum_{i=1}^{N} x(t_i)}{N}$$

$$\sigma_x = \frac{\sqrt{\sum_{i=1}^{N} (x(t_i) - \bar{x})^2}}{N}$$

$$MR(2\ h\ 20\ min)(t) = \frac{\sum_{i=-3}^{3} x(t + i\Delta t)}{7}$$

$$MR(3\ h)(t) = \frac{\sum_{i=-4}^{4} x(t + i\Delta t)}{9}$$

$$MR(3\ h\ 40\ min)(t) = \frac{\sum_{i=-5}^{5} x(t + i\Delta t)}{11}$$

Dove, per una serie di N dati:

- $x(t_i)$ è l'i-esimo elemento della serie, in questo caso la rotazione all'istante t_i;
- $\bar{x}$ è il valor medio della serie, in questo caso la rotazione media;
- σ_x è lo scarto quadratico medio della serie;
- $MR(2\ h\ 20\ min)(t)$ è la media mobile centrata su 7 valori, calcolata all'istante t;
- $MR(3\ h)(t)$ è la media mobile centrata su 9 valori, calcolata all'istante t;
- $MR(3\ h\ 40\ min)(t)$ è la media mobile centrata su 11 valori, calcolata all'istante t.

Si noti che il numero di misurazioni considerate per calcolare una media mobile in un dato intervallo di tempo è strettamente collegato al tempo di campionamento. Per riferirsi alle medie mobili calcolate su 2 ore e 20 minuti, 3 ore e 3 ore e 40 minuti, considerando che con un tempo di campionamento di 20 minuti esse si riferiscono rispettivamente a 7, 9 e 11 misure, si userà la denominazione MR7, MR9 e MR11.

Si sottolinea il fatto che l'algoritmo usato per i calcoli permette all'utente di indicare la durata delle tre finestre temporali in cui eseguire la media mobile.

Segue uno schema esplicativo per il calcolo della media mobile centrata su 9 valori (3 ore).

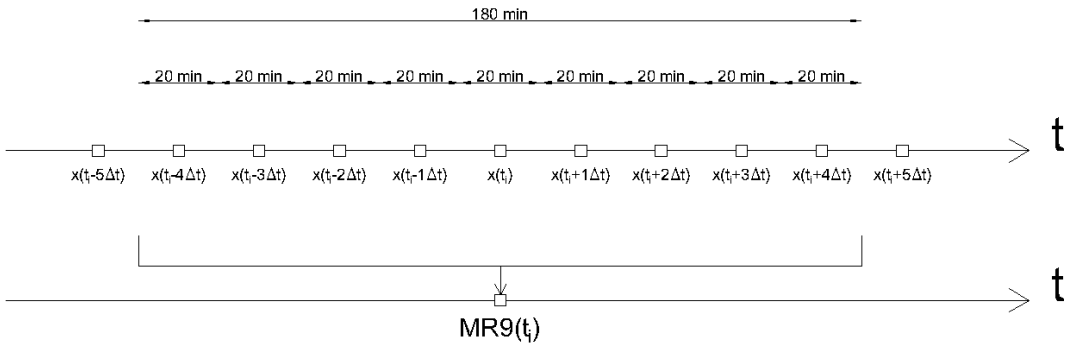


Figura 16: Media mobile su 9 valori centrata

La seguente *Tabella 1* riporta i valori medi e gli scarti quadratici medi relativi alla serie di rotazioni grezze di ciascun sensore.

Tabella 1: Valor medio e scarto quadratico medio delle rotazioni grezze

CODICE SENSORE	MEDIA RAW [mrad]	SQM RAW [mrad]
C23-T2-N	-0.072	0.093
C23-T2-S	-0.058	0.072
C23-T3-N	0.006	0.102
C23-T3-S	-0.072	0.086
C23-T4-N	-0.048	0.066
C23-T4-S	-0.065	0.086
C23-T5-N	-0.058	0.077
C23-T5-S	-0.205	0.169

Analogamente, a partire dalla serie di temperature misurate, sono state calcolate le medie mobili su 60, 80, 100 e 120 minuti antecedenti l'istante considerato (rispettivamente relative a 3, 4, 5 e 6 misure per un tempo di campionamento pari a 20 minuti).
Il punto centrale della media mobile su 2 ore antecedenti (6 misure), dunque, si trova arretrato di un'ora rispetto all'istante considerato t.
Sono state quindi calcolate le medie mobili su 120 minuti antecedenti, con un ritardo da 1 a 23 ore. Si evidenzia che con un ritardo di 23 ore, il valore centrale della media ha un ritardo di 24 ore rispetto all'istante considerato.
Questa operazione può servire per tenere in considerazione il fatto che la risposta del ponte può non essere immediatamente successiva a una variazione di temperatura a causa dell'inerzia termica dell'impalcato. Si riportano di seguito le formulazioni usate:

$$MT(1\ h)(t) = \frac{\sum_{i=-3}^{-1} y(t + i\Delta t)}{3}$$

$$MT(1\ h\ 20\ \text{min})(t) = \frac{\sum_{i=-4}^{-1} y(t + i\Delta t)}{4}$$

$$MT(1\ h\ 40\ \text{min})(t) = \frac{\sum_{i=-5}^{-1} y(t + i\Delta t)}{5}$$

$$MT(2\ h)(t) = \frac{\sum_{i=-6}^{-1} y(t + i\Delta t)}{6}$$

$$MT(2\ h)\_r1(t) = \frac{\sum_{i=-9}^{-4} y(t + i\Delta t)}{6}$$

$$MT(2\ h)\_r2(t) = \frac{\sum_{i=-12}^{-7} y(t + i\Delta t)}{6}$$

$$MT(2\ h)\_rn(t) = \frac{\sum_{i=-(6+3n)}^{-(1+3n)} y(t + i\Delta t)}{6}$$

Dove, per una serie di N dati:

- $MT(1\ h)(t)$ è la media mobile su 3 valori precedenti, calcolata all'istante t;
- $MT(1\ h\ 20\ min)4(t)$ è la media mobile su 4 valori precedenti, calcolata all'istante t;
- $MT(1\ h\ 40\ min)(t)$ è la media mobile su 5 valori precedenti, calcolata all'istante t;
- $MT(2\ h)(t)$ è la media mobile su 6 valori precedenti, calcolata all'istante t;
- $MT(2\ h)\_r1(t)$ è la media mobile su 6 valori precedenti, con ritardo di 1 ora, calcolata all'istante t;
- $MT(2\ h)\_r2(t)$ è la media mobile su 6 valori precedenti, con ritardo di 2 ore, calcolata all'istante t;
- $MT(2\ h)\_rn(t)$ è la media mobile su 6 valori precedenti, con ritardo di n ore, calcolata all'istante t (con n=1, 23).

Ovviamente, anche in questo caso il numero di misurazioni utilizzate per il calcolo di una media mobile in un dato intervallo di tempo dipende dal tempo di campionamento. Per riferirsi alle medie mobili calcolate su 60, 80, 100 e 120 minuti, considerando che con un tempo di campionamento di 20 minuti esse si riferiscono rispettivamente a 3, 4, 5 e 6 misure, si userà la denominazione MT3, MT4, MT5 e MT6.

Si evidenzia che nel codice descritto al capitolo 4, usato per l'analisi, non è possibile definire da parte dell'utente la durata delle finestre temporali usate per il calcolo delle medie mobili sulla temperatura; queste sono fissate agli intervalli temporali definiti.

Viene riportato lo schema usato per il calcolo delle medie mobili su 6 misure antecedenti, e quello usato per calcolare le medie mobili su 6 valori, con ritardo di un'ora.

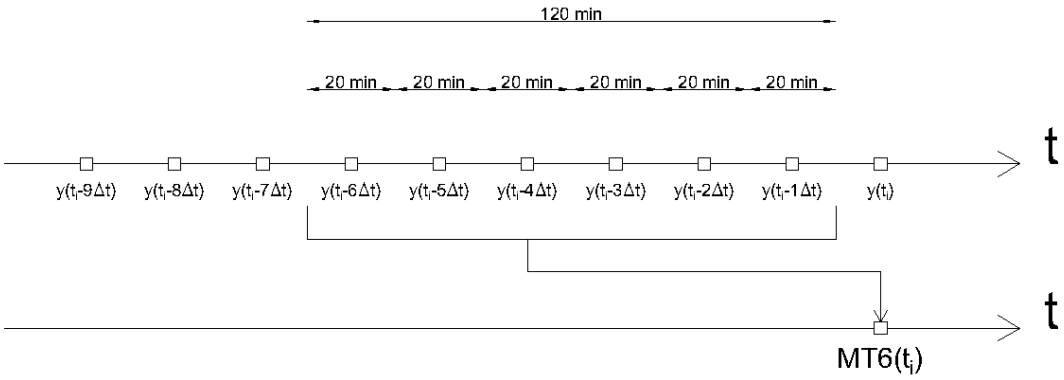


Figura 17: Media mobile su 6 valori di temperatura precedenti al tempo t_i

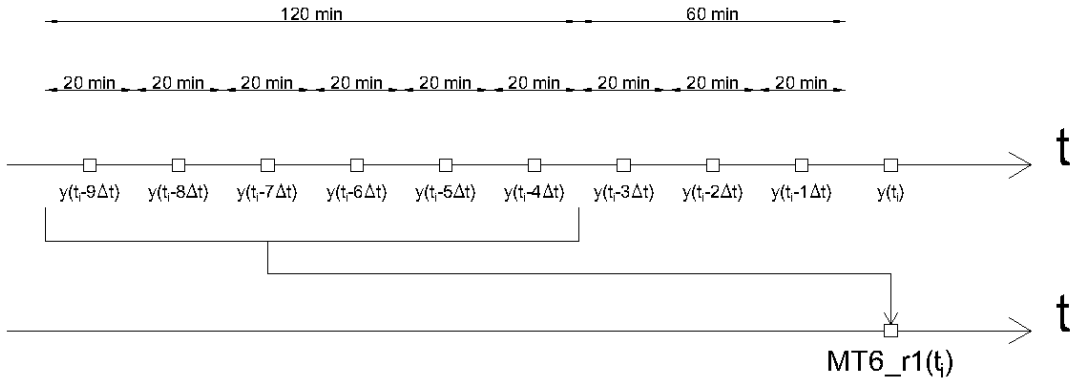


Figura 18: Media mobile su 6 valori di temperatura precedenti al tempo t_i, con ritardo di un'ora

3.2. Coefficienti di correlazione dei dati grezzi

3.2.1. Coefficiente di correlazione

In statistica, il coefficiente di correlazione lineare, detto anche indice di correlazione di Pearson, tra due serie di dati è un indice che esprime un'eventuale relazione di linearità tra esse. Esso può assumere un valore compreso tra 1 e -1: il valore di correlazione che rappresenta una perfetta correlazione lineare positiva ($x = ky$) è pari ad 1, mentre quello corrispondente alla perfetta correlazione lineare negativa ($x = -ky$) è -1.
Il coefficiente di correlazione che indice un'assenza di correlazione lineare (variabili completamente svincolate) è 0.

Considerando due serie di dati X e Y aventi entrambi lunghezza pari ad N, esso viene definito come:

$$Correlazione(X, Y) = \rho_{xy} = \frac{\sigma_{xy}}{\sigma_x \cdot \sigma_y}$$

Dove:

- σ_{xy} è la covarianza tra le due serie di dati;
- σ_x, σ_y sono gli scarti quadratici medi (detti anche deviazioni standard) rispettivamente delle due serie di dati.

Esplicitando i termini della formula scritta sopra, tramite alcuni passaggi matematici si ottiene:

$$Correlazione(X,Y) = \rho_{xy} = \frac{\sum(x-\bar{x})(y-\bar{y})}{\sqrt{\sum(x-\bar{x})^2 \sum(y-\bar{y})^2}}$$

Dove $\bar{x}$ e $\bar{y}$ sono i valori medi delle serie X e Y.

3.2.2. Correlazione tra rotazioni raw e temperature

Sono stati quindi calcolati i valori di correlazione tra le rotazioni grezze (rotazioni 'raw') e le temperature istantanee, con la formula indicata.
È stata calcolata la correlazione tra:

- rot: rotazioni istantanee;
- MR7: media mobile centrata su 7 letture di rotazione (2 ore e 20 minuti);
- MR9: media mobile centrata su 9 letture di rotazione (3 ore);
- MR11: media mobile centrata su 11 letture di rotazione (3 ore e 40 minuti).

e le temperature istantanee.

Le medie mobili sono utili per poter depurare gli effetti istantanei del traffico e delle variazioni di temperatura istantanee, che hanno un effetto prevalente sulla strumentazione e non sull'impalcato.

Si osserva che le rotazioni istantanee hanno una correlazione alta, in valore assoluto, con le temperature, ma le rotazioni mediate, a prescindere dall'ampiezza dell'intervallo su cui avviene la media, hanno una correlazione più alta con le temperature, sempre in valore assoluto.
Quindi, eliminata la variabilità del traffico, esiste una correlazione altissima tra rotazioni e temperature istantanee. Si riportano di seguito i risultati ottenuti.

Tabella 2: Coefficienti di correlazione tra rotazioni 'raw' (istantanee e mediate) e temperatura istantanea

CODICE SENSORE		temp. istant.
C23-T2-N	rot. istant.	-0.630
	MR7	-0.743
	MR9	-0.747
	MR11	-0.750
C23-T2-S	rot. istant.	-0.482
	MR7	-0.582
	MR9	-0.587
	MR11	-0.589
C23-T3-N	rot. istant.	0.433
	MR7	0.610
	MR9	0.621
	MR11	0.629
C23-T3-S	rot. istant.	-0.665
	MR7	-0.773
	MR9	-0.777
	MR11	-0.779

CODICE SENSORE		temp. istant.
C23-T4-N	rot. istant.	-0.462
	MR7	-0.559
	MR9	-0.563
	MR11	-0.566
C23-T4-S	rot. istant.	-0.410
	MR7	-0.483
	MR9	-0.487
	MR11	-0.49
C23-T5-N	rot. istant.	-0.593
	MR7	-0.696
	MR9	-0.700
	MR11	-0.702
C23-T5-S	rot. istant.	-0.855
	MR7	-0.877
	MR9	-0.876
	MR11	-0.875

Sono stati calcolati gli stessi valori di correlazione non solo con le temperature istantanee, ma anche con le medie delle temperature misurate nei 60, 80, 100, 120 minuti precedenti alla misura della rotazione (rispettivamente MT3, MT4, MT5, MT6), immaginando che l'effetto di una azione termica sul ponte abbia bisogno di tempo per produrre dei risultati e che quindi la rotazione misurata al tempo t sia frutto dell'integrale delle temperature su un arco di tempo precedente a t. Anche in questo caso vengono esposti di seguito i risultati.

Tabella 3: Coefficienti di correlazione tra rotazioni 'raw' (istantanee e mediate) e temperatura (istantanea e mediata)

CODICE SENSORE	TEMPER. / ROTAZ.	temp. istant.	MT3	MT4	MT5	MT6
C23-T2-N	rot. istant.	-0.630	-0.605	-0.597	-0.589	-0.581
	MR7	-0.743	-0.714	-0.706	-0.697	-0.688
	MR9	-0.747	-0.719	-0.711	-0.703	-0.694
	MR11	-0.750	-0.722	-0.714	-0.706	-0.697
C23-T2-S	rot. istant.	-0.482	-0.454	-0.447	-0.440	-0.433
	MR7	-0.582	-0.552	-0.544	-0.535	-0.527
	MR9	-0.587	-0.557	-0.549	-0.540	-0.532
	MR11	-0.589	-0.560	-0.552	-0.544	-0.535
C23-T3-N	rot. istant.	0.433	0.456	0.462	0.467	0.472
	MR7	0.610	0.643	0.651	0.658	0.665
	MR9	0.621	0.655	0.663	0.671	0.678
	MR11	0.629	0.664	0.672	0.679	0.687
C23-T3-S	rot. istant.	-0.665	-0.639	-0.632	-0.624	-0.616
	MR7	-0.773	-0.747	-0.738	-0.730	-0.721
	MR9	-0.777	-0.751	-0.743	-0.735	-0.726
	MR11	-0.779	-0.753	-0.746	-0.738	-0.729
C23-T4-N	rot. istant.	-0.462	-0.430	-0.421	-0.413	-0.405
	MR7	-0.559	-0.522	-0.512	-0.503	-0.493
	MR9	-0.563	-0.527	-0.518	-0.508	-0.499
	MR11	-0.566	-0.531	-0.521	-0.512	-0.503
C23-T4-S	rot. istant.	-0.41	-0.401	-0.399	-0.397	-0.395
	MR7	-0.483	-0.471	-0.469	-0.466	-0.464
	MR9	-0.487	-0.475	-0.472	-0.470	-0.468
	MR11	-0.490	-0.478	-0.475	-0.473	-0.471
C23-T5-N	rot. istant.	-0.593	-0.565	-0.557	-0.55	-0.542
	MR7	-0.696	-0.666	-0.658	-0.649	-0.640
	MR9	-0.700	-0.671	-0.663	-0.654	-0.645
	MR11	-0.702	-0.674	-0.666	-0.658	-0.649
C23-T5-S	rot. istant.	-0.855	-0.841	-0.837	-0.833	-0.828
	MR7	-0.877	-0.867	-0.863	-0.859	-0.855
	MR9	-0.876	-0.867	-0.864	-0.860	-0.856
	MR11	-0.875	-0.866	-0.863	-0.860	-0.856

Si può osservare come tutti i valori di correlazione scendano in valore assoluto se si estende la media sulle temperature ad un periodo più lungo, contrariamente a quello che si può pensare a seguito di un'azione reale sulla struttura. La correlazione più alta rimane sulle temperature istantanee lasciando intuire che si tratti di una risposta della strumentazione e non della struttura all'effetto termico.

3.3. Calcolo della deriva termica istantanea

Si procede quindi alla correzione della serie di rotazioni grezze ('raw') rispetto alle temperature istantanee.
La formula usata per il calcolo della rotazione corretta è riportata di seguito:

$$\theta_K(t) = \theta_{raw}(t) - D * [T(t) - T(to)]$$

Dove:

- $\theta_K(t)$: serie di rotazioni corrette;
- $\theta_{raw}(t)$: serie di rotazioni grezze, 'raw';
- D : valore di deriva termica istantanea;
- $T(t)$: serie di temperature istantanee;
- $T(to)$: temperatura relativa alla misura a ponte scarico.

Il valore calcolato per la deriva termica istantanea D è tale per cui il coefficiente di correlazione calcolato tra le rotazioni corrette e la temperatura istantanea (valori evidenziati in celeste nella *Tabella 3*) sia il più possibile vicino allo 0.

Per ottenere il valore ottimale della deriva termica istantanea, è stato usato il metodo della bisezione, usando come criterio di ottimizzazione la minimizzazione del valore di correlazione tra rotazioni corrette e temperatura istantanea.
Si considera un intervallo di valori che possono essere assunti dalla deriva istantanea, e si calcola la serie di rotazioni corrette usando come valore di deriva istantanea ciascuno dei due estremi dell'intervallo; per ciascuna di queste serie si calcola il valore di correlazione con le temperature istantanee. Quindi, si assume come deriva istantanea il valore intermedio tra i due precedenti.

Si calcola la serie di rotazioni corrette e il relativo valore di correlazione con le temperature. Se il valore di correlazione relativo al primo estremo è minore di quello relativo al secondo, il valor medio diventa il nuovo estremo sinistro dell'intervallo, altrimenti diventa il destro.

Si prosegue con nuovi intervalli, fintanto che la differenza tra i due estremi dell'intervallo è maggiore della precisione cercata per la deriva. Il valore assunto dagli estremi dell'intervallo è il valore cercato per la deriva istantanea.

Le derive termiche istantanee ottenute sono riportati in *Tabella 4*.

Tabella 4: Deriva termica istantanea

CODICE SENSORE	DERIVA ISTANTANEA [mrad/°C]
C23-T2-N	-0.0161
C23-T2-S	-0.0096
C23-T3-N	0.0121
C23-T3-S	-0.0156
C23-T4-N	-0.0085
C23-T4-S	-0.0097
C23-T5-N	-0.0129
C23-T5-S	-0.0394

Ricordiamo che la deriva termica degli strumenti (dichiarata dal produttore) potrebbe essere intorno a 0.020 mrad, quindi derive prossime o inferiori a questo valore sono del tutto compatibili con le tolleranze di produzione.

3.4. Analisi dei dati corretti

Le rotazioni corrette o compensate con la deriva istantanea vengono dette “rotazioni k”, e non sono più correlate con le temperature istantanee.

Per questa nuova serie vengono nuovamente calcolati il valor medio, lo scarto quadratico medio, le medie mobili, la differenza tra serie compensata e sua media mobile su 9 valori, usando le formule già riportate.

Si può osservare che le rotazioni compensate sono molto più centrate sul valore di zero rispetto a quelle grezze e le loro escursioni sono più ridotte, quindi la compensazione istantanea in temperatura ha prodotto un segnale molto più stabile di quello grezzo.

Una misura quantitativa di questo fenomeno è fornita dal confronto tra i valori medi e gli scarti quadratici medi dei due segnali. Si riportano di seguito i risultati.

Tabella 5: Valor medio e scarto quadratico medio delle rotazioni grezze e compensate

CODICE SENSORE	MEDIA RAW [mrad]	SQM RAW [mrad]	MEDIA K [mrad]	SQM K [mrad]
C23-T2-N	-0.072	0.093	-0.017	0.070
C23-T2-S	-0.058	0.072	-0.024	0.062
C23-T3-N	0.006	0.102	-0.037	0.090
C23-T3-S	-0.072	0.086	-0.019	0.063
C23-T4-N	-0.048	0.066	-0.019	0.057
C23-T4-S	-0.065	0.086	-0.031	0.077
C23-T5-N	-0.058	0.077	-0.010	0.062
C23-T5-S	-0.205	0.169	-0.067	0.087
MEDIE	-0,072	0,094	-0,028	0,071

Si osserva che le rotazioni grezze forniscono un valore medio di rotazione che corrisponde a una inflessione verso il basso, con uno scarto quadratico medio dovuto sia al traffico che alla termica, mentre quelle compensate danno una rotazione media (sempre verso il basso) e uno scarto quadratico medio inferiori, avendo probabilmente tolto una quota di fenomeno termico.

Unica eccezione è il sensore T3-N (trave 3, lato nord della campata), in cui il valor medio delle rotazioni grezze (‘raw’) assume un valore positivo, ma molto vicino allo zero, mentre

il valor medio delle rotazioni compensate torna ad essere negativo (rotazione verso il basso) e con un valore di poco più distante dallo zero.
Ricordiamo inoltre che la risoluzione degli strumenti è intorno a 0.020 mrad, quindi misure prossime o inferiori a questo valore non hanno significato fisico.

3.5. Coefficienti di correlazione dei dati corretti

Successivamente, sono stati calcolati i coefficienti di correlazione tra rotazioni compensate. In particolare, è stata calcolata la correlazione tra le rotazioni:

- rot. istant. k: rotazioni istantanee corrette;
- MR7_k: media mobile centrata su 7 letture di rotazione corrette (2 ore e 20 minuti);
- MR9_k: media mobile centrata su 9 letture di rotazione corrette (3 ore);
- MR11_k: media mobile centrata su 11 letture di rotazione corrette (3 ore e 40 minuti).

e le temperature:

- temp. istant.: temperatura istantanea;
- MT3: media mobile su 3 letture di temperatura antecedenti (1 ora);
- MT4: media mobile su 4 letture di temperatura antecedenti (1 ora e 20 minuti);
- MT5: media mobile su 5 letture di temperatura antecedenti (1 ora e 40 minuti);
- MT6: media mobile su 6 letture di temperatura antecedenti (2 ore);
- MT6_rn: media mobile su 6 letture di temperatura antecedenti con un ritardo di n ore (2 ore).

Si riportano di seguito i risultati.

Tabella 6: Coefficienti di correlazione tra rotazioni corrette (istantanee e mediate) e temperatura (istantanea, mediata e mediata con ritardo)

CODICE SENSORE	TEMPER. / ROTAZ.	temp. istant.	MT3	MT4	MT5	MT7C (da -1h a 1h)	MT6 (da -2h a 0h)	MT6_r1 (da -3h a -1h)	MT6_r2 (da -4h a -2h)	MT6_r3 (da -5h a -3h)	MT6_r4 (da -6h a -4h)
C23-T2-N	rot. istant. k	0.001	0.026	0.032	0.039	0	0.045	0.084	0.111	0.127	0.130
	MR7 k	0	0.036	0.044	0.052	-0.001	0.061	0.115	0.150	0.172	0.175
	MR9 k	-0.001	0.036	0.044	0.053	-0.002	0.061	0.115	0.151	0.172	0.175
	MR11 k	-0.002	0.035	0.044	0.053	-0.002	0.061	0.115	0.151	0.171	0.174
C23-T2-S	rot. istant. k	0.002	0.027	0.032	0.037	0	0.042	0.076	0.097	0.106	0.099
	MR7 k	0	0.033	0.040	0.047	-0.002	0.054	0.098	0.124	0.135	0.126
	MR9 k	-0.001	0.032	0.039	0.047	-0.003	0.054	0.098	0.124	0.134	0.126
	MR11 k	-0.003	0.031	0.038	0.046	-0.004	0.053	0.097	0.122	0.132	0.124
C23-T3-N	rot. istant. k	0.001	0.032	0.04	0.048	0.003	0.056	0.109	0.149	0.176	0.192
	MR7 k	0.005	0.052	0.065	0.077	0.007	0.090	0.174	0.236	0.280	0.306
	MR9 k	0.007	0.055	0.068	0.081	0.009	0.093	0.179	0.241	0.287	0.314
	MR11 k	0.009	0.058	0.07	0.083	0.011	0.096	0.181	0.245	0.291	0.319
C23-T3-S	rot. istant. k	0.003	0.026	0.031	0.037	0.001	0.042	0.078	0.102	0.118	0.124
	MR7 k	0.001	0.035	0.043	0.050	0	0.058	0.106	0.139	0.161	0.168
	MR9 k	0	0.034	0.042	0.050	0	0.058	0.107	0.140	0.162	0.169
	MR11 k	-0.001	0.034	0.042	0.05	-0.001	0.058	0.107	0.140	0.162	0.168
C23-T4-N	rot. istant. k	0.001	0.032	0.039	0.046	-0.001	0.052	0.091	0.111	0.117	0.108
	MR7 k	-0.001	0.039	0.048	0.057	-0.002	0.065	0.115	0.140	0.148	0.137
	MR9 k	-0.002	0.038	0.047	0.056	-0.003	0.064	0.115	0.140	0.148	0.137
	MR11 k	-0.003	0.037	0.046	0.054	-0.005	0.063	0.113	0.139	0.146	0.135
C23-T4-S	rot. istant. k	0.002	0.006	0.006	0.006	-0.005	0.005	-0.001	-0.015	-0.038	-0.071
	MR7 k	-0.005	0.004	0.005	0.004	-0.011	0.004	-0.003	-0.021	-0.050	-0.089
	MR9 k	-0.009	0.001	0.002	0.002	-0.014	0.002	-0.005	-0.023	-0.052	-0.092
	MR11 k	-0.014	-0.002	-0.001	-0.001	-0.018	-0.001	-0.007	-0.025	-0.055	-0.095
C23-T5-N	rot. istant. k	0	0.028	0.034	0.041	0	0.046	0.083	0.103	0.110	0.103
	MR7 k	0	0.036	0.044	0.052	-0.001	0.060	0.107	0.133	0.142	0.133
	MR9 k	0	0.035	0.044	0.052	-0.001	0.059	0.107	0.133	0.142	0.133
	MR11 k	-0.001	0.035	0.043	0.051	-0.002	0.058	0.106	0.131	0.140	0.132
C23-T5-S	rot. istant. k	0.002	0.006	0.006	0.005	-0.005	0.004	-0.007	-0.026	-0.053	-0.086
	MR7 k	-0.005	0.003	0.003	0.002	-0.010	0.001	-0.01	-0.032	-0.062	-0.1
	MR9 k	-0.009	0	0	0	-0.013	-0.001	-0.012	-0.033	-0.063	-0.102
	MR11 k	-0.013	-0.004	-0.003	-0.003	-0.017	-0.004	-0.014	-0.035	-0.065	-0.104

Tabella 7: coefficienti di correlazione tra rotazioni corrette (istantanee e mediate) e temperatura (istantanea, mediata e mediata con ritardo)

CODICE SENSORE	TEMPER. / ROTAZ.	MT6_r5 (da -7h a -5h)	MT6_r6 (da -8h a 6h)	MT6_r7 (da -9h a 7h)	MT6_r8 (da -10h a -8h)	MT6_r9 (da -11h a -9h)	MT6_r10 (da -12h a 10h)	MT6_r11 (da -13h a -11h)	MT6_r12 (da -14h a -12h)	MT6_r13 (da -15h a -13h)	MT6_r14 (da -16h a -14h)
C23-T2-N	rot. istant. k	0.118	0.092	0.058	0.018	-0.026	-0.069	-0.110	-0.147	-0.181	-0.207
	MR7 k	0.159	0.125	0.079	0.024	-0.036	-0.095	-0.151	-0.203	-0.248	-0.284
	MR9 k	0.159	0.126	0.079	0.023	-0.037	-0.097	-0.154	-0.206	-0.252	-0.287
	MR11 k	0.158	0.125	0.078	0.023	-0.037	-0.097	-0.155	-0.208	-0.254	-0.289
C23-T2-S	rot. istant. k	0.078	0.048	0.009	-0.038	-0.084	-0.127	-0.165	-0.199	-0.227	-0.248
	MR7 k	0.100	0.060	0.009	-0.050	-0.110	-0.166	-0.216	-0.261	-0.298	-0.324
	MR9 k	0.100	0.060	0.008	-0.051	-0.111	-0.168	-0.219	-0.264	-0.301	-0.328
	MR11 k	0.099	0.058	0.007	-0.051	-0.112	-0.169	-0.221	-0.266	-0.303	-0.33
C23-T3-N	rot. istant. k	0.196	0.187	0.168	0.144	0.112	0.076	0.039	0.002	-0.031	-0.059
	MR7 k	0.312	0.299	0.270	0.229	0.179	0.122	0.063	0.005	-0.048	-0.093
	MR9 k	0.32	0.307	0.278	0.236	0.184	0.126	0.065	0.006	-0.048	-0.095
	MR11 k	0.325	0.312	0.283	0.240	0.188	0.129	0.067	0.007	-0.049	-0.096
C23-T3-S	rot. istant. k	0.116	0.095	0.067	0.032	-0.008	-0.048	-0.086	-0.121	-0.151	-0.175
	MR7 k	0.158	0.13	0.091	0.043	-0.011	-0.066	-0.118	-0.167	-0.208	-0.241
	MR9 k	0.158	0.131	0.091	0.043	-0.011	-0.067	-0.120	-0.169	-0.211	-0.244
	MR11 k	0.157	0.131	0.091	0.042	-0.011	-0.067	-0.121	-0.170	-0.212	-0.245
C23-T4-N	rot. istant. k	0.086	0.052	0.007	-0.043	-0.095	-0.148	-0.196	-0.237	-0.267	-0.288
	MR7 k	0.109	0.064	0.008	-0.057	-0.124	-0.192	-0.254	-0.306	-0.347	-0.374
	MR9 k	0.108	0.064	0.007	-0.058	-0.127	-0.194	-0.257	-0.310	-0.351	-0.379
	MR11 k	0.107	0.062	0.005	-0.060	-0.128	-0.196	-0.259	-0.312	-0.353	-0.382
C23-T4-S	rot. istant. k	-0.112	-0.159	-0.211	-0.262	-0.310	-0.35	-0.382	-0.405	-0.415	-0.414
	MR7 k	-0.139	-0.196	-0.259	-0.321	-0.378	-0.427	-0.467	-0.494	-0.506	-0.506
	MR9 k	-0.142	-0.199	-0.262	-0.324	-0.381	-0.431	-0.470	-0.497	-0.510	-0.510
	MR11 k	-0.145	-0.202	-0.264	-0.326	-0.384	-0.434	-0.473	-0.50	-0.513	-0.513
C23-T5-N	rot. istant. k	0.084	0.055	0.017	-0.025	-0.07	-0.114	-0.156	-0.189	-0.216	-0.236
	MR7 k	0.108	0.070	0.021	-0.034	-0.093	-0.152	-0.206	-0.251	-0.286	-0.311
	MR9 k	0.108	0.070	0.021	-0.035	-0.095	-0.154	-0.208	-0.254	-0.290	-0.315
	MR11 k	0.107	0.069	0.020	-0.037	-0.096	-0.155	-0.210	-0.256	-0.292	-0.317
C23-T5-S	rot. istant. k	-0.127	-0.173	-0.222	-0.269	-0.311	-0.344	-0.368	-0.382	-0.384	-0.377
	MR7 k	-0.146	-0.198	-0.253	-0.306	-0.352	-0.39	-0.417	-0.432	-0.435	-0.427
	MR9 k	-0.148	-0.200	-0.255	-0.307	-0.354	-0.391	-0.418	-0.434	-0.437	-0.430
	MR11 k	-0.150	-0.202	-0.256	-0.308	-0.355	-0.392	-0.419	-0.435	-0.438	-0.431

Tabella 8: Coefficienti di correlazione tra rotazioni corrette (istantanee e mediate) e temperatura (istantanea, mediata e mediata con ritardo)

CODICE SENSORE	TEMPER. / ROTAZ.	MT6_r15 (da -17h a -15h)	MT6_r16 (da -18h a -16h)	MT6_r17 (da -19h a -17h)	MT6_r18 (da -20h a -18h)	MT6_r19 (da -21h a -19h)	MT6_r20 (da -22h a -20h)	MT6_r21 (da -23h a -21h)	MT6_r22 (da -24h a -22h)	MT6_r23 (da -25h a -23h)
C23-T2-N	rot. istant. k	-0.224	-0.231	-0.228	-0.215	-0.191	-0.160	-0.124	-0.086	-0.050
	MR7 k	-0.307	-0.317	-0.313	-0.294	-0.262	-0.220	-0.171	-0.120	-0.071
	MR9 k	-0.311	-0.321	-0.317	-0.298	-0.266	-0.224	-0.175	-0.123	-0.073
	MR11 k	-0.313	-0.323	-0.319	-0.300	-0.268	-0.226	-0.177	-0.125	-0.075
C23-T2-S	rot. istant. k	-0.260	-0.260	-0.250	-0.231	-0.204	-0.169	-0.130	-0.090	-0.052
	MR7 k	-0.339	-0.340	-0.327	-0.303	-0.267	-0.222	-0.171	-0.119	-0.071
	MR9 k	-0.342	-0.343	-0.331	-0.306	-0.270	-0.225	-0.174	-0.122	-0.073
	MR11 k	-0.344	-0.345	-0.333	-0.308	-0.272	-0.227	-0.176	-0.124	-0.075
C23-T3-N	rot. istant. k	-0.082	-0.098	-0.105	-0.102	-0.090	-0.069	-0.041	-0.007	0.031
	MR7 k	-0.130	-0.155	-0.166	-0.162	-0.142	-0.109	-0.064	-0.010	0.049
	MR9 k	-0.133	-0.158	-0.170	-0.165	-0.145	-0.111	-0.065	-0.009	0.051
	MR11 k	-0.134	-0.160	-0.171	-0.167	-0.147	-0.112	-0.065	-0.009	0.052
C23-T3-S	rot. istant. k	-0.190	-0.196	-0.192	-0.178	-0.154	-0.125	-0.093	-0.057	-0.017
	MR7 k	-0.261	-0.269	-0.264	-0.244	-0.213	-0.173	-0.128	-0.078	-0.025
	MR9 k	-0.265	-0.273	-0.267	-0.247	-0.216	-0.176	-0.130	-0.079	-0.026
	MR11 k	-0.266	-0.274	-0.268	-0.249	-0.218	-0.177	-0.131	-0.079	-0.027
C23-T4-N	rot. istant. k	-0.300	-0.299	-0.286	-0.263	-0.230	-0.189	-0.145	-0.099	-0.054
	MR7 k	-0.388	-0.387	-0.371	-0.341	-0.299	-0.247	-0.189	-0.130	-0.073
	MR9 k	-0.393	-0.392	-0.375	-0.345	-0.302	-0.250	-0.192	-0.132	-0.075
	MR11 k	-0.395	-0.394	-0.378	-0.348	-0.305	-0.253	-0.195	-0.135	-0.078
C23-T4-S	rot. istant. k	-0.404	-0.384	-0.357	-0.325	-0.287	-0.247	-0.208	-0.174	-0.150
	MR7 k	-0.493	-0.469	-0.436	-0.397	-0.351	-0.303	-0.257	-0.216	-0.187
	MR9 k	-0.497	-0.474	-0.441	-0.401	-0.355	-0.307	-0.261	-0.221	-0.191
	MR11 k	-0.500	-0.477	-0.444	-0.404	-0.358	-0.310	-0.264	-0.225	-0.195
C23-T5-N	rot. istant. k	-0.245	-0.245	-0.235	-0.216	-0.189	-0.152	-0.111	-0.069	-0.030
	MR7 k	-0.324	-0.324	-0.310	-0.285	-0.249	-0.201	-0.147	-0.093	-0.042
	MR9 k	-0.328	-0.328	-0.314	-0.289	-0.252	-0.204	-0.150	-0.095	-0.044
	MR11 k	-0.330	-0.330	-0.317	-0.291	-0.253	-0.206	-0.152	-0.097	-0.046
C23-T5-S	rot. istant. k	-0.361	-0.337	-0.306	-0.270	-0.231	-0.190	-0.152	-0.121	-0.099
	MR7 k	-0.410	-0.383	-0.348	-0.307	-0.262	-0.217	-0.175	-0.140	-0.115
	MR9 k	-0.412	-0.385	-0.350	-0.309	-0.265	-0.220	-0.178	-0.143	-0.117
	MR11 k	-0.413	-0.386	-0.351	-0.311	-0.266	-0.222	-0.180	-0.146	-0.120

Si noti come il coefficiente di correlazione tra le rotazioni corrette e le temperature sia praticamente nullo per tutti i sensori (valori evidenziati in giallo nella *Tabella 6*); ciò deriva dall'operazione di compensazione descritta al paragrafo precedente.

Si osserva, inoltre, che le rotazioni istantanee hanno una correlazione con le temperature sempre minore di quella tra le loro medie mobili centrate e le stesse temperature, mentre le rotazioni mediate, hanno valori simili di correlazione con le temperature a prescindere dall'ampiezza dell'intervallo su cui avviene la media.

Nel grafico mostrato in *Figura 19* si osserva l'andamento dell'indice di correlazione tra la media mobile centrata su 180 minuti delle rotazioni (9 valori) e la media su 120 minuti delle temperature (6 valori), al variare del ritardo con cui si calcola la media sulle temperature; l'asse delle ascisse rappresenta il ritardo che ha il valore centrale della media delle temperature rispetto all'istante considerato, mentre le ordinate rappresentano il relativo coefficiente di correlazione.

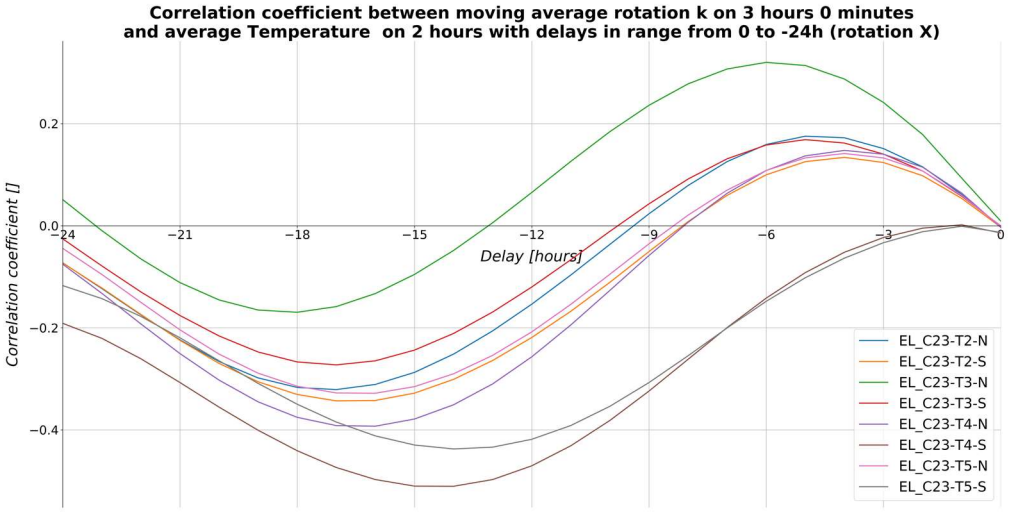


Figura 19: Coefficienti di correlazione tra rotazioni corrette mediate su 180 minuti e temperatura mediata con diversi ritardi (da 0h a -24h)

La forma della curva è simile per tutti i sensori, ed ha andamento sinusoidale, come è corretto aspettarsi da un fenomeno fisico che ha una periodicità di 24h.

Si possono distinguere tre gruppi di curve:

- Sensore T3-N (curva verde): massimo con ritardo di -6h;
- Sensori T4-S e T5-S (curve marrone e grigia): massimo con ritardo di 0h (circa);
- Sensori T2-N, T2-S, T3-S, T4-N, T5-N (altre curve): massimo con ritardo di -4.5h (circa).

Cinque su otto sensori presentano una risposta molto correlata, mentre il T3-N ha un ritardo superiore agli altri, e il T4-S e il T5-S sembrano avere un ritardo analogo a ritardo zero.

Il ritardo cui corrisponde il massimo della curva dei coefficienti di correlazione (il cui valore è evidenziato in celeste nelle *Tabelle 6* e *7*) indica a quale serie di temperature la rotazione corretta è maggiormente correlata.

In pratica, se la temperatura dell'aria sale all'istante t, l'impalcato sembra inflettersi con un certo ritardo (6h, 0h o 4.5h), compatibilmente con la sua inerzia termica.

Quindi, si presume che il dato grezzo sia una misura affetta sia da una termica sul sensore che da una termica sull'impalcato.

A seguito di questa prima pulizia si osserva soltanto la termica reale dell'impalcato.

Il passo successivo, dunque, è la correzione di questo secondo fenomeno termico.

3.6. Calcolo della deriva termica differita

Le operazioni seguenti mirano a correggere la serie di rotazioni compensate ('k') in modo tale che la serie di rotazioni doppiamente corretta non sia correlata con la media mobile delle temperature calcolata con il ritardo relativo al massimo dell'indice di correlazione.

La formula usata per il calcolo della rotazione doppiamente corretta è riportata di seguito:

$$\theta_{kk}(t) = \theta_k(t) - DD * [T(t) - T(to)]$$

Dove:

- $\theta_{kk}(t)$: serie di rotazioni doppiamente corrette;
- $\theta_k(t)$: serie di rotazioni corrette;
- DD : valore di deriva termica differita;
- $T(t)$: serie di medie mobili delle temperature calcolata con il ritardo relativo al massimo dell'indice di correlazione;
- $T(to)$: temperatura relativa alla misura a ponte scarico.

Per ottenere il valore ottimale della deriva termica differita, è stato usato lo stesso metodo citato prima, avendo come criterio di ottimizzazione la minimizzazione del valore di correlazione tra rotazioni doppiamente compensate (calcolate assumendo come T(t) la

serie di medie mobili delle temperature calcolata con il ritardo relativo al massimo dell'indice di correlazione) e media mobile della temperatura calcolata con il ritardo relativo al massimo dell'indice di correlazione.
Le derive termiche differite ottenute e l'intervallo temporale su cui è stata mediata la temperatura usata per il calcolo indicato sopra, sono riportati di seguito:

Tabella 9: Deriva termica differita

CODICE SENSORE	DERIVA DIFFERITA [mrad/°C]	RITARDO per calcolo deriva differita
C23-T2-N	0.0025	da -6h a -4h
C23-T2-S	0.0018	da -5h a -3h
C23-T3-N	0.0049	da -7h a -5h
C23-T3-S	0.0021	da -6h a -4h
C23-T4-N	0.0018	da -5h a -3h
C23-T4-S	0	da -2h a 0h
C23-T5-N	0.0019	da -5h a -3h
C23-T5-S	0	da -2h a 0h

Ricordiamo che i valori di deriva termica differita mostrati in *Tabella 9* sono relativi a un probabile movimento della struttura e non al comportamento degli strumenti. Questi valori sono però talmente bassi da ricadere ben al di sotto della sensibilità dello strumento (dichiarata dal produttore intorno a 0.020 mrad/°C) quindi le affermazioni che seguono hanno puro carattere speculativo.

Si noti che per i sensori T4-S e T5-S, i quali presentavano medie su 9 valori delle rotazioni corrette (MR9_k) maggiormente correlate con la temperatura media calcolata sulle due ore precedenti, la deriva differita ha valore nullo.
Si ottiene la serie delle rotazioni doppiamente corrette, dette "rotazioni kk".

3.7. Analisi dei dati doppiamente corretti

Come per le altre due serie di rotazioni analizzate (raw e k), anche per le rotazioni doppiamente corrette vengono calcolate le grandezze statistiche: valor medio, scarto quadratico medio, medie mobili, differenza tra serie compensata e sua media mobile su 9 valori.

Nella *Tabella 10* si mostrano i valori medi e gli scarti quadratici medi delle tre serie ottenute. Dai dati riportati si può notare come la media delle rotazioni doppiamente corrette, rispetto a quella delle rotazioni corrette, si allontani leggermente dallo 0; al contrario, lo scarto quadratico medio rimane praticamente costante. Da ciò si evince che la serie di rotazioni doppiamente corrette ha meno rumore dell'altra.

Ricordiamo che durante la taratura del sistema di misura si sono osservate rotazioni massime sulla trave 2 pari a -0.50 mrad in presenza di 4 mezzi pesanti.
Si può quindi affermare che in media, nei 4 mesi monitorati il ponte è stato inflesso verso il basso con una rotazione agli appoggi di circa -0.03 mrad, cioè circa 15 volte inferiore a quella misurata sotto carico statico.

Tabella 10: Valor medio e scarto quadratico medio delle rotazioni grezze, compensate e doppiamente compensate

CODICE SENSORE	MEDIA RAW [mrad]	SQM RAW [mrad]	MEDIA K [mrad]	SQM K [mrad]	MEDIA KK [mrad]	SQM KK [mrad]
C23-T2-N	-0.072	0.093	-0.017	0.070	-0.026	0.069
C23-T2-S	-0.058	0.072	-0.024	0.062	-0.030	0.062
C23-T3-N	0.006	0.102	-0.037	0.090	-0.054	0.088
C23-T3-S	-0.072	0.086	-0.019	0.063	-0.026	0.062
C23-T4-N	-0.048	0.066	-0.019	0.057	-0.025	0.057
C23-T4-S	-0.065	0.086	-0.031	0.077	-0.031	0.077
C23-T5-N	-0.058	0.077	-0.01	0.062	-0.016	0.061
C23-T5-S	-0.205	0.169	-0.067	0.087	-0.067	0.087
MEDIE	-0,072	0,094	-0,028	0,071	-0,034	0,070

3.8. Coefficienti di correlazione per dati doppiamente corretti

Vengono quindi calcolati i coefficienti di correlazione tra le rotazioni doppiamente corrette e le temperature. In particolare, è stata calcolata la correlazione tra le rotazioni:

- rot. istant. kk: rotazioni istantanee doppiamente corrette;
- MR7_kk: media mobile centrata su 7 letture di rotazione doppiamente corrette (2 ore e 20 minuti);
- MR9_kk: media mobile centrata su 9 letture di rotazione doppiamente corrette (3 ore);
- MR11_kk: media mobile centrata su 11 letture di rotazione doppiamente corrette (3 ore e 40 minuti).

e le temperature:

- temp. istant.: temperatura istantanea;
- MT3: media mobile su 3 letture di temperatura antecedenti (1 ora);
- MT4: media mobile su 4 letture di temperatura antecedenti (1 ora e 20 minuti);
- MT5: media mobile su 5 letture di temperatura antecedenti (1 ora e 40 minuti);
- MT6: media mobile su 6 letture di temperatura antecedenti (2 ore);
- MT6_rn: media mobile su 6 letture di temperatura antecedenti con un ritardo di n ore (2 ore).

Tabella 11: Coefficienti di correlazione tra rotazioni doppiamente corrette (istantanee e mediate) e temperatura (istantanea, mediata e mediata con ritardo)

CODICE SENSORE	TEMPER. / ROTAZ.	temp. istant.	MT3	MT4	MT5	MT7C (da -1h a 1h)	MT6 (da -2h a 0h)	MT6_r1 (da -3h a -1h)	MT6_r2 (da -4h a -2h)	MT6_r3 (da -5h a -3h)	MT6_r4 (da -6h a -4h)
C23-T2-N	rot. istant. kk	-0.091	-0.072	-0.067	-0.062	-0.093	-0.058	-0.027	-0.007	0.004	0.005
	MR7 kk	-0.128	-0.100	-0.094	-0.087	-0.130	-0.081	-0.039	-0.013	0.002	0.003
	MR9 kk	-0.132	-0.103	-0.096	-0.090	-0.134	-0.084	-0.041	-0.015	0	0.001
	MR11 kk	-0.134	-0.105	-0.098	-0.092	-0.136	-0.086	-0.043	-0.017	-0.003	-0.002
C23-T2-S	rot. istant. kk	-0.077	-0.057	-0.053	-0.049	-0.079	-0.045	-0.018	-0.001	0.006	0.001
	MR7 kk	-0.104	-0.077	-0.072	-0.066	-0.107	-0.061	-0.026	-0.005	0.004	-0.003
	MR9 kk	-0.107	-0.080	-0.074	-0.068	-0.109	-0.063	-0.027	-0.007	0.002	-0.005
	MR11 kk	-0.110	-0.083	-0.076	-0.071	-0.112	-0.065	-0.029	-0.009	-0.001	-0.008
C23-T3-N	rot. istant. kk	-0.126	-0.104	-0.098	-0.093	-0.125	-0.087	-0.049	-0.022	-0.005	0.002
	MR7 kk	-0.205	-0.172	-0.163	-0.154	-0.206	-0.145	-0.083	-0.040	-0.012	0.002
	MR9 kk	-0.212	-0.178	-0.169	-0.160	-0.212	-0.151	-0.088	-0.044	-0.015	0
	MR11 kk	-0.217	-0.182	-0.173	-0.164	-0.217	-0.155	-0.093	-0.048	-0.018	-0.003
C23-T3-S	rot. istant. kk	-0.082	-0.064	-0.06	-0.056	-0.084	-0.053	-0.027	-0.010	0.001	0.004
	MR7 kk	-0.117	-0.091	-0.085	-0.080	-0.119	-0.075	-0.039	-0.016	-0.001	0.003
	MR9 kk	-0.120	-0.094	-0.088	-0.083	-0.122	-0.077	-0.040	-0.017	-0.003	0.002
	MR11 kk	-0.123	-0.097	-0.091	-0.085	-0.125	-0.079	-0.042	-0.019	-0.005	-0.001
C23-T4-N	rot. istant. kk	-0.086	-0.061	-0.055	-0.049	-0.089	-0.044	-0.012	0.002	0.007	0
	MR7 kk	-0.115	-0.081	-0.074	-0.067	-0.117	-0.061	-0.020	0	0.005	-0.003
	MR9 kk	-0.118	-0.084	-0.077	-0.070	-0.120	-0.064	-0.022	-0.002	0.003	-0.006
	MR11 kk	-0.120	-0.087	-0.080	-0.073	-0.123	-0.067	-0.025	-0.005	0	-0.008
C23-T4-S	rot. istant. kk	0.002	0.006	0.006	0.006	-0.005	0.005	-0.001	-0.015	-0.038	-0.071
	MR7 kk	-0.005	0.004	0.005	0.004	-0.011	0.004	-0.003	-0.021	-0.05	-0.089
	MR9 kk	-0.009	0.001	0.002	0.002	-0.014	0.002	-0.005	-0.023	-0.052	-0.092
	MR11 kk	-0.014	-0.002	-0.001	-0.001	-0.018	-0.001	-0.007	-0.025	-0.055	-0.095
C23-T5-N	rot. istant. kk	-0.083	-0.060	-0.055	-0.050	-0.084	-0.046	-0.016	-0.001	0.004	-0.001
	MR7 kk	-0.111	-0.082	-0.075	-0.069	-0.112	-0.063	-0.024	-0.004	0.002	-0.004
	MR9 kk	-0.113	-0.084	-0.078	-0.071	-0.115	-0.065	-0.027	-0.007	0	-0.006
	MR11 kk	-0.115	-0.086	-0.080	-0.074	-0.117	-0.068	-0.029	-0.009	-0.002	-0.009
C23-T5-S	rot. istant. kk	0.002	0.006	0.006	0.005	-0.005	0.004	-0.007	-0.026	-0.053	-0.086
	MR7 kk	-0.005	0.003	0.003	0.002	-0.010	0.001	-0.010	-0.032	-0.062	-0.1
	MR9 kk	-0.009	0	0	0	-0.013	-0.001	-0.012	-0.033	-0.063	-0.102
	MR11 kk	-0.013	-0.004	-0.003	-0.003	-0.017	-0.004	-0.014	-0.035	-0.065	-0.104

Tabella 12: Coefficienti di correlazione tra rotazioni doppiamente corrette (istantanee e mediate) e temperatura (istantanea, mediata e mediata con ritardo)

CODICE SENSORE	TEMPER. / ROTAZ.	MT6_r5 (da -7h a -5h)	MT6_r6 (da -8h a 6h)	MT6_r7 (da -9h a 7h)	MT6_r8 (da -10h a -8h)	MT6_r9 (da -11h a -9h)	MT6_r10 (da -12h a 10h)	MT6_r11 (da -13h a -11h)	MT6_r12 (da -14h a -12h)	MT6_r13 (da -15h a -13h)	MT6_r14 (da -16h a -14h)
C23-T2-N	rot. istant. kk	-0.005	-0.026	-0.053	-0.086	-0.121	-0.157	-0.189	-0.219	-0.246	-0.267
	MR7 kk	-0.011	-0.038	-0.076	-0.120	-0.169	-0.217	-0.263	-0.305	-0.342	-0.371
	MR9 kk	-0.013	-0.040	-0.078	-0.123	-0.172	-0.221	-0.268	-0.310	-0.347	-0.376
	MR11 kk	-0.016	-0.043	-0.081	-0.126	-0.175	-0.224	-0.271	-0.314	-0.351	-0.380
C23-T2-S	rot. istant. kk	-0.016	-0.040	-0.073	-0.112	-0.152	-0.188	-0.220	-0.249	-0.273	-0.291
	MR7 kk	-0.023	-0.055	-0.098	-0.149	-0.200	-0.247	-0.290	-0.328	-0.360	-0.383
	MR9 kk	-0.025	-0.058	-0.101	-0.151	-0.202	-0.251	-0.294	-0.333	-0.364	-0.388
	MR11 kk	-0.028	-0.061	-0.104	-0.153	-0.204	-0.253	-0.297	-0.336	-0.367	-0.391
C23-T3-N	rot. istant. kk	0.003	-0.002	-0.013	-0.027	-0.046	-0.069	-0.092	-0.116	-0.137	-0.154
	MR7 kk	0.003	-0.006	-0.023	-0.047	-0.078	-0.115	-0.154	-0.192	-0.227	-0.256
	MR9 kk	0.001	-0.007	-0.025	-0.050	-0.082	-0.120	-0.160	-0.199	-0.235	-0.266
	MR11 kk	-0.001	-0.009	-0.027	-0.053	-0.085	-0.124	-0.164	-0.204	-0.241	-0.272
C23-T3-S	rot. istant. kk	-0.001	-0.016	-0.038	-0.064	-0.095	-0.127	-0.157	-0.184	-0.208	-0.227
	MR7 kk	-0.004	-0.025	-0.054	-0.091	-0.133	-0.177	-0.219	-0.257	-0.290	-0.315
	MR9 kk	-0.006	-0.026	-0.056	-0.094	-0.136	-0.180	-0.223	-0.261	-0.295	-0.320
	MR11 kk	-0.009	-0.029	-0.059	-0.096	-0.139	-0.183	-0.225	-0.264	-0.297	-0.323
C23-T4-N	rot. istant. kk	-0.017	-0.045	-0.083	-0.125	-0.170	-0.214	-0.256	-0.291	-0.317	-0.334
	MR7 kk	-0.026	-0.063	-0.110	-0.165	-0.222	-0.280	-0.334	-0.379	-0.413	-0.436
	MR9 kk	-0.029	-0.065	-0.113	-0.168	-0.226	-0.284	-0.338	-0.383	-0.418	-0.442
	MR11 kk	-0.032	-0.069	-0.116	-0.171	-0.229	-0.287	-0.341	-0.387	-0.422	-0.446
C23-T4-S	rot. istant. kk	-0.112	-0.159	-0.211	-0.262	-0.310	-0.350	-0.382	-0.405	-0.415	-0.414
	MR7 kk	-0.139	-0.196	-0.259	-0.321	-0.378	-0.427	-0.467	-0.494	-0.506	-0.506
	MR9 kk	-0.142	-0.199	-0.262	-0.324	-0.381	-0.431	-0.470	-0.497	-0.510	-0.510
	MR11 kk	-0.145	-0.202	-0.264	-0.326	-0.384	-0.434	-0.473	-0.500	-0.513	-0.513
C23-T5-N	rot. istant. kk	-0.015	-0.038	-0.068	-0.103	-0.140	-0.178	-0.212	-0.240	-0.262	-0.278
	MR7 kk	-0.023	-0.054	-0.093	-0.139	-0.188	-0.237	-0.282	-0.319	-0.349	-0.370
	MR9 kk	-0.026	-0.056	-0.096	-0.142	-0.192	-0.241	-0.286	-0.324	-0.354	-0.375
	MR11 kk	-0.028	-0.059	-0.098	-0.145	-0.194	-0.243	-0.289	-0.327	-0.357	-0.378
C23-T5-S	rot. istant. kk	-0.127	-0.173	-0.222	-0.269	-0.311	-0.344	-0.368	-0.382	-0.384	-0.377
	MR7 kk	-0.146	-0.198	-0.253	-0.306	-0.352	-0.390	-0.417	-0.432	-0.435	-0.427
	MR9 kk	-0.148	-0.200	-0.255	-0.307	-0.354	-0.391	-0.418	-0.434	-0.437	-0.430
	MR11 kk	-0.150	-0.202	-0.256	-0.308	-0.355	-0.392	-0.419	-0.435	-0.438	-0.431

Tabella 13: Coefficienti di correlazione tra rotazioni doppiamente corrette (istantanee e mediate) e temperatura (istantanea, mediata e mediata con ritardo)

CODICE SENSORE	TEMPER. / ROTAZ.	MT6_r15 (da -17h a -15h)	MT6_r16 (da -18h a -16h)	MT6_r17 (da -19h a -17h)	MT6_r18 (da -20h a -18h)	MT6_r19 (da -21h a -19h)	MT6_r20 (da -22h a -20h)	MT6_r21 (da -23h a -21h)	MT6_r22 (da -24h a -22h)	MT6_r23 (da -25h a -23h)
C23-T2-N	rot. istant. kk	-0.281	-0.286	-0.284	-0.272	-0.252	-0.225	-0.194	-0.162	-0.131
	MR7 kk	-0.389	-0.397	-0.393	-0.377	-0.349	-0.312	-0.270	-0.227	-0.184
	MR9 kk	-0.395	-0.403	-0.399	-0.382	-0.354	-0.318	-0.275	-0.231	-0.189
	MR11 kk	-0.399	-0.407	-0.402	-0.386	-0.358	-0.321	-0.279	-0.235	-0.192
C23-T2-S	rot. istant. kk	-0.301	-0.301	-0.293	-0.277	-0.253	-0.222	-0.188	-0.152	-0.119
	MR7 kk	-0.396	-0.396	-0.385	-0.364	-0.333	-0.293	-0.248	-0.202	-0.159
	MR9 kk	-0.400	-0.401	-0.390	-0.368	-0.337	-0.297	-0.252	-0.206	-0.163
	MR11 kk	-0.403	-0.404	-0.393	-0.371	-0.340	-0.300	-0.255	-0.209	-0.166
C23-T3-N	rot. istant. kk	-0.170	-0.180	-0.184	-0.182	-0.173	-0.158	-0.137	-0.111	-0.083
	MR7 kk	-0.281	-0.298	-0.305	-0.301	-0.286	-0.261	-0.226	-0.184	-0.139
	MR9 kk	-0.291	-0.308	-0.315	-0.311	-0.296	-0.270	-0.234	-0.191	-0.144
	MR11 kk	-0.298	-0.315	-0.322	-0.318	-0.302	-0.276	-0.239	-0.196	-0.148
C23-T3-S	rot. istant. kk	-0.238	-0.242	-0.239	-0.226	-0.206	-0.181	-0.154	-0.123	-0.090
	MR7 kk	-0.331	-0.337	-0.331	-0.314	-0.287	-0.253	-0.214	-0.172	-0.126
	MR9 kk	-0.336	-0.342	-0.336	-0.319	-0.292	-0.257	-0.218	-0.175	-0.129
	MR11 kk	-0.339	-0.345	-0.339	-0.321	-0.294	-0.260	-0.220	-0.177	-0.132
C23-T4-N	rot. istant. kk	-0.344	-0.344	-0.332	-0.313	-0.284	-0.248	-0.210	-0.169	-0.129
	MR7 kk	-0.448	-0.448	-0.433	-0.408	-0.371	-0.325	-0.275	-0.222	-0.171
	MR9 kk	-0.454	-0.453	-0.439	-0.413	-0.376	-0.330	-0.279	-0.226	-0.176
	MR11 kk	-0.458	-0.457	-0.443	-0.416	-0.379	-0.334	-0.282	-0.229	-0.180
C23-T4-S	rot. istant. kk	-0.404	-0.384	-0.357	-0.325	-0.287	-0.247	-0.208	-0.174	-0.150
	MR7 kk	-0.493	-0.469	-0.436	-0.397	-0.351	-0.303	-0.257	-0.216	-0.187
	MR9 kk	-0.497	-0.474	-0.441	-0.401	-0.355	-0.307	-0.261	-0.221	-0.191
	MR11 kk	-0.500	-0.477	-0.444	-0.404	-0.358	-0.310	-0.264	-0.225	-0.195
C23-T5-N	rot. istant. kk	-0.287	-0.286	-0.278	-0.262	-0.239	-0.207	-0.172	-0.136	-0.103
	MR7 kk	-0.381	-0.380	-0.370	-0.349	-0.317	-0.276	-0.230	-0.183	-0.139
	MR9 kk	-0.386	-0.386	-0.375	-0.353	-0.321	-0.280	-0.234	-0.187	-0.142
	MR11 kk	-0.389	-0.389	-0.378	-0.356	-0.324	-0.283	-0.237	-0.190	-0.146
C23-T5-S	rot. istant. kk	-0.361	-0.337	-0.306	-0.270	-0.231	-0.190	-0.152	-0.121	-0.099
	MR7 kk	-0.410	-0.383	-0.348	-0.307	-0.262	-0.217	-0.175	-0.140	-0.115
	MR9 kk	-0.412	-0.385	-0.350	-0.309	-0.265	-0.220	-0.178	-0.143	-0.117
	MR11 kk	-0.413	-0.386	-0.351	-0.311	-0.266	-0.222	-0.180	-0.146	-0.120

3.9. Coefficienti di correlazione tra sensori

Altre informazioni utili possono essere ricavate calcolando i fattori di correlazione tra le serie di rotazioni grezze, corrette e doppiamente corrette (istantanee e mediate) relative a un sensore e quelle relative a ogni altro sensore. La formula usata è quella presentata nel capitolo 3.2.1.

Il programma permette di calcolare la correlazione tra sensori suddivisi in diversi gruppi, e su un arco temporale definiti dall'utente.
Considerata la simmetria del viadotto e del posizionamento dei sensori, e considerato il verso scelto come positivo per le rotazioni, ha senso effettuare un calcolo dei valori di correlazione in un gruppo formato da tutti i sensori installati; l'arco temporale scelto è l'intero periodo in cui sono state prese le misurazioni, ma si possono effettuare analisi più specifiche.

Per ciascun tipo di rotazioni (rotazioni istantanee o mediate su 7, 9 o 11 valori, calcolate per i dataset grezzo, corretto o doppiamente corretto), si riportano i coefficienti di correlazione tra le serie relative a due diversi sensori in una matrice ovviamente simmetrica, in cui la diagonale principale è vuota, in quanto rappresenta il coefficiente di correlazione tra una serie e sé stessa.

Inoltre, per ogni tipo di rotazione, vengono calcolate le medie dei coefficienti relativi a ciascun sensore, la media della famiglia e lo scarto tra le due quantità descritte sopra, calcolato come percentuale:

$$scarto = \frac{media\ sensore - media\ tot}{media\ tot} \cdot 100$$

I risultati sono mostrati nelle *Tabelle 14, 15* e *16*.

Tabella 14: Coefficienti di correlazione tra rotazioni raw

ROTAZIONI ISTANTANEE	C23-T2-N	C23-T2-S	C23-T3-N	C23-T3-S	C23-T4-N	C23-T4-S	C23-T5-N	C23-T5-S
C23-T2-N		0.694	-0.029	0.665	0.685	0.560	0.618	0.669
C23-T2-S	0.694		0.026	0.750	0.606	0.644	0.571	0.617
C23-T3-N	-0.029	0.026		-0.082	0.125	-0.025	-0.008	-0.319
C23-T3-S	0.665	0.750	-0.082		0.621	0.631	0.640	0.745
C23-T4-N	0.685	0.606	0.125	0.621		0.662	0.770	0.626
C23-T4-S	0.560	0.644	-0.025	0.631	0.662		0.585	0.700
C23-T5-N	0.618	0.571	-0.008	0.640	0.770	0.585		0.730
C23-T5-S	0.669	0.617	-0.319	0.745	0.626	0.700	0.730	

MEDIA DEL SENSORE	0.552	0.558	-0.045	0.567	0.585	0.537	0.558	0.538
SCARTO [%]	14.6	16	-109.3	17.8	21.5	11.5	16	11.8
MEDIA TOTALE	0.481							

ROT. MEDIE 7 VALORI	C23-T2-N	C23-T2-S	C23-T3-N	C23-T3-S	C23-T4-N	C23-T4-S	C23-T5-N	C23-T5-S
C23-T2-N		0.904	-0.155	0.897	0.881	0.797	0.815	0.840
C23-T2-S	0.904		0.029	0.901	0.879	0.822	0.802	0.758
C23-T3-N	-0.155	0.029		-0.149	0.06	-0.053	-0.092	-0.411
C23-T3-S	0.897	0.901	-0.149		0.843	0.752	0.868	0.858
C23-T4-N	0.881	0.879	0.06	0.843		0.872	0.889	0.770
C23-T4-S	0.797	0.822	-0.053	0.752	0.872		0.781	0.784
C23-T5-N	0.815	0.802	-0.092	0.868	0.889	0.781		0.854
C23-T5-S	0.840	0.758	-0.411	0.858	0.770	0.784	0.854	

MEDIA DEL SENSORE	0.711	0.728	-0.110	0.710	0.742	0.679	0.702	0.636
SCARTO [%]	18.6	21.3	-118.3	18.4	23.7	13.2	17.1	6.1
MEDIA TOTALE	0.6							

ROT. MEDIE 9 VALORI	C23-T2-N	C23-T2-S	C23-T3-N	C23-T3-S	C23-T4-N	C23-T4-S	C23-T5-N	C23-T5-S
C23-T2-N		0.914	-0.164	0.908	0.891	0.810	0.825	0.849
C23-T2-S	0.914		0.028	0.910	0.893	0.833	0.814	0.765
C23-T3-N	-0.164	0.028		-0.154	0.053	-0.058	-0.098	-0.418
C23-T3-S	0.908	0.910	-0.154		0.856	0.761	0.879	0.864
C23-T4-N	0.891	0.893	0.053	0.856		0.885	0.897	0.779
C23-T4-S	0.810	0.833	-0.058	0.761	0.885		0.793	0.790
C23-T5-N	0.825	0.814	-0.098	0.879	0.897	0.793		0.861
C23-T5-S	0.849	0.765	-0.418	0.864	0.779	0.790	0.861	

MEDIA DEL SENSORE	0.719	0.737	-0.116	0.718	0.751	0.688	0.710	0.641
SCARTO [%]	18.7	21.6	-119.1	18.4	23.9	13.5	17.2	5.9
MEDIA TOTALE	0.606							

ROT. MEDIE 11 VALORI	C23-T2-N	C23-T2-S	C23-T3-N	C23-T3-S	C23-T4-N	C23-T4-S	C23-T5-N	C23-T5-S
C23-T2-N		0.921	-0.172	0.915	0.898	0.819	0.831	0.854
C23-T2-S	0.921		0.026	0.915	0.903	0.841	0.821	0.770
C23-T3-N	-0.172	0.026		-0.159	0.047	-0.062	-0.104	-0.423
C23-T3-S	0.915	0.915	-0.159		0.864	0.767	0.886	0.867
C23-T4-N	0.898	0.903	0.047	0.864		0.894	0.901	0.785
C23-T4-S	0.819	0.841	-0.062	0.767	0.894		0.801	0.795
C23-T5-N	0.831	0.821	-0.104	0.886	0.901	0.801		0.865
C23-T5-S	0.854	0.770	-0.423	0.867	0.785	0.795	0.865	

MEDIA DEL SENSORE	0.724	0.743	-0.121	0.722	0.756	0.693	0.714	0.645
SCARTO [%]	18.8	21.8	-119.8	18.5	24	13.8	17.2	5.8
MEDIA TOTALE	0.61							

Tabella 15: Coefficienti di correlazione tra rotazioni k

ROTAZIONI ISTANTANEE	C23-T2-N	C23-T2-S	C23-T3-N	C23-T3-S	C23-T4-N	C23-T4-S	C23-T5-N	C23-T5-S
C23-T2-N		0.557	0.437	0.397	0.549	0.380	0.368	0.251
C23-T2-S	0.557		0.350	0.643	0.477	0.539	0.390	0.406
C23-T3-N	0.437	0.350		0.370	0.469	0.240	0.407	0.192
C23-T3-S	0.397	0.643	0.370		0.454	0.501	0.398	0.414
C23-T4-N	0.549	0.477	0.469	0.454		0.561	0.686	0.459
C23-T4-S	0.380	0.539	0.240	0.501	0.561		0.429	0.701
C23-T5-N	0.368	0.390	0.407	0.398	0.686	0.429		0.506
C23-T5-S	0.251	0.406	0.192	0.414	0.459	0.701	0.506	

MEDIA DEL SENSORE	0.420	0.480	0.352	0.454	0.522	0.479	0.455	0.418
SCARTO [%]	-6.2	7.3	-21.3	1.4	16.6	7	1.7	-6.5
MEDIA TOTALE	0.448							

ROT. MEDIE 7 VALORI	C23-T2-N	C23-T2-S	C23-T3-N	C23-T3-S	C23-T4-N	C23-T4-S	C23-T5-N	C23-T5-S
C23-T2-N		0.862	0.697	0.755	0.826	0.697	0.605	0.524
C23-T2-S	0.862		0.685	0.867	0.813	0.739	0.667	0.586
C23-T3-N	0.697	0.685		0.737	0.706	0.439	0.672	0.424
C23-T3-S	0.755	0.867	0.737		0.776	0.649	0.727	0.554
C23-T4-N	0.826	0.813	0.706	0.776		0.808	0.832	0.669
C23-T4-S	0.697	0.739	0.439	0.649	0.808		0.658	0.815
C23-T5-N	0.605	0.667	0.672	0.727	0.832	0.658		0.683
C23-T5-S	0.524	0.586	0.424	0.554	0.669	0.815	0.683	

MEDIA DEL SENSORE	0.709	0.745	0.623	0.724	0.776	0.687	0.692	0.608
SCARTO [%]	2	7.2	-10.4	4.1	11.5	-1.3	-0.5	-12.6
MEDIA TOTALE	0.695							

ROT. MEDIE 9 VALORI	C23-T2-N	C23-T2-S	C23-T3-N	C23-T3-S	C23-T4-N	C23-T4-S	C23-T5-N	C23-T5-S
C23-T2-N		0.881	0.721	0.779	0.843	0.716	0.620	0.541
C23-T2-S	0.881		0.715	0.882	0.834	0.752	0.684	0.596
C23-T3-N	0.721	0.715		0.770	0.726	0.457	0.696	0.444
C23-T3-S	0.779	0.882	0.770		0.799	0.659	0.749	0.563
C23-T4-N	0.843	0.834	0.726	0.799		0.824	0.843	0.683
C23-T4-S	0.716	0.752	0.457	0.659	0.824		0.672	0.821
C23-T5-N	0.620	0.684	0.696	0.749	0.843	0.672		0.695
C23-T5-S	0.541	0.596	0.444	0.563	0.683	0.821	0.695	

MEDIA DEL SENSORE	0.729	0.764	0.647	0.743	0.793	0.700	0.709	0.621
SCARTO [%]	2.2	7.1	-9.3	4.2	11.2	-1.8	-0.6	-13
MEDIA TOTALE	0.713							

ROT. MEDIE 11 VALORI	C23-T2-N	C23-T2-S	C23-T3-N	C23-T3-S	C23-T4-N	C23-T4-S	C23-T5-N	C23-T5-S
C23-T2-N		0.894	0.737	0.795	0.855	0.729	0.630	0.552
C23-T2-S	0.894		0.737	0.893	0.848	0.76	0.695	0.603
C23-T3-N	0.737	0.737		0.795	0.740	0.470	0.712	0.459
C23-T3-S	0.795	0.893	0.795		0.814	0.665	0.763	0.568
C23-T4-N	0.855	0.848	0.740	0.814		0.835	0.849	0.692
C23-T4-S	0.729	0.760	0.470	0.665	0.835		0.681	0.824
C23-T5-N	0.630	0.695	0.712	0.763	0.849	0.681		0.702
C23-T5-S	0.552	0.603	0.459	0.568	0.692	0.824	0.702	

MEDIA DEL SENSORE	0.742	0.776	0.664	0.756	0.805	0.709	0.719	0.629
SCARTO [%]	2.3	7	-8.4	4.3	11	-2.2	-0.8	-13.3
MEDIA TOTALE	0.725							

Tabella 16: Coefficienti di correlazione tra rotazioni kk

ROTAZIONI ISTANTANEE	C23-T2-N	C23-T2-S	C23-T3-N	C23-T3-S	C23-T4-N	C23-T4-S	C23-T5-N	C23-T5-S
C23-T2-N		0.552	0.426	0.386	0.545	0.399	0.359	0.261
C23-T2-S	0.552		0.344	0.639	0.471	0.550	0.382	0.410
C23-T3-N	0.426	0.344		0.355	0.466	0.280	0.395	0.218
C23-T3-S	0.386	0.639	0.355		0.447	0.518	0.387	0.424
C23-T4-N	0.545	0.471	0.466	0.447		0.574	0.681	0.464
C23-T4-S	0.399	0.550	0.280	0.518	0.574		0.442	0.701
C23-T5-N	0.359	0.382	0.395	0.387	0.681	0.442		0.513
C23-T5-S	0.261	0.410	0.218	0.424	0.464	0.701	0.513	

MEDIA DEL SENSORE	0.418	0.478	0.355	0.451	0.521	0.495	0.451	0.427
SCARTO [%]	-6.9	6.4	-21	0.3	15.9	10	0.4	-5
MEDIA TOTALE	0.450							

ROT. MEDIE 7 VALORI	C23-T2-N	C23-T2-S	C23-T3-N	C23-T3-S	C23-T4-N	C23-T4-S	C23-T5-N	C23-T5-S
C23-T2-N		0.860	0.699	0.747	0.826	0.733	0.595	0.543
C23-T2-S	0.860		0.702	0.865	0.810	0.758	0.660	0.594
C23-T3-N	0.699	0.702		0.738	0.724	0.535	0.674	0.491
C23-T3-S	0.747	0.865	0.738		0.771	0.681	0.718	0.572
C23-T4-N	0.826	0.810	0.724	0.771		0.830	0.828	0.678
C23-T4-S	0.733	0.758	0.535	0.681	0.830		0.684	0.815
C23-T5-N	0.595	0.660	0.674	0.718	0.828	0.684		0.697
C23-T5-S	0.543	0.594	0.491	0.572	0.678	0.815	0.697	

MEDIA DEL SENSORE	0.715	0.750	0.652	0.727	0.781	0.719	0.694	0.627
SCARTO [%]	0.9	5.9	-7.9	2.7	10.3	1.6	-2	-11.4
MEDIA TOTALE	0.708							

ROT. MEDIE 9 VALORI	C23-T2-N	C23-T2-S	C23-T3-N	C23-T3-S	C23-T4-N	C23-T4-S	C23-T5-N	C23-T5-S
C23-T2-N		0.880	0.726	0.771	0.843	0.753	0.611	0.561
C23-T2-S	0.880		0.737	0.881	0.831	0.772	0.678	0.605
C23-T3-N	0.726	0.737		0.776	0.749	0.560	0.701	0.517
C23-T3-S	0.771	0.881	0.776		0.795	0.692	0.741	0.582
C23-T4-N	0.843	0.831	0.749	0.795		0.847	0.839	0.693
C23-T4-S	0.753	0.772	0.560	0.692	0.847		0.699	0.821
C23-T5-N	0.611	0.678	0.701	0.741	0.839	0.699		0.710
C23-T5-S	0.561	0.605	0.517	0.582	0.693	0.821	0.710	

MEDIA DEL SENSORE	0.735	0.769	0.681	0.748	0.800	0.735	0.711	0.641
SCARTO [%]	1	5.7	-6.4	2.8	9.9	1	-2.2	-11.8
MEDIA TOTALE	0.728							

ROT. MEDIE 11 VALORI	C23-T2-N	C23-T2-S	C23-T3-N	C23-T3-S	C23-T4-N	C23-T4-S	C23-T5-N	C23-T5-S
C23-T2-N		0.893	0.745	0.788	0.856	0.767	0.621	0.573
C23-T2-S	0.893		0.763	0.892	0.846	0.781	0.688	0.612
C23-T3-N	0.745	0.763		0.803	0.766	0.579	0.72	0.536
C23-T3-S	0.788	0.892	0.803		0.810	0.698	0.755	0.587
C23-T4-N	0.856	0.846	0.766	0.810		0.859	0.845	0.702
C23-T4-S	0.767	0.781	0.579	0.698	0.859		0.708	0.824
C23-T5-N	0.621	0.688	0.72	0.755	0.845	0.708		0.717
C23-T5-S	0.573	0.612	0.536	0.587	0.702	0.824	0.717	

MEDIA DEL SENSORE	0.749	0.782	0.702	0.762	0.812	0.745	0.722	0.650
SCARTO [%]	1.1	5.6	-5.2	2.9	9.6	0.6	-2.5	-12.2
MEDIA TOTALE	0.741							

3.10. Analisi nel dominio della frequenza

3.10.1. Trasformata di Fourier

Le serie di rotazioni grezze ('raw'), corrette ('k') e doppiamente corrette('kk') possono essere anche analizzate con gli approcci della teoria dei segnali, in quanto un segnale s(t) è una qualsiasi funzione reale o complessa del tempo che sia fisicamente realizzabile.
Oltre alla descrizione nel dominio del tempo, come visto finora, si può rappresentare un segnale anche nel dominio della frequenza. Un segnale, infatti, può essere scomposto nella somma di segnali armonici di ampiezza infinitesima, secondo la formula riportata:

$$s(t) = \int_{-\infty}^{+\infty} S(f) \cdot e^{j2\pi ft}\, df$$

avendo definito:

$$S(f) = \int_{-\infty}^{+\infty} s(t) \cdot e^{-j2\pi ft}\, dt$$

come trasformata di Fourier del segnale s(t), ovvero spettro del segnale s(t).
Ora, si esegue un'analisi di Fourier dei segnali rappresentati dalle serie di rotazioni (grezze, corrette o doppiamente corrette).
Innanzitutto, si deve considerare che questi sono segnali campionati, indicati con s_δ, ovvero hanno un certo periodo di campionamento (inverso della frequenza di campionamento):

$$T_c = 20\ min = 20 \cdot 60\ sec = 1200\ sec$$

$$f_c = \frac{1}{T_c} = \frac{1}{1200} = 0.00083\ Hz$$

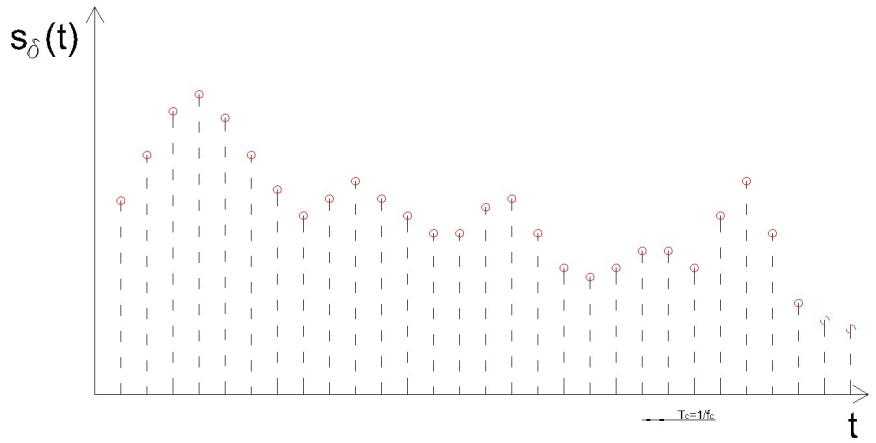


Figura 20: Generico segnale campionato

Segnali campionati possono essere visti come il prodotto del segnale reale, continuo nel tempo, e un treno di impulsi:

$$s_\delta(t) = \sum_{n=-\infty}^{\infty} T_c \cdot s(nT_c) \cdot \delta(t - nT_c) = T_c \cdot s(t) \cdot \sum_{n=-\infty}^{\infty} \delta(t - nT_c)$$

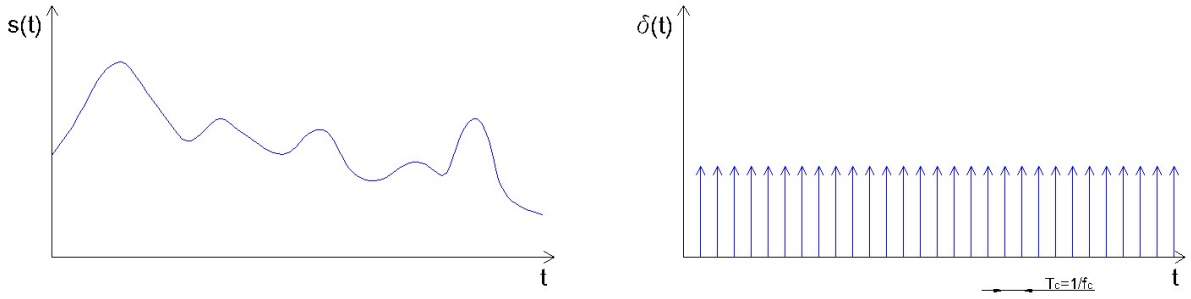


Figura 21: Segnale continuo e treno di impulsi

Trasformando secondo Fourier, utilizzando le proprietà dell'operatore trasformata di Fourier:

$$S_\delta(f) = T_c \cdot S(f) \cdot \frac{1}{T_c} \cdot \sum_{n=-\infty}^{\infty} \delta(f - nf_c) = \sum_{n=-\infty}^{\infty} S(f - nf_c)$$

Dove $S_\delta(f)$ è la trasformata di Fourier del segnale campionato; la relazione precedente mostra che la trasformata di Fourier di un segnale campionato fornisce una sequenza di trasformate del segnale originario continuo ($S(f)$).

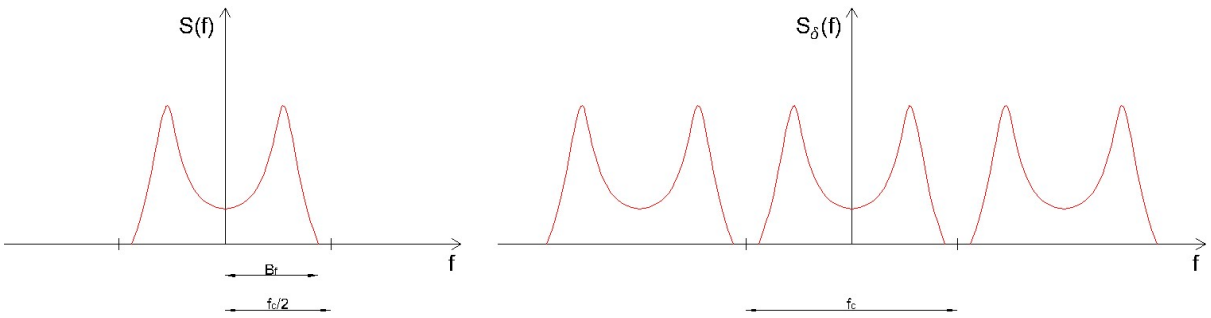


Figura 22: Trasformate di Fourier del segnale continuo e del corrispettivo campionato

La frequenza con cui queste trasformate si ripetono è pari alla frequenza di campionamento f_c, mentre la massima frequenza dello spettro del segnale continuo viene detta banda utile, ed è indicata con B_f. Si può facilmente notare che se la frequenza di campionamento è troppo piccola, si verifica la sovrapposizione delle repliche e la conseguente perdita di informazione (''aliasing''); per scongiurare ciò, si deve verificare il criterio di Nyquist:

$$f_c = \frac{1}{T_c} > 2B_f$$

La valutazione numerica della trasformata di Fourier è possibile grazie alla formulazione della DFT (Discrete Fourier Transform), che elimina i problemi legati all'utilizzo pratico delle formule viste (ovvero la presenza, nelle sommatorie, di infiniti termini, e la continuità delle variabili tempo e frequenza); questo tipo di formulazione ha, però, una complessità computazionale pari a N^2, dove N è il numero di campioni. Allora viene utilizzato l'algoritmo FFT (Fast Fourier Transform), che permette di passare ad una complessità computazionale pari a N logN.

Per ciascun sensore, vengono rappresentate le ampiezze delle trasformate di Fourier, in funzione delle frequenze positive, per le tre serie di dati ottenute (rotazioni grezze 'raw', corrette 'k' e doppiamente corrette 'kk'); l'ampiezza della trasformata relativa alla

frequenza f è proporzionale all'energia dell'armonica con tale frequenza nella composizione del segnale originale. Si riporta, a titolo esemplificativo, il grafico relativo al sensore posizionato sulla trave numero due, nel lato Nord della campata.

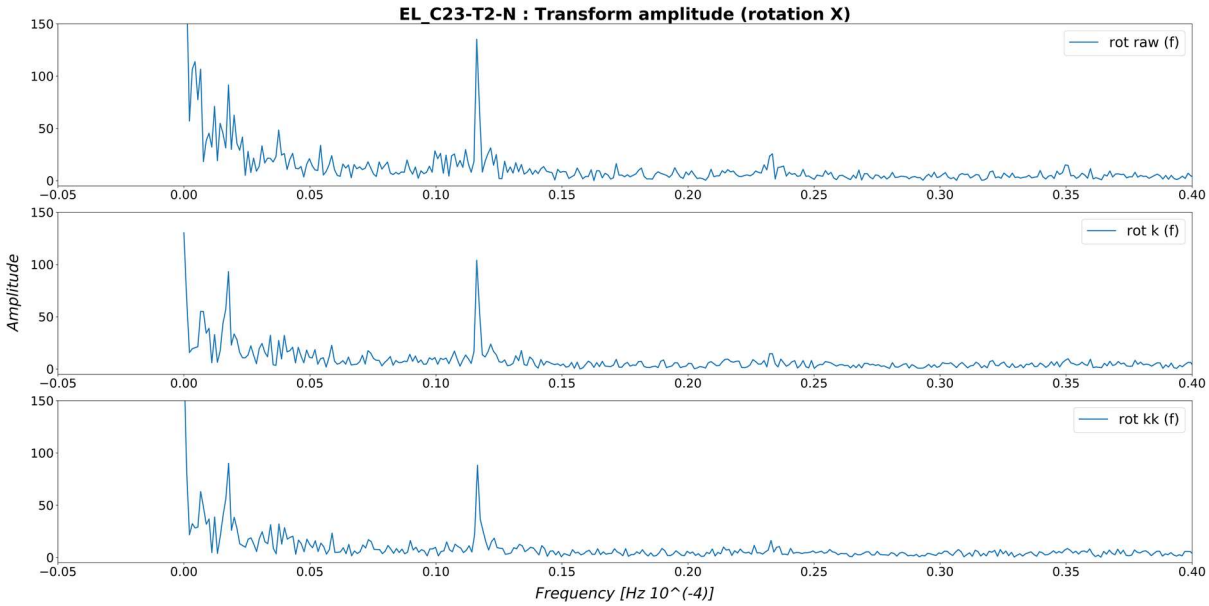


Figura 23: Ampiezze delle trasformate di Fourier delle rotazioni grezze ('raw'), corrette 'k' e doppiamente corrette 'kk'

Da questi grafici si possono ottenere informazioni qualitative riguardo le frequenze (e quindi i periodi) fondamentali dei data-set in esame. Innanzitutto, in tutti i grafici si distingue un picco in corrispondenza di una frequenza pari a $f \simeq 0.12 \cdot 10^{-4}\ Hz$; essendo che:

$$periodo = T = \frac{1}{f}$$

esso è relativo a un'armonica con periodo pari all'incirca a un giorno, dovuta dunque presumibilmente all'escursione termica giornaliera; infatti, si può notare come l'ampiezza relativa a tale frequenza vada diminuendo con le successive compensazioni del segnale.
Un altro elemento da osservare è la presenza di un picco, sempre più facilmente distinguibile man mano che si procede con le correzioni, collegato una frequenza $f \simeq 0.02 \cdot 10^{-4}\ Hz$, corrispondente a un'armonica con periodo pari all'incirca a 7 giorni; è possibile che questa descriva la variazione di traffico stradale dovuta ai week-end.

3.10.2. Densità Spettrale di Energia (ESD)

La Densità Spettrale di Energia di un segnale descrive la distribuzione della sua energia alle diverse frequenze; viene definita come:

$$E(f) = |S(f)|^2$$

Dove S(f) è la trasformata di Fourier del segnale s(t).

Per la relazione di Parseval, è valida la seguente equazione:

$$\int_{-\infty}^{\infty} \|S(f)\|^2 \, df = \int_{-\infty}^{\infty} \|s(t)\|^2 \, dt = E_s$$

Dove E_s è l'energia del segnale.

Per rotazioni grezze, corrette e doppiamente corrette sono stati calcolate anche le funzioni di Densità Spettrale di Energia; vengono riportati grafici relativi al sensore EL_C23-T2-N, come esempio.

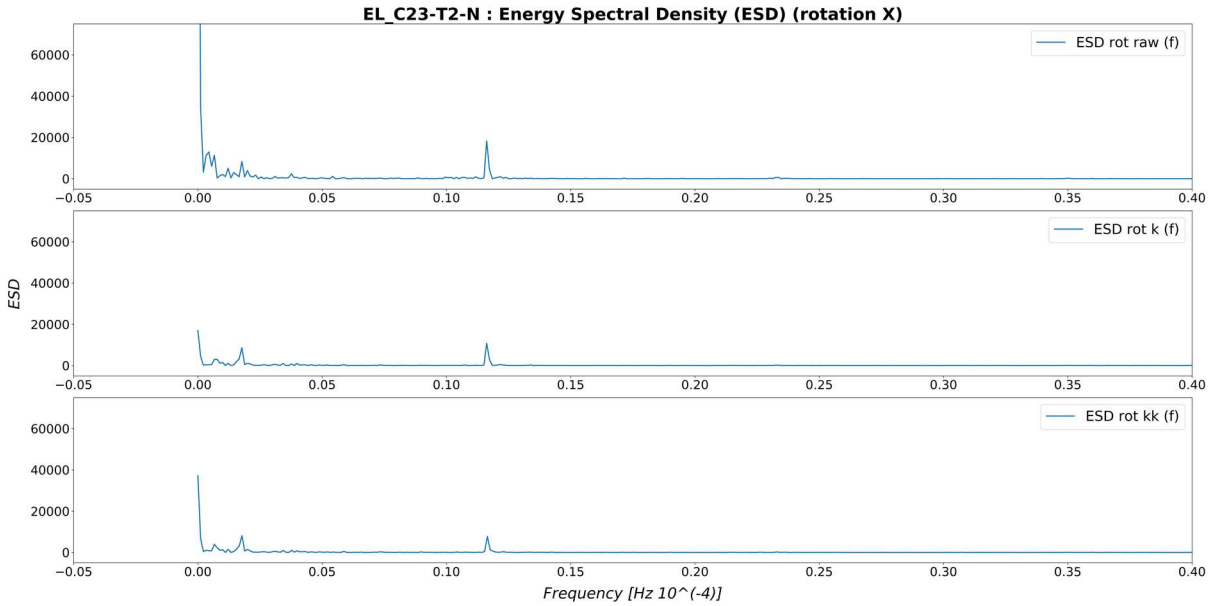


Figura 24: Ampiezze dell'Energy Spectral Density (ESD) delle rotazioni grezze 'raw', corrette 'k' e doppiamente corrette 'kk'

3.11. Grafici ottenuti

Si riportano, come esempio, i grafici relativi al sensore T2-N. Questi possono essere utili per confermare supposizioni ottenute mediante l'analisi, e per ottenere ulteriori informazioni qualitative.

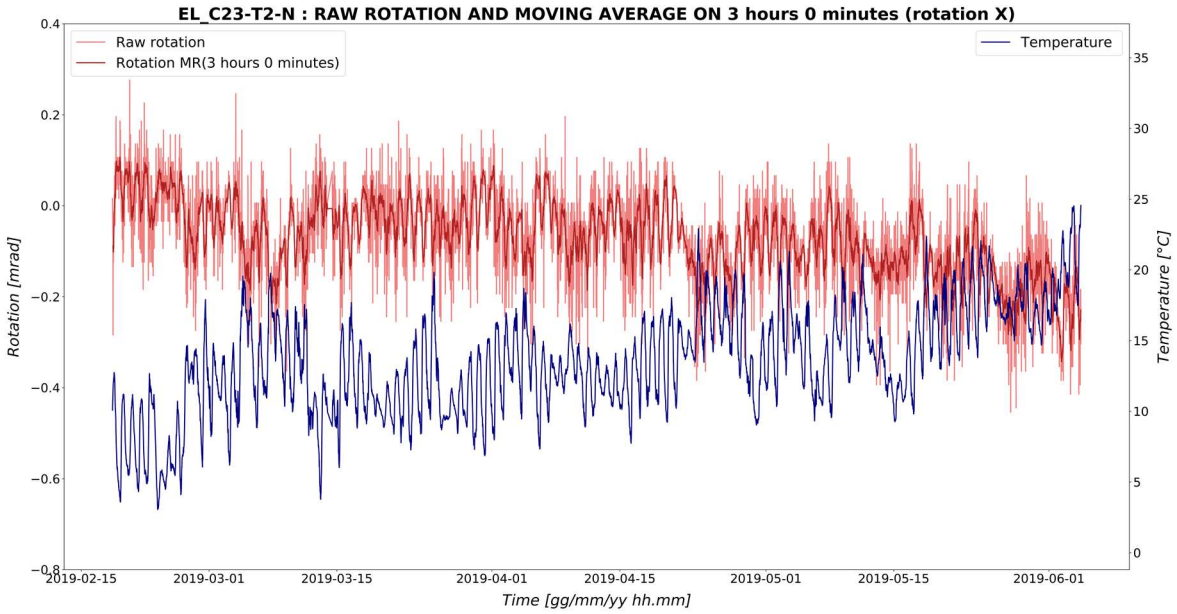


Figura 25: Rotazione raw e sua media mobile su 9 valori

Questo grafico rappresenta l'andamento delle rotazioni grezze e le medie mobili centrate su 9 valori di rotazione grezza (180 minuti), oltre alla temperatura. Si può notare come la media mobile tenda ad eliminare i picchi dovuti presumibilmente all'azione istantanea del traffico. Si osservi inoltre la tendenza dell'impalcato a inflettersi maggiormente (ovvero ad aumentare in valore assoluto il dato di rotazione) con l'aumento di temperatura stagionale.

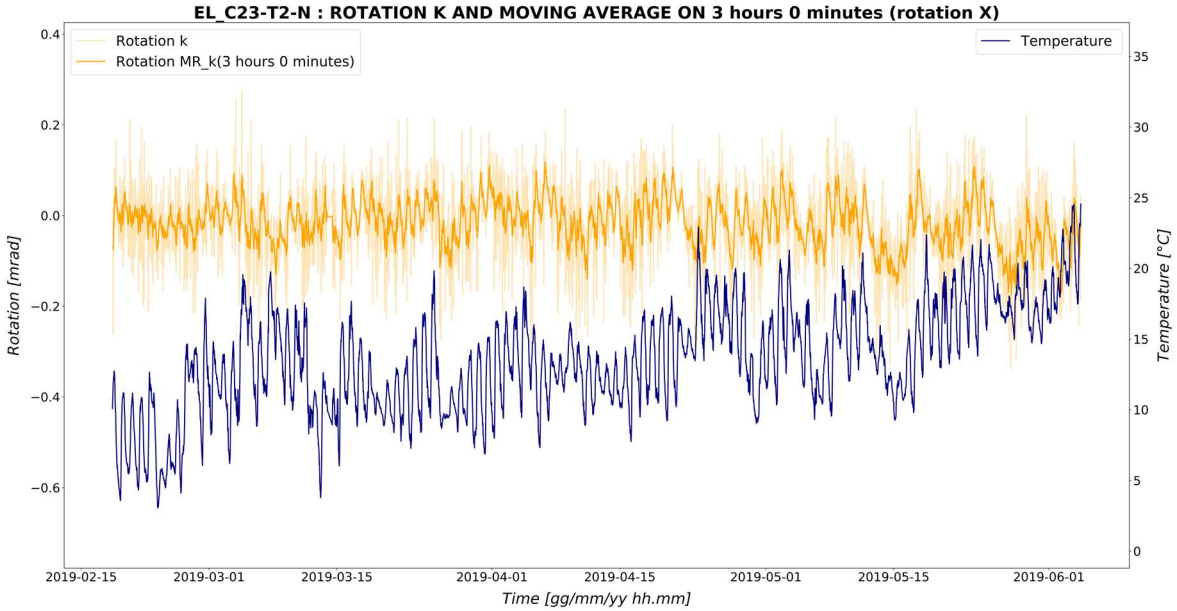


Figura 26: Rotazione k e sua media mobile su 9 valori

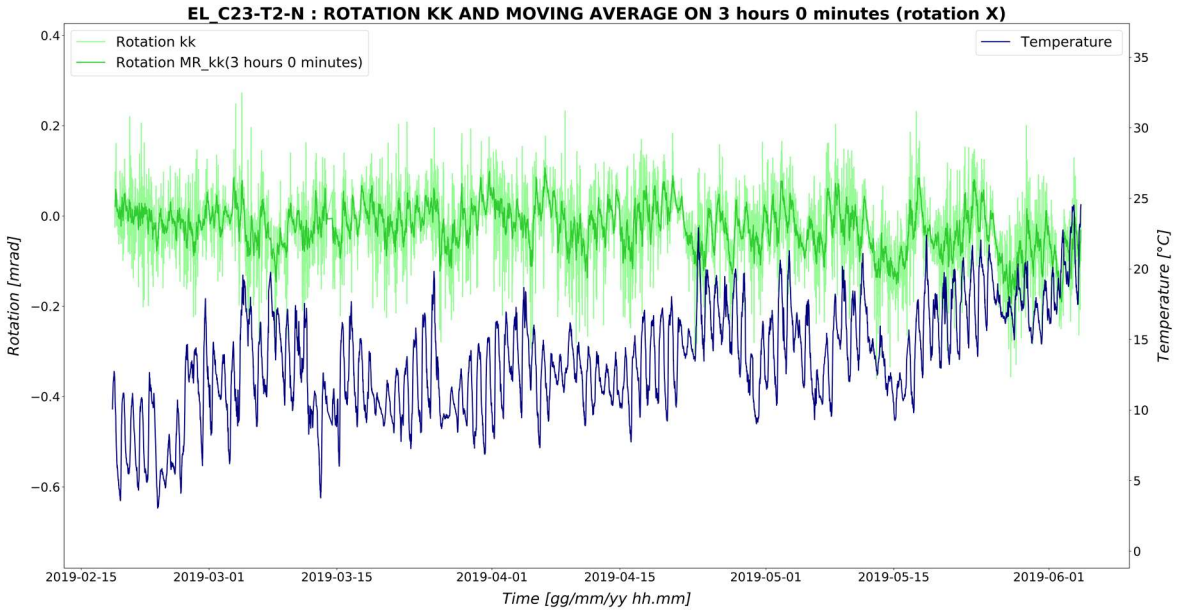


Figura 27: Rotazione kk e sua media mobile su 9 valori

Nel secondo e nel terzo grafico sono mostrate le stesse grandezze del primo, ma relative rispettivamente alle rotazioni corrette e doppiamente corrette. Si possono fare osservazioni analoghe a quelle già fatte per la prima immagine.

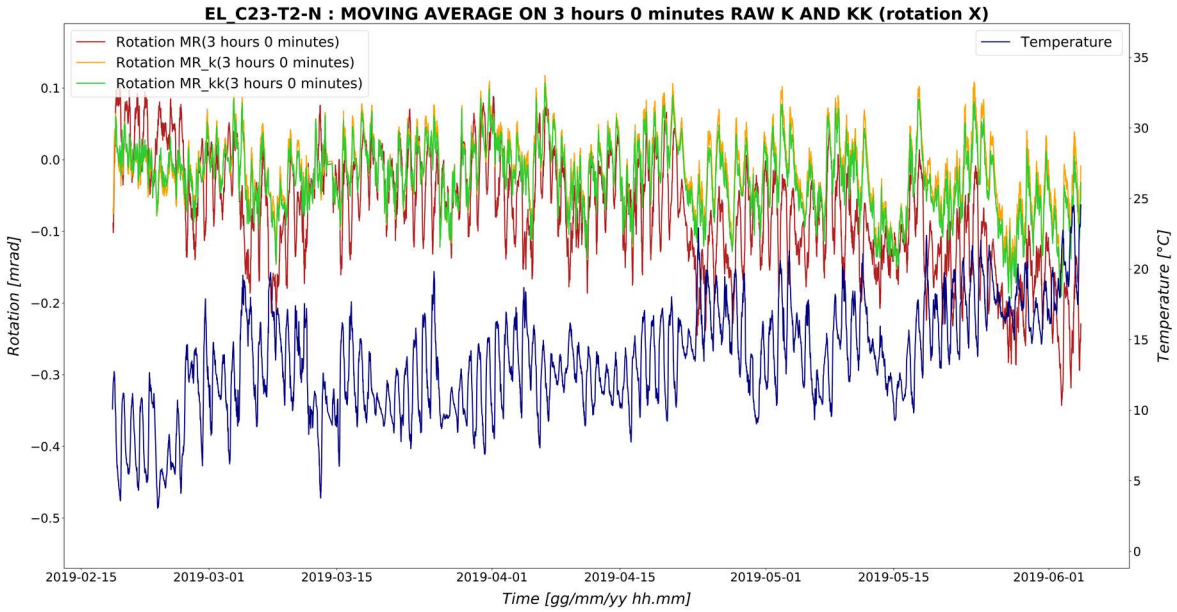


Figura 28: Media mobile su 9 valori della rotazione raw, k e kk

In questa immagine vengono messe a confronto le medie mobili su 9 valori di rotazione (180 minuti) relative a rotazioni grezze ('raw'), corrette ('k') e doppiamente corrette ('kk'). Si noti che i valori relativi alle serie corrette sono molto simili tra loro, e che essi presentano picchi di ampiezza minore. Inoltre, si può osservare come anche l'effetto stagionale della variazione di temperatura sia molto più lieve sulle serie corrette piuttosto che nella serie grezza: infatti, con l'aumento stagionale della temperatura, le rotazioni grezze tendono ad aumentare molto in valore assoluto, mentre quelle corrette sono molto più stabili. Nonostante ciò, non si può dire che le serie corrette siano completamente svincolate da effetti termici stagionali.

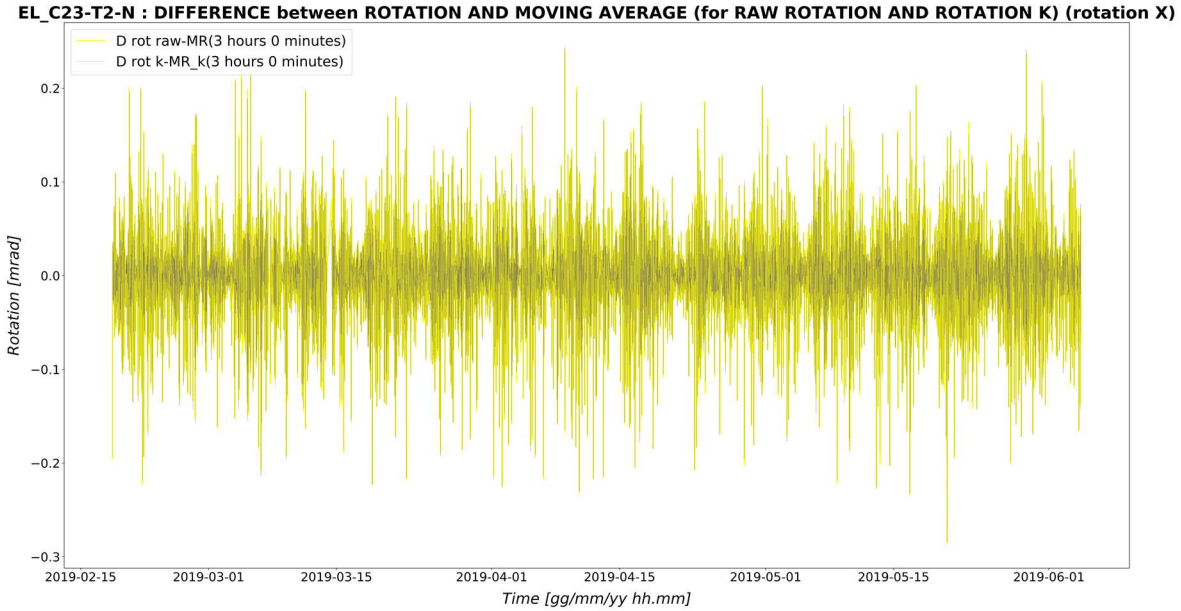


Figura 29: Differenza tra rotazione e sua media mobile su 9 valori, per rotazioni raw e k

Il quinto grafico è utile per osservare come la differenza tra la rotazione e la sua media mobile sia più piccola quando si tratta di rotazioni corrette, piuttosto che di rotazioni grezze. Ancora una volta, è una misura di come il dato corretto sia più stabile rispetto alle rotazioni grezze.

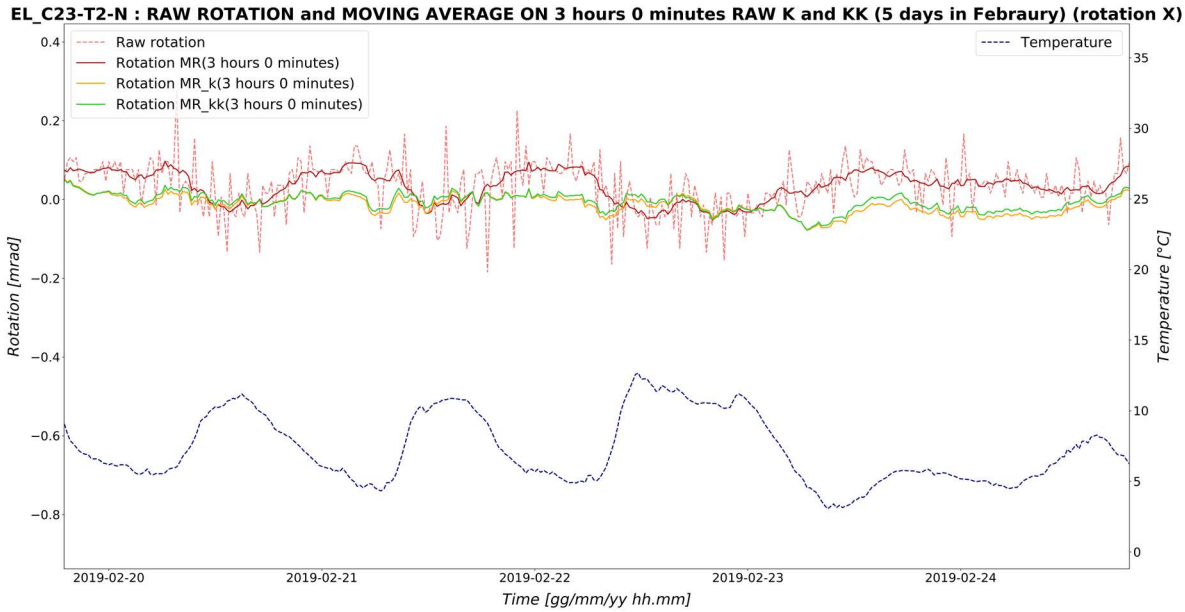


Figura 30: Rotazione raw e media mobile su 9 valori delle rotazioni raw, k e kk (19-24/02/2019)

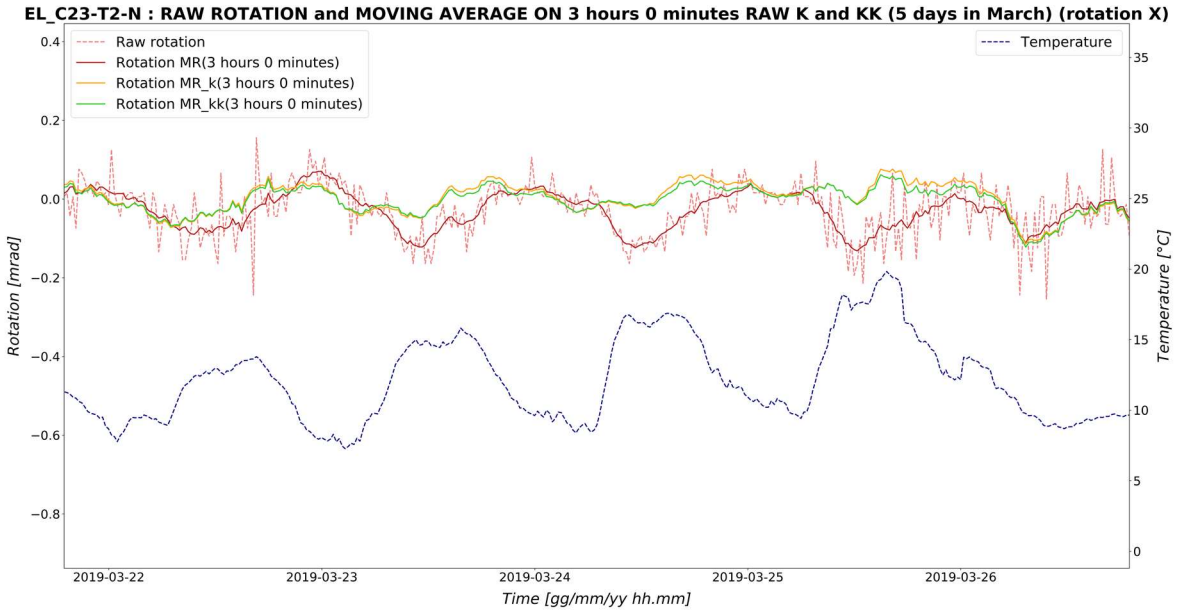


Figura 31: Rotazione raw e media mobile su 9 valori delle rotazioni raw, k e kk (21-26/03/2019)

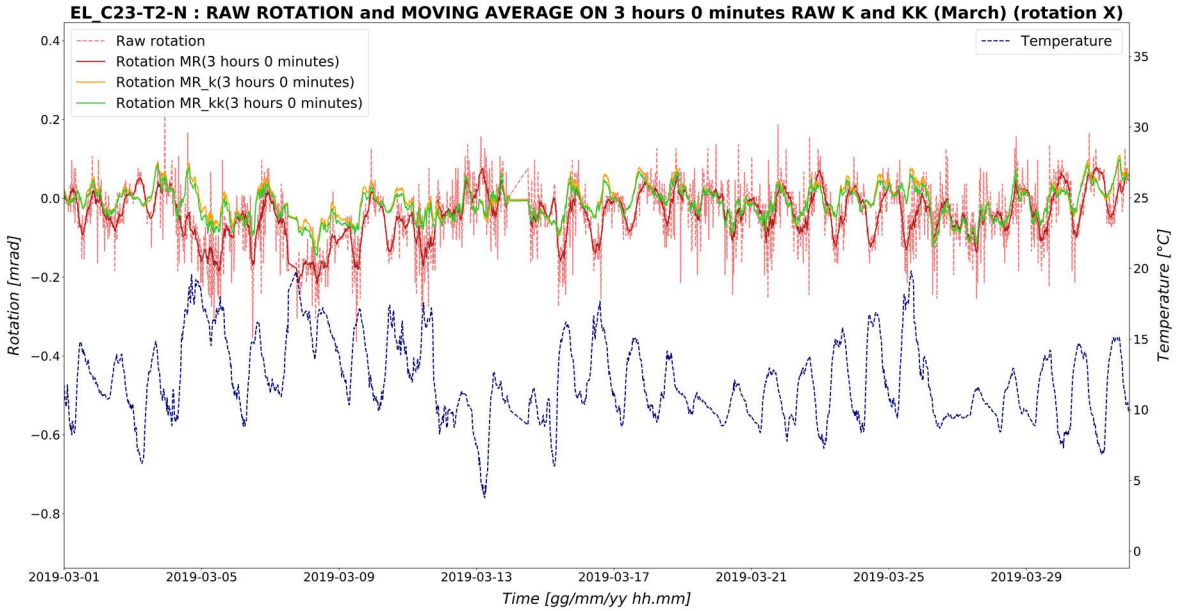


Figura 32: Rotazione raw e media mobile su 9 valori delle rotazioni raw, k e kk (01-31/03/2019)

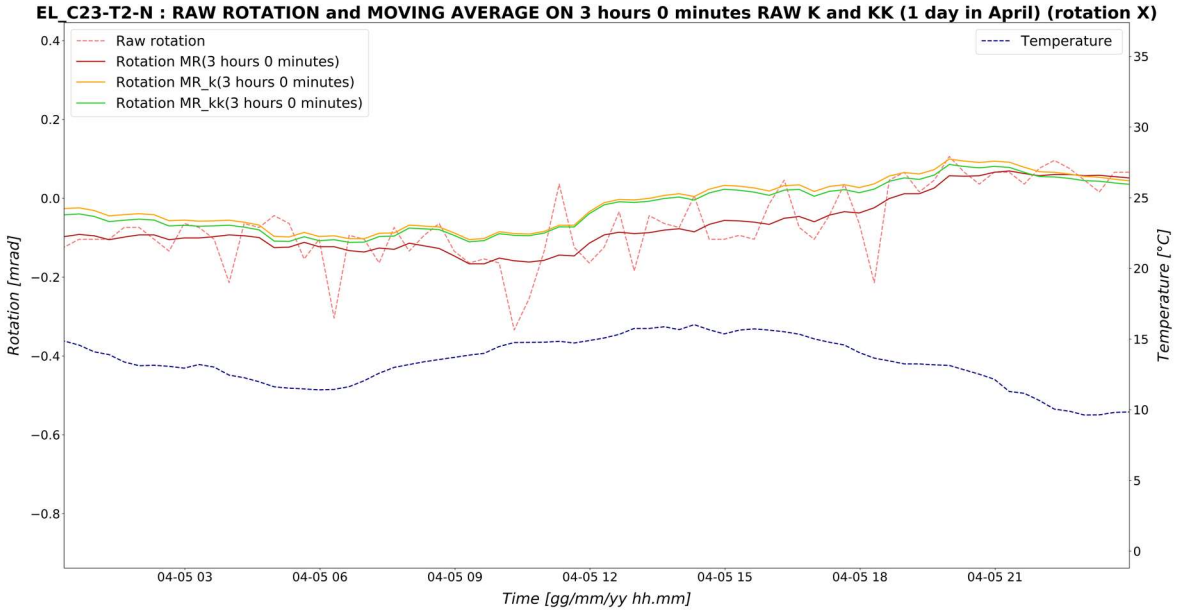


Figura 33: Rotazione raw e media mobile su 9 valori delle rotazioni raw, k e kk (05/04/2019)

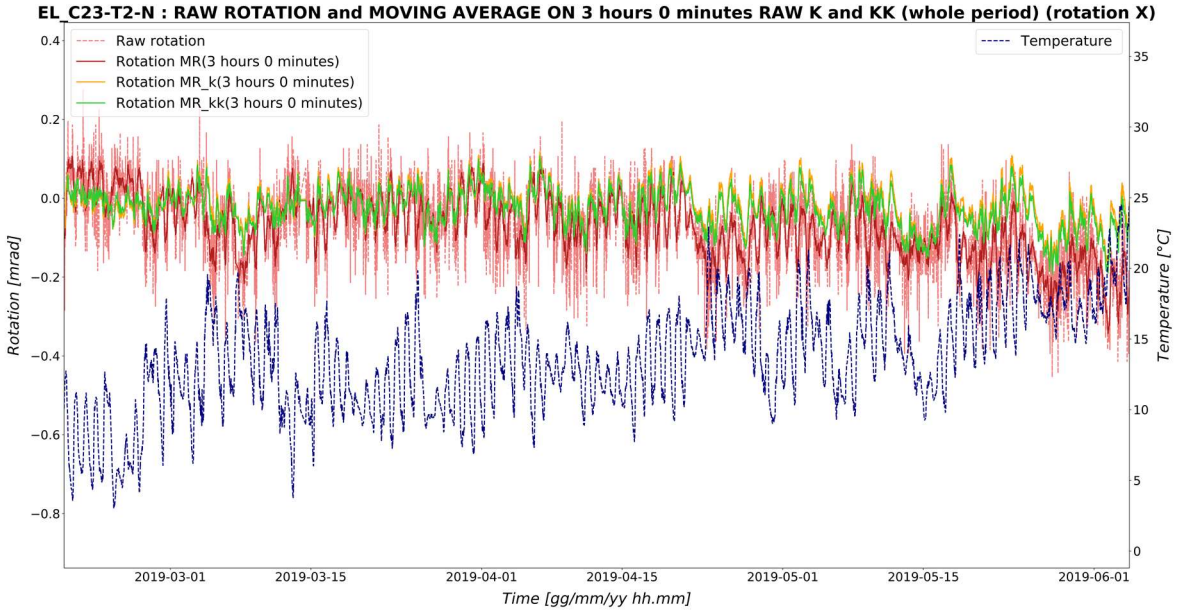


Figura 34: Rotazione raw e media mobile su 9 valori delle rotazioni raw, k e kk (18/02/2019-04/06/2019)

Questi ultimi 5 grafici mostrano, infine, un confronto tra rotazioni grezze e medie mobili centrate su 9 valori relativi a rotazioni grezze ('raw'), corrette ('k') e doppiamente corrette('kk') relativamente a diversi periodi temporali. In particolare, il grafico che rappresenta i dati di una giornata (15/04/2019) rende chiaro che una variazione di

temperatura dell'aria genera una rotazione dell'impalcato ritardata di circa 5 ore, come ricavato analiticamente.

4. ALGORITMO IN PYTHON

4.1. Introduzione

L'analisi è stata svolta mediante l'utilizzo di un algoritmo scritto in Python, suddiviso in due parti.

Nella prima parte, le serie di dati provenienti dai sensori vengono ripulite da eventuali anomalie macroscopiche, quali possono essere valori molto distanti dalla media (picchi), o periodi di tempo in cui il valore misurato per una determinata grandezza rimane invariato (plateau); entrambi questi fenomeni, presumibilmente, non derivano da un effettivo comportamento anormale della grandezza misurata, bensì essi sono causati da un malfunzionamento della strumentazione, e dunque vanno corretti in fase preliminare, prima di procedere all'analisi dei dati.

La seconda parte del codice tratta, invece, dell'analisi statistica delle serie di dati relative alle rotazioni (calcolo di valor medio della serie, scarto quadratico medio e medie mobili), del calcolo dei coefficienti di correlazione tra rotazioni e temperatura e tra rotazioni e umidità, qualora quest'ultima sia stata misurata dai sensori, e del calcolo delle derive utili per eliminare l'influenza della temperatura (o dell'umidità) sulle rotazioni.

Il programma è stato creato per poter analizzare dati provenienti da due tipi di sensore, definiti *"sensore A"* e *"sensore B"*.

Un sensore di tipo A misura rotazioni in due direzioni perpendicolari (rotazioni X e rotazioni Y), temperatura, e umidità. L'altro tipo di sensore, invece, valuta solo un tipo di rotazione (rotazione X), e la temperatura. Inoltre, ovviamente, entrambi posseggono un orologio interno che memorizza ora e data relative a una certa misurazione delle altre grandezze. I risultati esaminati nei precedenti capitoli sono relativi a sensori di tipo B.

4.2. Moduli aggiuntivi

Inizialmente, devono essere inseriti nel programma i moduli di Python che non sono compresi nella libreria standard, ma che sono necessari per l'esecuzione del codice; essi vengono importati e rinominati tramite il comando "import nome_modulo as nuovo_nome_modulo", come mostrato in *Figura 35*. Di alcuni di essi, sono necessarie solo

alcune specifiche funzioni, per cui vengono importate solo queste ultime tramite il comando "from nome_modulo import nome_funzione".

```
1.  import time
2.
3.  print("Orario di inizio:", time.ctime())    #Tempo di inizio esecuzione dell'intero algoritmo
4.
5.  from datetime import datetime
6.  from linecache import getline
7.  import math as m
8.  import numpy as np
9.  import pandas as pd
10. import matplotlib.pyplot as plt
11. import statistics as st
12. import os
13. import shutil
14. from pandas.plotting import register_matplotlib_converters
15. register_matplotlib_converters()
16. import openpyxl as xl
17. from openpyxl.styles import NamedStyle, Border, Side, Alignment
18.
19. stile2 = NamedStyle(name='stile2')
20. stile2.border = Border(left=Side(style='thin', color='FF000000'), right=Side(style='thin', color='FF000000')
    , top=Side(style='thin', color='FF000000'), bottom=Side(style='thin', color='FF000000'))
21. stile2.alignment = Alignment(horizontal='center')
```

Figura 35: Moduli importati

Questi moduli devono essere stati precedentemente installati sul calcolatore in cui viene eseguito il programma. Questi e tanti altri moduli e *package* aggiuntivi scritti in Python sono contenuti nel *Python Package Index* (PyPI), un repository da cui chiunque può scaricare pacchetti esistenti e in cui chiunque può condividerne di nuovi.
Si possono ottenere i *package* contenuti nel *Python Package Index* tramite un browser (all'indirizzo *https://pypi.org/*), oppure tramite un tool, installato assieme a Python per versioni successive rispetto alla 3.4 e 2.7.9, chiamato *pip* (Python Install Packages). Questo strumento permette di gestire i *package* di Python, in particolare esso consente la ricerca, il download e l'installazione di *package* che si trovano nel *Python Package Index*; inoltre esso permette di aggiornare o rimuovere quelli già installati.

Il tool *pip* viene richiamato dal Prompt dei comandi; per prima cosa si deve impostare come cartella di lavoro corrente la cartella in cui è contenuto *pip* con il comando "cd percorso_cartella_pip". Quindi, è possibile utilizzare il tool *pip*.
I principali comandi associati a quest'ultimo sono:

- pip search nome_package: comando utilizzato per cercare un package nel *Python Package Index*; oltre ai risultati della ricerca, esso indica quali package sono installati e se ne esista una versione più recente;
- pip install nome_package: comando utilizzato per installare un package dal *Python Package Index*; automaticamente viene scaricata la versione più recente;
- pip install -U nome_package: comando utilizzato per aggiornare un package già installato;
- pip list: comando utilizzato per elencare i package già installati;
- pip uninstall nome_package: comando utilizzato per disinstallare un package.

Per scaricare e installare i moduli necessari, è stato quindi usato il secondo comando.

4.3. Moduli aggiuntivi personalizzati

Considerata la complessità del codice, e considerato il fatto che alcune sue parti siano ricorsive, ovvero che alcune serie di operazioni vengano ripetute su diverse grandezze in punti differenti dell'algoritmo, alcuni blocchi di calcoli (ed in particolare tutti quelli contenenti operazioni ripetute) sono stati scritti all'interno di moduli esterni personalizzati, creati appositamente. In questa maniera il codice principale (o *main code*) risulta più snello, inoltre le operazioni (ricorsive e non) descritte nei codici esterni possono essere controllate e gestite più facilmente.

I moduli esterni vengono realizzati sotto forma di script scritti anch'essi in linguaggio Python, e suddivisi in base al macro-argomento che comprende le operazioni contenute. In ciascuno di essi vengono definite delle funzioni; queste sono uno strumento che permette di raggruppare un insieme di istruzioni che eseguono un compito specifico, accettando in input 0 o più argomenti, e restituendo in output un risultato, in genere.
La sintassi usata per definire una funzione è la seguente:

- Il comando usato per definire la funzione è def, seguito dal nome scelto per la funzione ("nome_funzione");

- dopo di questo, vengono specificati tra parentesi tonde i parametri (o argomenti) accettati dalla funzione (se non sono necessari parametri è sufficiente aprire e chiudere le parentesi), seguiti da due punti (:) che introducono un blocco di codice indentato;
- segue l'insieme di istruzioni da eseguire, contenente 0 o più return.

```
def nome_funzione (argomento1, argomento2, …) :
        blocco di istruzioni da eseguire
        return output1, output2, …
```

Queste funzioni sono richiamabili dal codice principale come fossero funzioni di un qualunque altro modulo installato sul calcolatore; dunque, innanzitutto il modulo va importato tramite il comando "import nome_modulo", poi si accede alla singola funzione con il comando "nome_modulo.nome_funzione (argomento1, argomento2, …).

I moduli esterni creati e le relative funzioni, che verranno analizzate in dettaglio nei paragrafi successivi, sono i seguenti:

- Excel: modulo contenente le istruzioni per creare il file Excel contenente i risultati e per definire le intestazioni delle tabelle; funzioni: crea_layout, crea;
- Plateau: modulo contenente le istruzioni per definire ed eliminare i plateaux dalle serie di dati; funzioni: trova_lim, pulisci_tipoA, pulisci_tipoB;
- Picchi: modulo contenente le istruzioni per definire ed eliminare i picchi dalle serie di dati; funzioni: elimina_picchiA, elimina_picchiB;
- Statistica: modulo contenente le operazioni statistiche effettuate sulle serie di dati, e le istruzioni per il calcolo delle derive; funzioni: calcoli_rotazioni, calcoli_temp_umidA, calcoli_temp_umidB, correlazione, deriva_istantanea, deriva_differita;
- Grafici: modulo contenente le istruzioni per tracciare i grafici; funzioni: plot_raw_vs_puliti, plot_no_derive, fourier, ritardi_plot.
- Cross_correlazione: modulo contenente le istruzioni per calcolare la cross-correlazione

tra diversi sensori e per creare il file Excel in cui salvare questi risultati; funzioni: crea_excel, salvataggio_dati, cross_corr.

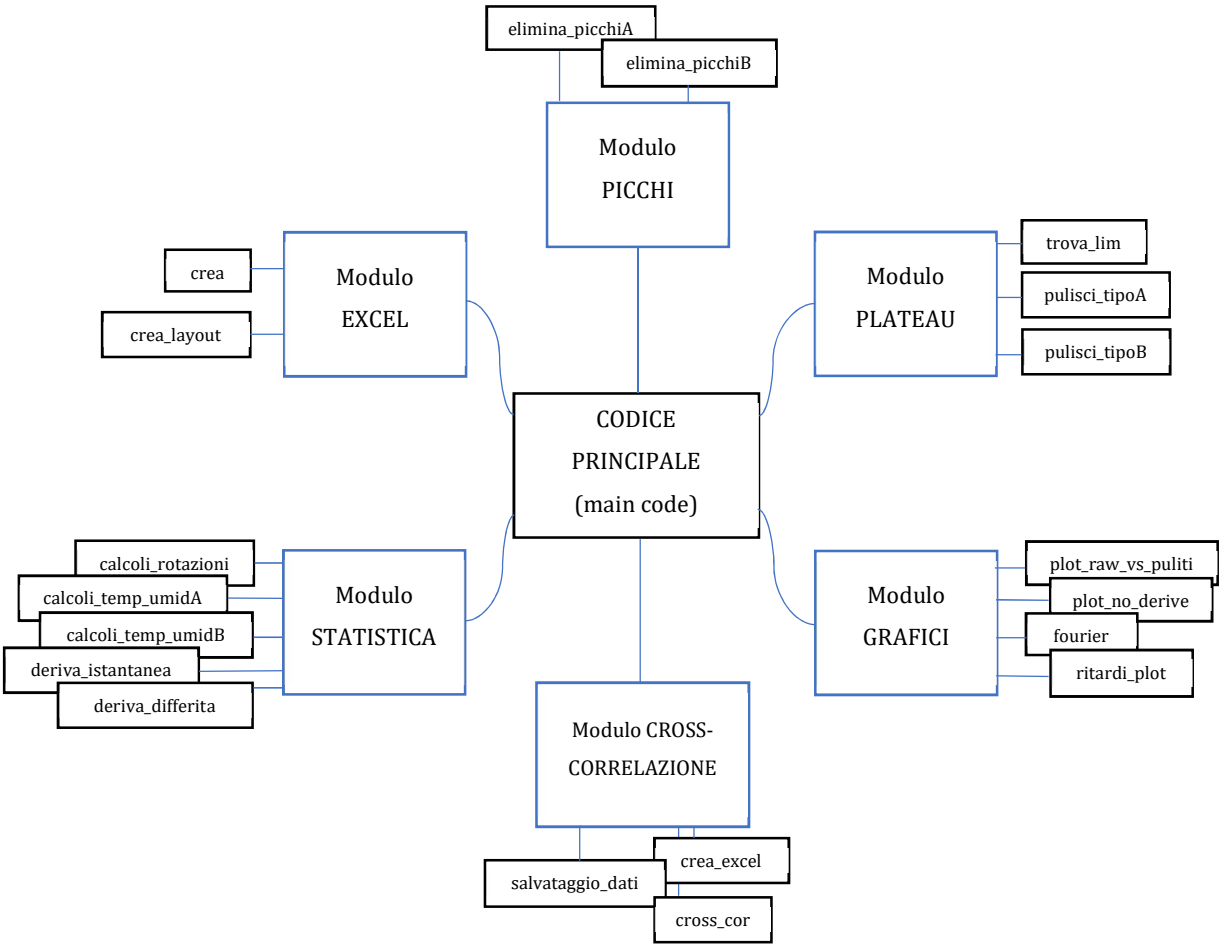


Figura 36: Moduli esterni creati e loro funzioni

Si noti che le variabili interne ai moduli, tra cui gli argomenti delle funzioni, non hanno nulla a che vedere con le variabili del codice principale; per questo, all'interno del codice principale, si possono assegnare agli argomenti di una funzione variabili con un nome qualsiasi, anche diverso da quello usato nel modulo esterno.

4.4. File di input e parametri scelti dall'utente

Per ciascun sensore è stata creata una cartella, allocata arbitrariamente all'interno del calcolatore, il cui nome è il codice identificativo dello strumento (ad esempio, "EL_C23-T2-N"). In essa è presente, in partenza, solo il file contenente i dati grezzi acquisiti dal sensore, in formato Comma Separated Values (csv); il nome di tale file è uguale al codice

dello strumento che rappresenta. Di conseguenza, il file che contiene le misurazioni effettuate dal sensore avente codice "EL_C23-T2-N" avrà un nome comprensivo di estensione uguale a "EL_C23-T2-N.csv".

Figura 37: Esempio di percorso del file EL_C23-T2-N.csv

Nel caso di sensori di tipo B, quelli che sono stati analizzati in questo documento, il file contenente i dati grezzi è costituito da tre colonne e un numero di righe pari al numero di misurazioni più una. La prima colonna contiene la serie temporale, il cui formato è dd/mm/yy HH.MM; la seconda colonna rappresenta la serie delle rotazioni grezze, mentre la terza contiene la serie delle temperature misurate. La prima riga è costituita dalle intestazioni ("datetime", "rotation", "temperature").

```
datetime,rotation,temperature
18/2/19 10.20,0.016,10.07872
18/2/19 10.40,-0.034,10.6257
18/2/19 11.00,-0.064,10.80803
18/2/19 11.20,-0.044,11.20485
18/2/19 11.40,-0.284,11.52661
18/2/19 12.00,-0.064,11.83763
18/2/19 12.20,-0.094,12.00923
18/2/19 12.40,-0.124,12.10576
18/2/19 13.00,-0.104,12.09503
18/2/19 13.20,-0.044,12.33099
18/2/19 13.40,-0.094,12.29881
18/2/19 14.00,-0.014,12.38461
18/2/19 14.20,-0.034,12.65274
18/2/19 14.40,-0.044,12.73854
18/2/19 15.00,-0.034,12.65274
```

Figura 38: Configurazione del file EL_C23-T2-N.csv

Altri parametri che possono essere definiti dall'utente, quali ad esempio il percorso delle cartelle i cui cercare i file dei dati grezzi per i singoli sensori, o le impostazioni per tracciare i grafici, vengono inseriti in un file di input in formato testo (txt), che dovrà essere posizionato nella stessa cartella in cui si trova il codice; questo file viene fornito assieme al codice, e si chiama "file di input utente.txt". I singoli parametri verranno analizzati man mano che si incontreranno nell'analisi dettagliata del codice.

```
(1) PERCORSO FILE DA ACQUISIRE: INSERIRE IL NOME DEL FILE SENZA ESTENSIONE E IL PERCORSO DELLA CARTELLA DEL FILE
SEPARATI DA UNA VIRGOLA,PRECEDUTI DALLA PAROLA 'sensoreA' O 'sensoreB' A SECONDA DEL TIPO DI STRUMENTAZIONE CON
CUI SI SONO OTTENUTI I DATI,E SEGUITI DAL SIMBOLO '\'; PER ULTIMO INSERIRE 'sì' QUALORA SI VOGLIA INSERIRE LA
CURVA DEL SENSORE NEL GRAFICO RELATIVO AI COEFFICIENTI DI CORRELAZIONE CON TEMPERATURE RITARDATE; INFINE CHIUDERE
LA LINEA CON UNA VIRGOLA. QUALORA NON SI VOGLIANO ACQUISIRE I DATI  DI UNO SPECIFICO SENSORE, SI PONGA UN
QUALSIASI CARATTERE A INIZIO RIGA; DI SEGUITO, UN ESEMPIO.
Esempio: tipo sensore,nome sensore,percorso file\,sì,

sensoreB,EL_C23-T2-N,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T2-N\,sì,
sensoreB,EL_C23-T2-S,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T2-S\,sì,
sensoreB,EL_C23-T3-N,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T3-N\,sì,
sensoreB,EL_C23-T3-S,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T3-S\,sì,
sensoreB,EL_C23-T4-N,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T4-N\,sì,
sensoreB,EL_C23-T4-S,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T4-S\,sì,
sensoreB,EL_C23-T5-N,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T5-N\,sì,
sensoreB,EL_C23-T5-S,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T5-S\,sì,
...

(2) PERCORSO CARTELLA DI OUTPUT: INSERIRE PERCORSO DELLA CARTELLA DI OUTPUT IN CUI INSERIRE I RISULTATI FINALI

risultati,C:\Users\utente\Desktop\Unione codici\RISULTATI,

(3.1) PARAMETRI PER GRAFICI: DATI I PARAMETRI SCRITTI SOTTO, LASCIARE UN UNICO SPAZIO
DOPO IL SIMBOLO ':' E INSERIRE IL VALORE DESIDERATO
(SI SUGGERISCE FORTEMENTE DI LASCIARE QUELLI PRESENTI):

Altezza_grafico: 15
Larghezza_grafico: 30
Dimensione_titolo: 30
Dimensioni_etichette_assi: 22
Distanza_etichette_da_asse: 12
Dimensioni_valori_numerici_assi: 19
Spessore_di_linea_della_curva: 2
Inizio_ascisse_tipoA: 1550000000
Fine_ascisse_tipoA: 1560745495
Inizio_ascisse_tipoB: 1550471000
Fine_ascisse_tipoB: 1559655000
```

Figura 39: Configurazione della prima parte di "file di input utente.txt"

```
(3.2) PARAMETRI PER GRAFICI: CIASCUNA RIGA RAPPRESENTA UN GRAFICO: è FORMATA DALLA PAROLA 'grafico', SEGUITA DAL
TITOLO DEL GRAFICO, E DALL'ISTANTE DI INIZIO E DI FINE DELL'ASSE TEMPORALE CHE SI VUOLE RAPPRESENTARE SULLE
ASCISSE (in formato dd/mm/yy HH.MM), DIVISI DA UNA VIRGOLA. SI CONSIGLIA DI NON MODIFICARE LE PRIME CINQUE RIGHE,
O AL PIù DI MODIFICARE SOLO I LIMITI TEMPORALI DELL'ASSE DELLE ASCISSE. LE RIGHE DALLA SESTA COMPRESA IN POI
DEFINISCONO LO STESSO TIPO DI GRAFICO PLOTTATO PER LIMITI TEMPORALI DELL'ASSE DELLE ASCISSE DIFFERENTI; PERCIò è
POSSIBILE ELIMINARE RIGHE, AGGIUNGERE RIGHE O VARIARE I LIMITI DELLE ASCISSE SULLE RIGHE PRESENTI.

Titolo,Inizio_ascisse,Fine_ascisse,
grafico,RAW ROTATION AND MOVING AVERAGE ON {} hours {} minutes,18/2/19 10.20,4/6/19 12.20,
grafico,ROTATION K AND MOVING AVERAGE ON {} hours {} minutes,18/2/19 10.20,4/6/19 12.20,
grafico,ROTATION KK AND MOVING AVERAGE ON {} hours {} minutes,18/2/19 10.20,4/6/19 12.20,
grafico,MOVING AVERAGE ON {} hours {} minutes RAW K AND KK,18/2/19 10.20,4/6/19 12.20,
grafico,DIFFERENCE between ROTATION AND MOVING AVERAGE (for RAW ROTATION AND ROTATION K),18/2/19 10.20,4/6/19
12.20,
grafico,RAW ROTATION and MOVING AVERAGE ON {} hours {} minutes RAW K and KK (5 days in Febraury),19/2/19
18.58,24/2/19 18.59,
grafico,RAW ROTATION and MOVING AVERAGE ON {} hours {} minutes RAW K and KK (5 days in March),21/3/19
18.58,26/3/19 18.59,
grafico,RAW ROTATION and MOVING AVERAGE ON {} hours {} minutes RAW K and KK (March),1/3/19 0.00,31/3/19 23.59,
grafico,RAW ROTATION and MOVING AVERAGE ON {} hours {} minutes RAW K and KK (1 day in April),5/4/19 0.19,5/4/19
23.59,
grafico,RAW ROTATION and MOVING AVERAGE ON {} hours {} minutes RAW K and KK (whole period),18/2/19 10.20,4/6/19
12.20,

(4) DATI PER MEDIE MOBILI: INSERIRE L'INTERVALLO TEMPORALE SU CUI ANDARE AD ESEGUIRE LA MEDIA MOBILE E IL TEMPO
DI CAMPIONAMENTO DEL SENSORE IN SECONDI (il tempo di campionamento deve essere un sottomultiplo degli intervalli)

intervallo1[s]: 8400
intervallo2[s]: 10800
intervallo3[s]: 13200
tempo_campionamento[s]: 1200

(5) DATI PER IL CALCOLO DELLE DERIVE E DEL COEFFICIENTE DI CORRELAZIONE

estremo_sinistro_intervallo_deriva_istantanea[mrad/°C]: -0.2
estremo_destro_intervallo_deriva_istantanea[mrad/°C]: 0.2
massimo_errore_deriva_istantanea[mrad/°C]: 0.00005
estremo_sinistro_intervallo_deriva_differita[mrad/°C]: -0.2
estremo_destro_intervallo_deriva_differita[mrad/°C]: 0.2
massimo_errore_deriva_differita[mrad/°C]: 0.00005
indice_posizione_iniziale_dell'intervallo_di_correlazione: 320

(6) PARAMETRI PER GRAFICI DELLE TRASFORMATE DI FOURIER

limite_inferiore_asse_xA: -0.05
limite_superiore_asse_xA: 1
limite_inferiore_asse_yA: -5
limite_superiore_asse_yA: 2000

limite_inferiore_asse_xB: -0.05
limite_superiore_asse_xB: 0.4
limite_inferiore_asse_yB: -5
limite_superiore_asse_yB: 150
```

Figura 40: Configurazione della seconda parte di "file di input utente.txt"

```
(7) CROSS-CORRELAZIONE TRA SENSORI: RISPONDERE ALLE DUE DOMANDE, LASCIANDO UNO SPAZIO BIANCO DOPO IL PUNTO DI
DOMANDA E UNO DOPO LA RISPOSTA.

Effettuare_la_cross_correlazione_tra_diversi_sensori? sì
Quanti_gruppi_di_sensori_tra_cui_effettuare_la_cross_correlazione_si_vogliono_creare? 3

PER CIASCUN GRUPPO DI SENSORI DA CROSS-CORRELARE, INSERIRE UNA RIGA. PER OGNI RIGA, INSERIRE LA PAROLA "gruppo" E
IL
NUMERO DEL GRUPPO (indicizzato a partire da 0), SEGUITI DA DUE PUNTI E UNO SPAZIO; POI, INSERIRE I NOMI DEI
SENSORI
DA CROSS-CORRELARE IN CIASCUN GRUPPOSUDDIVISI DA VIRGOLE. CHIUDERE LA RIGA CON UNA VIRGOLA.

I sensori tra cui effettuare la cross_correlazione sono:
gruppo0: EL_C23-T2-N,EL_C23-T2-S,EL_C23-T3-N,EL_C23-T3-S,EL_C23-T4-N,EL_C23-T4-S,EL_C23-T5-N,EL_C23-T5-S,
gruppo1: EL_C23-T2-N,EL_C23-T3-N,EL_C23-T4-N,EL_C23-T5-N,
gruppo2: EL_C23-T2-S,EL_C23-T3-S,EL_C23-T4-S,EL_C23-T5-S,

PER CIASCUN GRUPPO DI SENSORI DA CROSS-CORRELARE, INSERIRE UNA RIGA. PER OGNI RIGA, INSERIRE I TERMINI
"parametri_gruppo" E IL
NUMERO DEL GRUPPO (indicizzato a partire da 0), SEGUITI DA DUE PUNTI E UNO SPAZIO; POI, INSERIRE
RISPETTIVAMENTE,SUDDIVISI DA VIRGOLE:
         - L'ISTANTE INIZIALE  DELL'INTERVALLO DI TEMPO IN CUI EFFETTUARE LA CROSS-CORRELAZIONE,
         - L'ISTANTE FINALE DELL'INTERVALLO DI TEMPO IN CUI EFFETTUARE LA CROSS-CORRELAZIONE,
         - OGNI QUANTI SECONDI PRENDERE UN DATO DA CROSS-CORRELARE,
         - MASSIMA DISTANZA, IN SECONDI, DELLA MISURAZIONE ACCETTATA RISPETTO ALL'ISTANTE VOLUTO,
           NEL CASO IN CUI NON ESISTA UNA MISURAZIONE RELATIVA ALL'ISTANTE DESIDERATO.
CHIUDERE LA RIGA CON UNA VIRGOLA.

parametri_gruppo0: 18/2/19 10.20,4/6/19 12.20,1200,180,
parametri_gruppo1: 18/2/19 10.20,4/6/19 12.20,1200,180,
parametri_gruppo2: 18/2/19 10.20,4/6/19 12.20,1200,180,
```

Figura 41: Configurazione della terza parte di "file di input utente.txt"

Poi, i valori di rotazione a ponte scarico e la corrispettiva misurazione di temperatura vengono inseriti in un altro file Comma Separated Values (csv), contenuto anch'esso nella stessa cartella in cui viene posizionato il codice, e chiamato "dati_iniziali_B.csv". Esso è costituito da tre colonne e un numero di righe pari al numero di sensori più una. La prima colonna contiene i nomi dei sensori; la seconda colonna rappresenta le rotazioni misurate a ponte scarico relative a ciascun sensore, mentre la terza contiene la serie delle temperature misurate a ponte scarico. La prima riga è costituita dalle intestazioni ("nome_sensore", "rotazione ponte scarico X [mrad]", "temperatura ponte scarico [°C]").

```
nome sensore,rotazione ponte scarico X [mrad],temperatura ponte scarico [°C]
EL_C23-T2-N,-0.01,10.14
EL_C23-T2-S,0,10.07
EL_C23-T3-N,0,10
EL_C23-T3-S,0,10.08
EL_C23-T4-N,0,9.95
EL_C23-T4-S,0,9.98
EL_C23-T5-N,0,9.89
EL_C23-T5-S,0,9.99
```

Figura 42: Configurazione di dati_iniziali_B.csv

Per i sensori di tipo A, si può usare lo stesso "file di input utente.txt", mentre esistono i corrispettivi per il file di input dei dati grezzi ("nome_sensore.csv") e per il file contenente i dati iniziali ("dati_iniziali_A.csv"); entrambi saranno formati da un maggior numero di colonne, in virtù del fatto che sensori di tipo A misurano un più alto numero di grandezze.

4.5. File di output generati dal programma

All'interno della cartella definita per ciascun sensore, viene generato un file in formato csv (Comma Separated Value) avente stesso layout del file contenente i dati grezzi misurati dal sensore (nome_sensore.csv), ma contenente le serie di dati ripulite dalle anomalie (plateaux e picchi). Si veda la descrizione del blocco (7) al paragrafo 4.7.6.
Inoltre, viene generata una cartella, denominata "Immagini di output", la quale a sua volta contiene due cartelle (tre per sensori di tipo A):

- "1) Dati ripuliti": cartella contenente i grafici creati alla fine della prima parte del codice, quella relativa alla pulizia dei dati da anomalie (blocco 6); tali grafici permettono il confronto tra le serie di misurazioni dei sensori e le serie pulite.

- "2) Dati scorrelati da temperatura": cartella contenente i grafici creati alla fine della seconda parte del codice, al termine della scorrelazione dalla temperatura (blocchi 18, 19 e 23); tali grafici permettono di confrontare rotazioni grezze, corrette e doppiamente corrette dopo lo svincolamento dei dati dalla temperatura.

- "3) Dati scorrelati da temperatura e umidità": cartella presente solo per sensori di tipo A, contenente i grafici creati alla fine della seconda parte del codice, al termine della scorrelazione dall'umidità (blocchi 18, 19 e 23); tali grafici permettono di confrontare rotazioni grezze, corrette e doppiamente corrette dopo lo svincolamento dei dati dalla temperatura e dall'umidità.

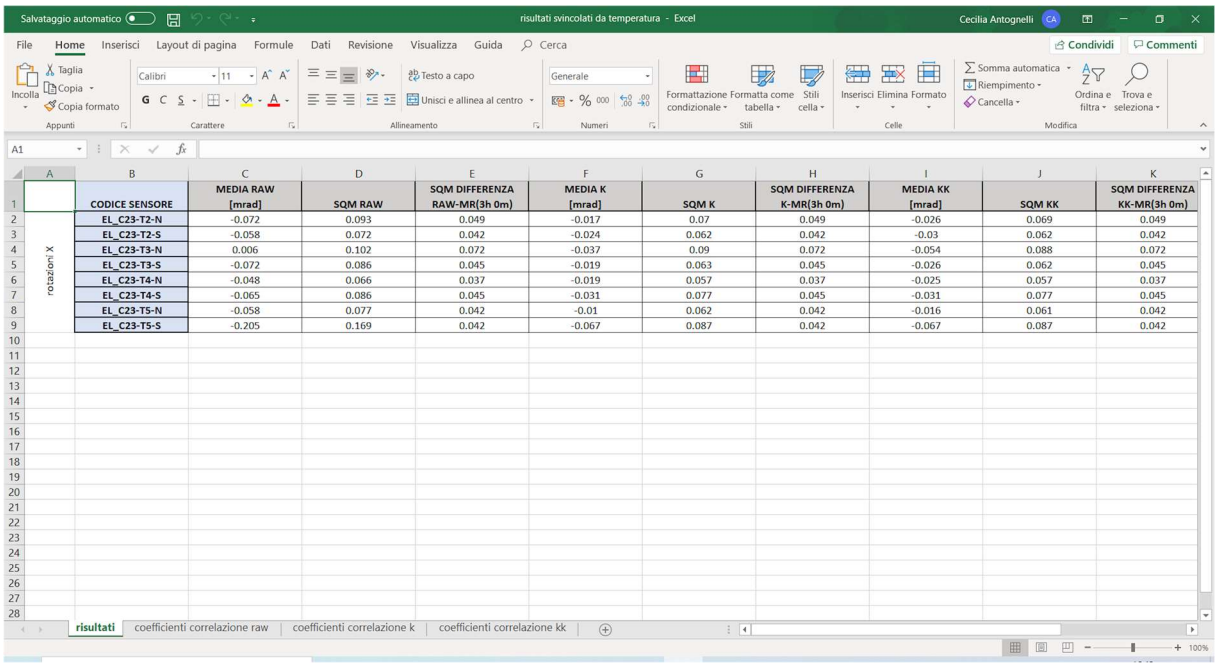
Al momento, i grafici che vengono generati nel blocco (18) sono uguali per tutti i sensori, e rappresentano:

- rotazione grezza ('raw') e sua media mobile centrata su calcolata sul secondo intervallo temporale definito dall'utente;
- rotazione corretta (k) e sua media mobile centrata su calcolata sul secondo intervallo temporale definito dall'utente;
- rotazione doppiamente corretta (kk) e sua media mobile centrata su calcolata sul secondo intervallo temporale definito dall'utente;
- media mobile calcolata sul secondo intervallo temporale definito dall'utente della serie grezza ('raw'), corretta (k) e doppiamente corretta (kk);
- differenza tra rotazione grezza ('raw') e sua media mobile centrata calcolata sul secondo intervallo temporale definito dall'utente, e differenza tra rotazione corretta (k) e sua media mobile centrata calcolata sul secondo intervallo temporale definito dall'utente;
- rotazione grezza ('raw') e media mobile centrata calcolata sul secondo intervallo temporale definito dall'utente per rotazioni grezze ('raw'), corrette (k), e doppiamente corrette (kk) per diversi intervalli temporali (19/02-24/02; 21/06-26/06; 01/03-31/03; 05/04; intero periodo); sono dunque cinque diversi grafici.

Poi, l'immagine contenente i tre grafici di ampiezza di fase delle trasformate di Fourier della serie di rotazioni grezza, corretta e doppiamente corretta, e quella contenente lo spettro di densità di energia per le stesse tre serie di dati, sono generate nel blocco (19).
Infine, il blocco (23) genera il grafico dei coefficienti di correlazione al variare del ritardo.

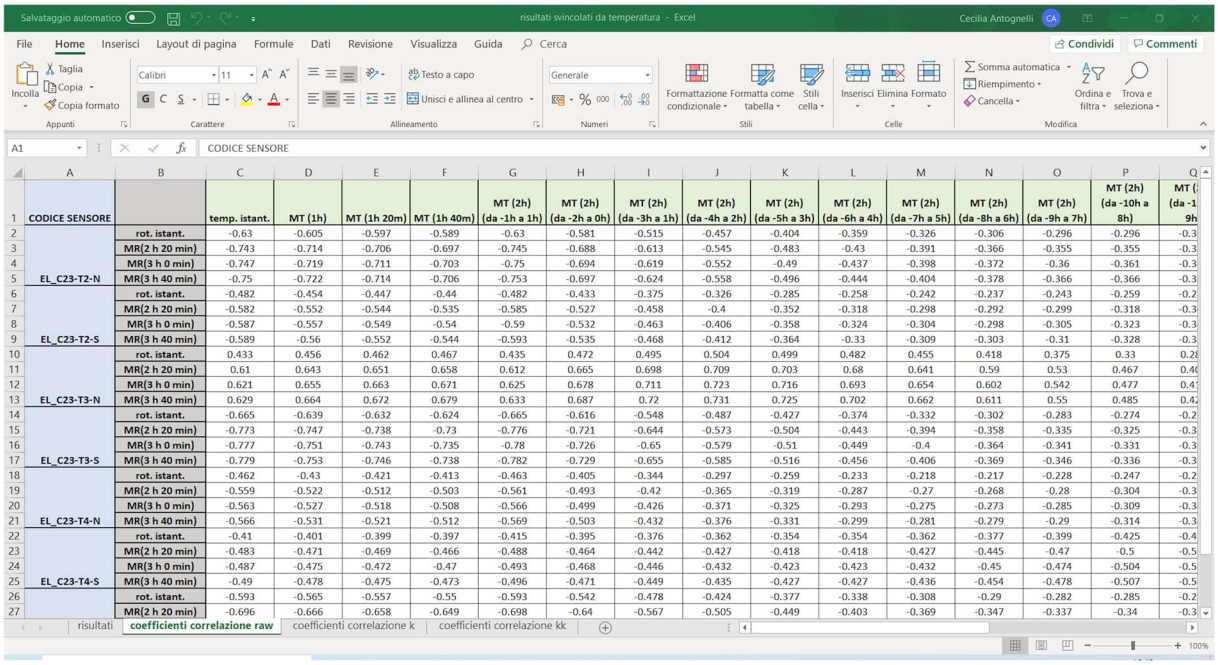
Inoltre, nel percorso definito per salvare i risultati (su "file di input utente.txt"), viene salvato un file Excel denominato "risultati scorrelati da temperatura.xlsx" al termine del ciclo di svincolamento dei dati dalla temperatura. Per sensori di tipo A, viene salvato anche un altro file denominato "risultati scorrelati da temperatura e umidità.xlsx", alla fine del secondo ciclo di scorrelazione, quello relativo all'umidità.
Per ciascuno strumento l'algoritmo scrive una riga sul primo foglio del file Excel dei risultati; in particolare per ognuno dei tre dataset descritti sopra (rotazioni 'raw', rotazioni k e rotazioni kk) vengono salvate la media, lo scarto quadratico medio della serie e lo scarto quadratico medio della differenza tra la serie di rotazioni e la sua media mobile centrata calcolata sul secondo intervallo temporale definito dall'utente. Inoltre, vengono salvati i valori di deriva termica istantanea e differita, e il ritardo ottimale delle

temperature usato per il calcolo delle rotazioni kk. Sul secondo, sul terzo e sul quarto foglio del file Excel dei risultati vengono salvati i coefficienti di correlazione tra le temperature (istantanee e loro medie) e rispettivamente le rotazioni raw, le rotazioni k e le rotazioni kk (istantanee e loro medie).



	A	B	C	D	E	F	G	H	I	J	K
1		CODICE SENSORE	MEDIA RAW [mrad]	SQM RAW	SQM DIFFERENZA RAW-MR(3h 0m)	MEDIA K [mrad]	SQM K	SQM DIFFERENZA K-MR(3h 0m)	MEDIA KK [mrad]	SQM KK	SQM DIFFERENZA KK-MR(3h 0m)
2	rotazioni X	EL_C23-T2-N	-0.072	0.093	0.049	-0.017	0.07	0.049	-0.026	0.069	0.049
3		EL_C23-T2-S	-0.058	0.072	0.042	-0.024	0.062	0.042	-0.03	0.062	0.042
4		EL_C23-T3-N	0.006	0.102	0.072	-0.037	0.09	0.072	-0.054	0.088	0.072
5		EL_C23-T3-S	-0.072	0.086	0.045	-0.019	0.063	0.045	-0.026	0.062	0.045
6		EL_C23-T4-N	-0.048	0.066	0.037	-0.019	0.057	0.037	-0.025	0.057	0.037
7		EL_C23-T4-S	-0.065	0.086	0.045	-0.031	0.077	0.045	-0.031	0.077	0.045
8		EL_C23-T5-N	-0.058	0.077	0.042	-0.01	0.062	0.042	-0.016	0.061	0.042
9		EL_C23-T5-S	-0.205	0.169	0.042	-0.067	0.087	0.042	-0.067	0.087	0.042

Figura 43: Foglio "risultati" del file "risultati scorrelati da temperatura.xlsx"

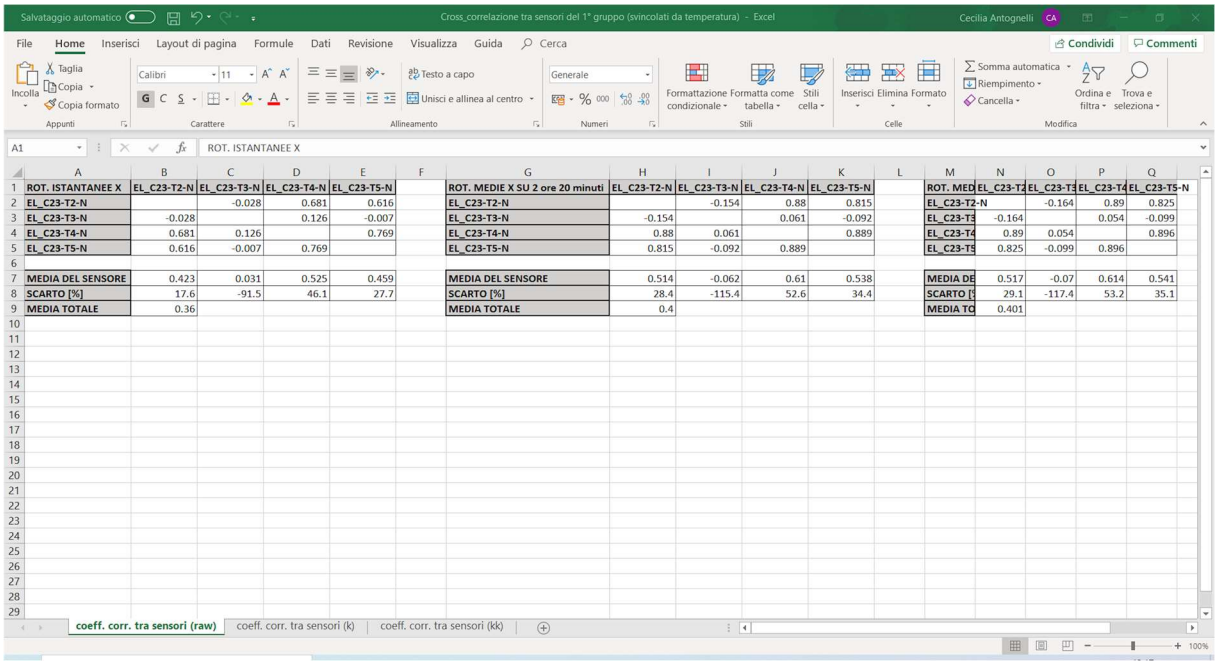


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	CODICE SENSORE		temp. istant.	MT (1h)	MT (1h 20m)	MT (1h 40m)	MT (2h) (da -1h a 1h)	MT (2h) (da -2h a 0h)	MT (2h) (da -3h a 1h)	MT (2h) (da -4h a 2h)	MT (2h) (da -5h a 3h)	MT (2h) (da -6h a 4h)	MT (2h) (da -7h a 5h)	MT (2h) (da -8h a 6h)	MT (2h) (da -9h a 7h)	MT (2h) (da -10h a 8h)	MT (2h) (da -1 9h
2		rot. istant.	-0.63	-0.605	-0.597	-0.589	-0.63	-0.581	-0.515	-0.457	-0.404	-0.359	-0.326	-0.306	-0.296	-0.296	-0.3
3		MR(2 h 20 min)	-0.743	-0.714	-0.706	-0.697	-0.745	-0.688	-0.613	-0.545	-0.483	-0.43	-0.391	-0.366	-0.355	-0.355	-0.3
4		MR(3 h 0 min)	-0.747	-0.719	-0.711	-0.703	-0.75	-0.694	-0.619	-0.552	-0.49	-0.437	-0.398	-0.372	-0.36	-0.361	-0.3
5	EL_C23-T2-N	MR(3 h 40 min)	-0.75	-0.722	-0.714	-0.706	-0.753	-0.697	-0.624	-0.558	-0.496	-0.444	-0.404	-0.378	-0.366	-0.366	-0.3
6		rot. istant.	-0.482	-0.454	-0.447	-0.44	-0.482	-0.433	-0.375	-0.326	-0.285	-0.258	-0.242	-0.237	-0.243	-0.259	-0.2
7		MR(2 h 20 min)	-0.582	-0.552	-0.544	-0.535	-0.585	-0.527	-0.458	-0.4	-0.352	-0.318	-0.298	-0.292	-0.299	-0.318	-0.3
8		MR(3 h 0 min)	-0.587	-0.557	-0.549	-0.54	-0.59	-0.532	-0.463	-0.406	-0.358	-0.324	-0.304	-0.298	-0.305	-0.323	-0.3
9	EL_C23-T2-S	MR(3 h 40 min)	-0.589	-0.56	-0.552	-0.544	-0.593	-0.535	-0.468	-0.412	-0.364	-0.33	-0.309	-0.303	-0.31	-0.328	-0.3
10		rot. istant.	0.433	0.456	0.462	0.467	0.435	0.472	0.495	0.504	0.499	0.482	0.455	0.418	0.375	0.33	0.2
11		MR(2 h 20 min)	0.61	0.643	0.651	0.658	0.612	0.665	0.698	0.709	0.703	0.68	0.641	0.59	0.53	0.467	0.4
12		MR(3 h 0 min)	0.621	0.655	0.663	0.671	0.625	0.678	0.711	0.723	0.716	0.693	0.654	0.602	0.542	0.477	0.4
13	EL_C23-T3-N	MR(3 h 40 min)	0.629	0.664	0.672	0.679	0.633	0.687	0.72	0.731	0.725	0.702	0.662	0.611	0.55	0.485	0.4
14		rot. istant.	-0.665	-0.639	-0.632	-0.624	-0.665	-0.616	-0.548	-0.487	-0.427	-0.374	-0.332	-0.302	-0.283	-0.274	-0.2
15		MR(2 h 20 min)	-0.773	-0.747	-0.738	-0.73	-0.776	-0.721	-0.644	-0.573	-0.504	-0.443	-0.394	-0.358	-0.335	-0.325	-0.3
16		MR(3 h 0 min)	-0.777	-0.751	-0.743	-0.735	-0.78	-0.726	-0.65	-0.579	-0.51	-0.449	-0.4	-0.364	-0.341	-0.331	-0.3
17	EL_C23-T3-S	MR(3 h 40 min)	-0.779	-0.753	-0.746	-0.738	-0.782	-0.729	-0.655	-0.585	-0.516	-0.456	-0.406	-0.369	-0.346	-0.336	-0.3
18		rot. istant.	-0.462	-0.43	-0.421	-0.413	-0.463	-0.405	-0.344	-0.297	-0.259	-0.233	-0.218	-0.217	-0.228	-0.247	-0.2
19		MR(2 h 20 min)	-0.559	-0.522	-0.512	-0.503	-0.561	-0.493	-0.42	-0.365	-0.319	-0.287	-0.27	-0.268	-0.28	-0.304	-0.3
20		MR(3 h 0 min)	-0.563	-0.527	-0.518	-0.508	-0.566	-0.499	-0.426	-0.371	-0.325	-0.293	-0.275	-0.273	-0.285	-0.309	-0.3
21	EL_C23-T4-N	MR(3 h 40 min)	-0.566	-0.531	-0.521	-0.512	-0.569	-0.503	-0.432	-0.376	-0.331	-0.299	-0.281	-0.279	-0.29	-0.314	-0.3
22		rot. istant.	-0.41	-0.401	-0.399	-0.397	-0.415	-0.395	-0.376	-0.362	-0.354	-0.354	-0.362	-0.377	-0.399	-0.425	-0.4
23		MR(2 h 20 min)	-0.483	-0.471	-0.469	-0.466	-0.488	-0.464	-0.442	-0.427	-0.418	-0.418	-0.427	-0.445	-0.47	-0.5	-0.5
24		MR(3 h 0 min)	-0.487	-0.475	-0.472	-0.47	-0.493	-0.468	-0.446	-0.432	-0.423	-0.423	-0.432	-0.45	-0.474	-0.504	-0.5
25	EL_C23-T4-S	MR(3 h 40 min)	-0.49	-0.478	-0.475	-0.473	-0.496	-0.471	-0.449	-0.435	-0.427	-0.427	-0.436	-0.454	-0.478	-0.507	-0.5
26		rot. istant.	-0.593	-0.565	-0.557	-0.55	-0.593	-0.542	-0.478	-0.424	-0.377	-0.338	-0.308	-0.29	-0.282	-0.285	-0.2
27		MR(2 h 20 min)	-0.696	-0.666	-0.658	-0.649	-0.698	-0.64	-0.567	-0.505	-0.449	-0.403	-0.369	-0.347	-0.337	-0.34	-0.3

Figura 44: Foglio "coefficienti di correlazione raw" del file "risultati scorrelati da temperatura.xlsx"

Infine, se l'utente vuole eseguire la cross-correlazione tra sensori, suddivisi in gruppi, viene generato un file Excel per ogni gruppo, contenente i coefficienti di cross-correlazione tra i diversi sensori di esso, e chiamato "Cross_correlazione tra sensori del k° gruppo". Questo file contiene tre fogli (uno relativo a rotazioni grezze, uno relativo a rotazioni corrette e l'ultimo relativo a rotazioni doppiamente corrette); ciascuno di essi contiene quattro tabelle per ciascun tipo di rotazioni (per sensori di tipo B, che misurano solo rotazioni X, ci saranno solo quattro tabelle, mentre per sensori di tipo A, che ricevono anche rotazioni Y, ce ne saranno otto), contenenti i coefficienti di correlazione delle serie di rotazioni istantanee e di medie mobili sui tre intervalli definiti dall'utente tra diversi sensori.

Inoltre, sono presenti anche le medie dei coefficienti di correlazione relativi a ogni sensore, la media di ciascuna famiglia, e lo scarto percentuale tra queste ultime due grandezze.



ROT. ISTANTANEE X	EL_C23-T2-N	EL_C23-T3-N	EL_C23-T4-N	EL_C23-T5-N
EL_C23-T2-N		-0.028	0.681	0.616
EL_C23-T3-N	-0.028		0.126	-0.007
EL_C23-T4-N	0.681	0.126		0.769
EL_C23-T5-N	0.616	-0.007	0.769	
MEDIA DEL SENSORE	0.423	0.031	0.525	0.459
SCARTO [%]	17.6	-91.5	46.1	27.7
MEDIA TOTALE	0.36			

ROT. MEDIE X SU 2 ore 10 minuti	EL_C23-T2-N	EL_C23-T3-N	EL_C23-T4-N	EL_C23-T5-N
EL_C23-T2-N		-0.154	0.88	0.815
EL_C23-T3-N	-0.154		0.061	-0.092
EL_C23-T4-N	0.88	0.061		0.889
EL_C23-T5-N	0.815	-0.092	0.889	
MEDIA DEL SENSORE	0.514	-0.062	0.61	0.538
SCARTO [%]	28.4	-115.4	52.6	34.4
MEDIA TOTALE	0.4			

ROT. MED	EL_C23-T2	EL_C23-T3	EL_C23-T4	EL_C23-T5-N
EL_C23-T2-N		-0.164	0.89	0.825
EL_C23-T3	-0.164		0.054	-0.099
EL_C23-T4	0.89	0.054		0.896
EL_C23-T5	0.825	-0.099	0.896	
MEDIA DE	0.517	-0.07	0.614	0.541
SCARTO [%	29.1	-117.4	53.2	35.1
MEDIA TO	0.401			

Figura 45: Foglio "coeff. corr. tra sensori (raw)" del file "Cross_correlazione tra sensori del 1° gruppo"

4.6. Operazioni preliminari

I primi due blocchi di codice contengono delle operazioni preliminari utili per la successiva esecuzione del codice e per il salvataggio dei risultati.

4.6.1. Blocco (1): apertura file di input e acquisizione dei parametri scelti dall'utente

In questo blocco di codice, sono presenti istruzioni per acquisire alcuni dati di input impostati dall'utente (contenuti in "file di input.txt").
Vengono creati alcuni vettori, in cui l'i-esimo elemento è un'informazione relativa all'i-esimo sensore. Essi sono:

- *percorso_file*: vettore in cui l'i-esima riga rappresenta il percorso della cartella relativa all'i-esimo sensore; esso ha lunghezza pari al numero di sensori analizzati;
- *nome_file*: vettore in cui l'i-esima riga rappresenta il nome (codice identificativo) dell'i-esimo sensore; esso ha lunghezza pari al numero di sensori analizzati;
- *vuoi_grafico*: vettore formato da 'sì' oppure 'no', in cui l'i-esima riga indica se si vuole rappresentare il sensore i-esimo nel grafico che rappresenta il coefficiente di correlazione al variare del ritardo della serie di temperature con cui è stato calcolato; esso ha lunghezza pari al numero di sensori analizzati;
- *nome_file_grafico*: vettore in cui l'i-esima riga rappresenta il nome (codice identificativo) dell'i-esimo sensore che si vuole rappresentare nel grafico che rappresenta il coefficiente di correlazione al variare del ritardo della serie di temperature con cui è stato calcolato; esso ha lunghezza pari al numero di sensori che si vogliono rappresentare (in generale diverso dal numero totale di sensori analizzati).

Questi vettori vengono inizializzati come vettori vuoti, cui viene aggiunto man mano il valore i-esimo. Per fare ciò, inizialmente essi vengono definiti come liste vuote ([ ]).
Il passo successivo consiste nell'aprire il "file di input utente.txt" tramite il comando open, il quale accetta come argomenti il percorso del file di testo che si vuole importare, e una stringa che definisca le modalità di apertura di esso. In questo caso, in cui il file deve essere

aperto in sola lettura, la stringa che viene inserita è 'r', corrispondente alla modalità "read". Poi, viene memorizzato il numero di righe del file tramite la funzione count, la quale permette di calcolare quante volte viene ripetuto un determinato carattere nel file in esame; in questo caso, la stringa che viene usata è '\n', la quale indica un ritorno a capo. Quindi, utilizzando un ciclo for sul numero di righe, le singole righe del file di testo vengono lette e memorizzate nella variabile interna al ciclo "lettura", per la quale si assume come divisore tra diversi elementi la virgola (",") usando il comando split. In questo modo, si definisce la variabile *riga1*, i cui elementi sono distinti da una virgola. Si inserisce, di seguito, un'immagine rappresentante la parte del "file di input utente.txt" che viene usata in questo blocco di codice, in modo da facilitare la comprensione delle righe successive.

```
(1) PERCORSO FILE DA ACQUISIRE: INSERIRE IL NOME DEL FILE SENZA ESTENSIONE E IL PERCORSO DELLA CARTELLA DEL FILE
SEPARATI DA UNA VIRGOLA,PRECEDUTI DALLA PAROLA 'sensoreA' O 'sensoreB' A SECONDA DEL TIPO DI STRUMENTAZIONE CON
CUI SI SONO OTTENUTI I DATI,E SEGUITI DAL SIMBOLO '\'; PER ULTIMO INSERIRE 'sì' QUALORA SI VOGLIA INSERIRE LA
CURVA DEL SENSORE NEL GRAFICO RELATIVO AI COEFFICIENTI DI CORRELAZIONE CON TEMPERATURE RITARDATE; INFINE CHIUDERE
LA LINEA CON UNA VIRGOLA. QUALORA NON SI VOGLIANO ACQUISIRE I DATI  DI UNO SPECIFICO SENSORE, SI PONGA UN
QUALSIASI CARATTERE A INIZIO RIGA; DI SEGUITO, UN ESEMPIO.
Esempio: tipo sensore,nome sensore,percorso file\,sì,

sensoreB,EL_C23-T2-N,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T2-N\,sì,
sensoreB,EL_C23-T2-S,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T2-S\,sì,
sensoreB,EL_C23-T3-N,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T3-N\,sì,
sensoreB,EL_C23-T3-S,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T3-S\,sì,
sensoreB,EL_C23-T4-N,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T4-N\,sì,
sensoreB,EL_C23-T4-S,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T4-S\,sì,
sensoreB,EL_C23-T5-N,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T5-N\,sì,
sensoreB,EL_C23-T5-S,C:\Users\utente\Desktop\Unione codici\DATI DEI SENSORI\EL_C23-T5-S\,sì,
...

(2) PERCORSO CARTELLA DI OUTPUT: INSERIRE PERCORSO DELLA CARTELLA DI OUTPUT IN CUI INSERIRE I RISULTATI FINALI

risultati,C:\Users\utente\Desktop\Unione codici\RISULTATI,
```

Figura 46: Parte di "file di input utente.txt" usata nel blocco (1.3)

All'interno del ciclo for, viene inserito un if: se il primo elemento della riga i-esima è pari a "sensoreA" o "sensoreB", esso viene salvato come il tipo di sensore che si sta analizzando (nella variabile *tipo_sensore*), inoltre vengono salvati il secondo, il terzo e il quarto elemento della riga rispettivamente nel vettore *nome_file*, *percorso_file* e *vuoi_grafico*. Se il quarto elemento della riga è uguale a 'sì', il suo secondo elemento (quello relativo al nome del sensore) viene salvato anche nel vettore *nome_file_grafico*.

Poi, ancora all'interno del ciclo for, se il primo elemento della riga è pari a "risultati", il suo secondo elemento viene salvato nella variabile *percorso_file_risultati*, che così diventa una stringa contenente il percorso della cartella in cui si intendono salvare i risultati.
Per inserire queste informazioni nel "file di input utente.txt", si seguano le istruzioni contenute in esso.
Segue la definizione di una nuova variabile, *riga2*, definita allo stesso modo di *riga1*, ma per la quale si usa come divisore tra diversi elementi della riga uno spazio bianco (" "). Questa è utile per acquisire la risposta alla domanda "Effettuare la cross-correlazione tra diversi sensori?", e il numero di gruppi all'interno dei quali si vuole effettuare la cross-correlazione tra sensori. Infatti, può essere necessario verificare la cross-correlazione tra sensori posizionati alla medesima altezza di travi diverse (si può costituire un gruppo di sensori per ogni posizione sulla trave), o tra sensori appartenenti alla stessa trave ma in posizioni speculari rispetto alla mezzeria (altri gruppi possono essere formati da queste coppie di sensori); non è altrettanto utile verificare la correlazione tra sensori in posizioni diverse su travi diverse, ad esempio.

```
#   (1) APERTURA FILE DI INPUT E ACQUISIZIONE DEI PARAMETRI SCELTI DALL'UTENTE

# 1.1 Inizializzazione dei vettori contenenti i nomi dei file, il loro percorso, e i nomi dei file che si vo
gliono avere nel grafico coeff.corr.-ritardo

nome_file = []              #Vettore con i nomi dei sensori
percorso_file = []          #Vettore con i percorsi dei file
vuoi_grafico = []           #Vettore di 'sì' o 'no', a seconda che si voglia avere il sensore i-
esimo nel grafico coeff.corr.-ritardo
nome_file_grafico = []      #Vettore con i nomi dei sensori che si vogliono avere nel grafico coeff.corr.-ritardo


# 1.2 Apertura del file di input dell'utente e conteggio del numero di righe

file1 = open(os.getcwd()+"\File di input utente.txt",'r')

numero_righe = file1.read().count('\n')


# 1.3 Apertura del file di input dell'utente, valutazione del tipo di sensore e acquisizione di dati

cross_correlazione_tra_sensori = 'no'

for n in range(numero_righe):

    lettura = getline(os.getcwd()+"\File di input utente.txt", n+1)         #Apertura del file
    riga1 = lettura.split(',')                                              #Elemento divisore della riga

#Riconoscimento delle linee contententi i percorsi dei file dei sensori
    if riga1[0] == 'sensoreA' or riga1[0] == 'sensoreB':

        tipo_sensore = riga1[0]                 #Il primo parametro sulla riga è la tipologia di sensore
        nome_file.append(str(riga1[1]))         #Il secondo parametro sulla riga è il nome del sensore
```

```
32.          percorso_file.append(str(riga1[2]))      #il terzo parametro sulla riga è il percorso della cartella contenent
    e il file
33.          vuoi_grafico.append(str(riga1[3]))       #Il quarto parametro sulla riga è 'sì' o 'no', a seconda che si vogli
    a avere il sensore i-esimo nel grafico coeff.corr.-ritardo
34.
35.          if riga1[3] == 'sì':
36.
37.              #Se si vuole avere il sensore i-esimo nel grafico coeff.corr.-ritardo, salva il nome in un altro vettore
38.              nome_file_grafico.append(str(riga1[1]))
39.
40.          continue
41.
42.      #Riconoscimento della linea contenente il percorso della cartella dei risultati
43.      elif riga1[0] == 'risultati':
44.
45.          percorso_file_risultati = riga1[1]        #Il secondo parametro della riga è il percorso della cartella che co
    nterrà i risultati
46.
47.          continue
48.
49.      #Divisione della riga secondo lo spazio bianco
50.      riga2 = lettura.split(' ')
51.
52.      #Riconoscimento della linea
53.      if riga2[0] == 'Effettuare_la_cross_correlazione_tra_diversi_sensori?':
54.
55.          cross_correlazione_tra_sensori = riga2[1]
             #Acquisizione della risposta alla domanda 'Effettuare_la_cross_correlazione_tra_diversi_sensori?'
56.
57.          continue
58.
59.      #Riconoscimento della linea
60.      if riga2[0] == 'Quanti_gruppi_di_sensori_tra_cui_effettuare_la_cross_correlazione_si_vogliono_creare?':
61.
62.          numero_gruppi_cross_correlazione = int(riga2[1])
             #Acquisizione della risposta alla domanda 'Quanti_gruppi_di_sensori_tra_cui_effettuare_la_cross_correlazione_si_vogli
    ono_creare?'
63.
64.          continue
```

Figura 47: Blocchi (1.1), (1.2) e (1.3) del codice

Si accede al blocco (1.4) solo se si vuole effettuare la cross-correlazione tra sensori, ovvero solo se la risposta alla domanda posta poco sopra è affermativa.

Vengono inizializzati due dizionari (*dizionario_gruppi_cross_corr* e *dizionario_timestamp_cross_corr*); questi sono un tipo di *built-in* di Python, mutabile e non ordinato, che contiene elementi (*items*) formati da una chiave (*key*) e un valore (*value*), il quale non necessariamente è uno scalare, ma può essere costituito anche da liste o da *array*. Una volta creato un dizionario, si può usare la chiave per richiamare il valore corrispondente. Nell'algoritmo che si sta descrivendo, si vuole usare come chiave la stringa "gruppok", dove k è l'indice del gruppo (considerando che il primo gruppo ha indice 0), mentre per il primo dizionario i valori sono costituiti dalle liste contenenti i codici dei sensori appartenenti a un determinato gruppo, per il secondo dizionario i valori

sono le linee temporali costituite dagli istanti con le misurazioni dei quali si vuole calcolare la cross-correlazione.

Fatto ciò, per il gruppo k-esimo si legge il file di input dell'utente riga per riga, si riconoscono le righe che iniziano con "gruppok" e "parametri_gruppok", e si ricavano la lista dei codici dei sensori appartenenti a quel gruppo, e i parametri usati per la definizione della linea temporale relativa a quel gruppo:

- istante iniziale della linea temporale (in formato gg/mm/aa h.min);
- istante finale della linea temporale (in formato gg/mm/aa h.min);
- intervallo di tempo tra una misurazione da usare per la cross-correlazione e la successiva (in *Figura 49*, Δt);
- massima distanza di una misurazione, i termini temporali, dall'istante desiderato nel caso in cui non esistesse una misurazione per quell'istante (in *Figura 49*, errore).

```
(7) CROSS-CORRELAZIONE TRA SENSORI: RISPONDERE ALLE DUE DOMANDE, LASCIANDO UNO SPAZIO BIANCO DOPO IL PUNTO DI
DOMANDA E UNO DOPO LA RISPOSTA.

Effettuare_la_cross_correlazione_tra_diversi_sensori? sì
Quanti_gruppi_di_sensori_tra_cui_effettuare_la_cross_correlazione_si_vogliono_creare? 3

PER CIASCUN GRUPPO DI SENSORI DA CROSS-CORRELARE, INSERIRE UNA RIGA. PER OGNI RIGA, INSERIRE LA PAROLA "gruppo" E
IL NUMERO DEL GRUPPO (indicizzato a partire da 0), SEGUITI DA DUE PUNTI E UNO SPAZIO; POI, INSERIRE I NOMI DEI
SENSORI DA CROSS-CORRELARE IN CIASCUN GRUPPOSUDDIVISI DA VIRGOLE. CHIUDERE LA RIGA CON UNA VIRGOLA.

I sensori tra cui effettuare la cross_correlazione sono:
gruppo0: EL_C23-T2-N,EL_C23-T2-S,EL_C23-T3-N,EL_C23-T3-S,EL_C23-T4-N,EL_C23-T4-S,EL_C23-T5-N,EL_C23-T5-S,
gruppo1: EL_C23-T2-N,EL_C23-T3-N,EL_C23-T4-N,EL_C23-T5-N,
gruppo2: EL_C23-T2-S,EL_C23-T3-S,EL_C23-T4-S,EL_C23-T5-S,

PER CIASCUN GRUPPO DI SENSORI DA CROSS-CORRELARE, INSERIRE UNA RIGA. PER OGNI RIGA, INSERIRE I TERMINI
"parametri_gruppo" E IL
NUMERO DEL GRUPPO (indicizzato a partire da 0), SEGUITI DA DUE PUNTI E UNO SPAZIO; POI, INSERIRE
RISPETTIVAMENTE,SUDDIVISI DA VIRGOLE:
         - L'ISTANTE INIZIALE  DELL'INTERVALLO DI TEMPO IN CUI EFFETTUARE LA CROSS-CORRELAZIONE,
         - L'ISTANTE FINALE DELL'INTERVALLO DI TEMPO IN CUI EFFETTUARE LA CROSS-CORRELAZIONE,
         - OGNI QUANTI SECONDI PRENDERE UN DATO DA CROSS-CORRELARE,
         - MASSIMA DISTANZA, IN SECONDI, DELLA MISURAZIONE ACCETTATA RISPETTO ALL'ISTANTE VOLUTO,
           NEL CASO IN CUI NON ESISTA UNA MISURAZIONE RELATIVA ALL'ISTANTE DESIDERATO.
CHIUDERE LA RIGA CON UNA VIRGOLA.

parametri_gruppo0: 18/2/19 10.20,4/6/19 12.20,1200,180,
parametri_gruppo1: 18/2/19 10.20,4/6/19 12.20,1200,180,
parametri_gruppo2: 18/2/19 10.20,4/6/19 12.20,1200,180,
```

Figura 48: Parte di "file di input utente.txt" usata nel blocco (1.3) e (1.4)

Con questi dati, viene creata la linea temporale in cui effettuare la cross-correlazione, in formati Unix Timestamp (*timestamp_cross_corr*). La lista dei nomi dei sensori appartenenti al gruppo, e la sua linea temporale, vengono inseriti nel dizionario relativo in corrispondenza della chiave "gruppok".

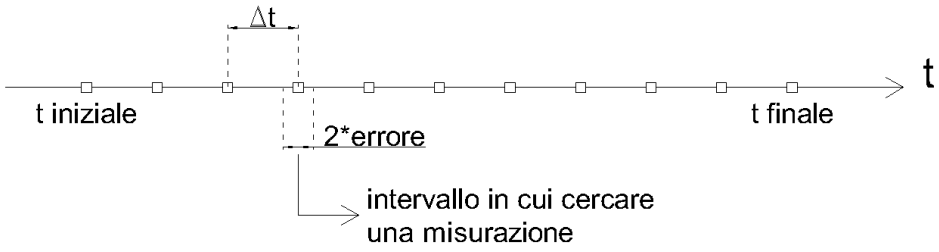


Figura 49: Linea temporale per cross-correlazione tra sensori

Poi, vengono inizializzati come vuoti dei dizionari di cui si parlerà più approfonditamente nel capitolo relativo al blocco (23).

Infine, dopo aver chiuso il "file di input utente.txt", si contano attraverso la funzione len il numero di sensori totali (*numero_sensori*) e il numero di sensori che si vuole rappresentare nel grafico coefficiente di correlazione - ritardo (*numero_sensori_grafico*); questa funzione restituisce la lunghezza di un vettore, ed è stata applicata rispettivamente a *nome_file* e *nome_file_grafico*.

```
# 1.4 Acquisizione dei parametri necessari per la cross correlazione

if cross_correlazione_tra_sensori == 'sì':

#Dizionario con i nomi dei sensori da cross-correlare, divisi per gruppi
    dizionario_gruppi_cross_corr = {}
#Vettore con le massime distanze accettate per i dati dall'istante in cui si voleva effettuare la cross-correlazione
    lista_cross_corr_err = []
#Dizionario con le linee temporali su cui effettuare la cross-correlazione, divisi per gruppi
    dizionario_timestamp_cross_corr = {}

    for k in range(numero_gruppi_cross_correlazione):

        for n in range(numero_righe):

            lettura = getline(os.getcwd()+"\File di input utente.txt", n+1)     #Apertura del file
            riga1 = lettura.split(': ')                                         #Elemento divisore della riga

#Riconoscimento della linea contenente i nomi dei sensori da cross-correlare, per ciascun gruppo
            if riga1[0] == 'gruppo{}'.format(k):


                nuovo_val.pop()                                 #Eliminazione del valore '/n' dalla lista creatasi
```

```
                dizionario_gruppi_cross_corr['gruppo{}'.format(k)] = nuovo_val      #Inserimento della lista dei nomi nel dizionario corrispondente, usando come chiave 'gruppoN', dove N è il numero del gruppo

#Riconoscimento della linea contenente i parametri per la cross-correlazione, per ciascun gruppo
            elif riga1[0] == 'parametri_gruppo{}'.format(k):

                cross_corr_t_ini = datetime.timestamp(datetime.strptime(riga1[1].split(',')[0], '%d/%m/%y %H.%M'))      #Acquisizione dell'istante iniziale da cui fare la cross-correlazione
                cross_corr_t_fin = datetime.timestamp(datetime.strptime(riga1[1].split(',')[1], '%d/%m/%y %H.%M'))      #Acquisizione dell'istante finale fino a cui fare la cross-correlazione
                cross_corr_delta_t = int(riga1[1].split(',')[2])
                #Acquisizione della distanza tra due istanti da cross-correlare
                lista_cross_corr_err.append(int(riga1[1].split(',')[3]))
                #Acquisizione delle massime distanze accettate per i dati dall'istante in cui si voleva effettuare la cross-correlazione

#Creazione della linea temporale su cui effettuare la cross-correlazione

              if tipo_sensore == 'sensoreA':                                        #Operazioni conseguenti al fatto che il sensore A acquisisce dati con linea temporale in millisecondi

                    lista_cross_corr_err[k] = lista_cross_corr_err[k] *1000         #Conversione della linea temporale in millisecondi
                    timestamp_cross_corr = timestamp_cross_corr * 1000              #Conversione della massima distanza temporale in millisecondi

              dizionario_timestamp_cross_corr['gruppo{}'.format(k)] = timestamp_cross_corr            #Inserimento della lista delle linee temporali nel dizionario corrispondente, usando come chiave 'gruppoN', dove N è il numero del gruppo

    #1.5 Inizializzazione di dizionari che serviranno per il salvataggio dei dati

    if tipo_sensore == 'sensoreA':

        diz_timestamp_cross_corrX_temp = {}
        diz_timestamp_cross_corrY_temp = {}
        diz_timestamp_cross_corrX_umid = {}
        diz_timestamp_cross_corrY_umid = {}

        diz_rotazioni_grezzeX_temp = {}
        diz_media_mobile_1X_temp = {}
        diz_media_mobile_2X_temp = {}
        diz_media_mobile_3X_temp = {}
        diz_rotazioni_correttеX_temp = {}
        diz_media_mobile_1_kX_temp = {}
        diz_media_mobile_2_kX_temp = {}
        diz_media_mobile_3_kX_temp = {}
        diz_rotazioni_corrette_due_volteX_temp = {}
        diz_media_mobile_1_kkX_temp = {}
        diz_media_mobile_2_kkX_temp = {}
        diz_media_mobile_3_kkX_temp = {}

        diz_rotazioni_grezzeY_temp = {}
        diz_media_mobile_1Y_temp = {}
        diz_media_mobile_2Y_temp = {}
        diz_media_mobile_3Y_temp = {}
        diz_rotazioni_correttеY_temp = {}
        diz_media_mobile_1_kY_temp = {}
        diz_media_mobile_2_kY_temp = {}
        diz_media_mobile_3_kY_temp = {}
        diz_rotazioni_corrette_due_volteY_temp = {}
        diz_media_mobile_1_kkY_temp = {}
        diz_media_mobile_2_kkY_temp = {}
        diz_media_mobile_3_kkY_temp = {}

        diz_rotazioni_grezzeX_umid = {}
```

```
        diz_media_mobile_1X_umid = {}
        diz_media_mobile_2X_umid = {}
        diz_media_mobile_3X_umid = {}
        diz_rotazioni_correttеX_umid = {}
        diz_media_mobile_1_kX_umid = {}
        diz_media_mobile_2_kX_umid = {}
        diz_media_mobile_3_kX_umid = {}
        diz_rotazioni_corrette_due_volteX_umid = {}
        diz_media_mobile_1_kkX_umid = {}
        diz_media_mobile_2_kkX_umid = {}
        diz_media_mobile_3_kkX_umid = {}

        diz_rotazioni_grezzeY_umid = {}
        diz_media_mobile_1Y_umid = {}
        diz_media_mobile_2Y_umid = {}
        diz_media_mobile_3Y_umid = {}
        diz_rotazioni_correttеY_umid = {}
        diz_media_mobile_1_kY_umid = {}
        diz_media_mobile_2_kY_umid = {}
        diz_media_mobile_3_kY_umid = {}
        diz_rotazioni_corrette_due_volteY_umid = {}
        diz_media_mobile_1_kkY_umid = {}
        diz_media_mobile_2_kkY_umid = {}
        diz_media_mobile_3_kkY_umid = {}

    if tipo_sensore == 'sensoreB':

        diz_timestamp_cross_corrX = {}

        diz_rotazioni_grezzeX = {}
        diz_media_mobile_1X = {}
        diz_media_mobile_2X = {}
        diz_media_mobile_3X = {}
        diz_rotazioni_correttеX = {}
        diz_media_mobile_1_kX = {}
        diz_media_mobile_2_kX = {}
        diz_media_mobile_3_kX = {}
        diz_rotazioni_corrette_due_volteX = {}
        diz_media_mobile_1_kkX = {}
        diz_media_mobile_2_kkX = {}
        diz_media_mobile_3_kkX = {}

file1.close()                                                    #Chiusura del file


# 1.6 Conteggio del numero dei sensori si cui l'utente vuole effettuare l'analisi e di quelli da graficare c
    on i ritardi

numero_sensori = len(nome_file)
numero_sensori_grafico = len(nome_file_grafico)
```

Figura 50: Blocchi (1.4), (1.5) e (1.6) del codice

La parte di codice successiva a questo punto, viene ripetuta per ogni sensore; per fare ciò, essa viene inserita in un ciclo for che assegna alla variabile “i” valori che vanno da 0 fino al numero dei sensori meno uno.

```
1.  #tutti punti seguenti verranno ripetute per ognuno dai sensori indicati dall'utente
2.  for i in range(numero_sensori):
```

Figura 51: Ciclo for sul numero dei sensori

4.7. Prima parte: pulizia da anomalie (plateaux e picchi)

Questa prima parte di codice tratta della pulizia dei dataset da eventuali anomalie (plateaux e picchi). Segue una spiegazione sommaria delle operazioni svolte dal codice.

4.7.1. Blocco (2): Acquisizione dei dati

In questa parte di codice vengono acquisite le misurazioni svolte dai sensori, le quali sono inserite nel file "nome_sensore.csv" descritto nel paragrafo 4.4. Questa operazione viene effettuata mediante una funzione appartenente al modulo Pandas, che permette di leggere file in formato Comma Separated Values. Così vengono generati i vettori relativi alle grandezze misurate dal sensore. Inoltre, visto che sensori di tipo A forniscono una linea temporale in formato Unix Timestamp, viene generato il vettore delle date e ore corrispondenti; per sensori B avviene esattamente l'opposto: le date fornite sono in formato stringa, dunque viene generato il vettore contenente i corrispettivi in formato Unix Timestamp.

4.7.2. Blocco (3): Operazioni di pulizia dei dati raw dai plateaux

Il blocco (3) contiene tutte le istruzioni necessarie per pulire i dati da eventuali plateau, ovvero quei tratti in cui il valore di una determinata grandezza misurata dal sensore si è mantenuto invariato per diverse acquisizioni. Si deve tener presente, però, che non tutti i tratti che presentano misure identiche sono da eliminare; dunque si usa la funzione Plateau.limiti per tarare, sulla base della media e dello scarto quadratico medio dei plateaux esistenti, il numero di acquisizioni limite oltre il quale il plateau viene considerato un errore. Poi, attraverso Plateau.pulisci, i valori costituenti i plateaux considerati anomali per una certa serie di dati vengono eliminati da esso, e con questi vengono scartati anche le corrispettive acquisizioni delle altre grandezze. Le operazioni appena descritte sono ripetute per tutte le grandezze misurate. Si ottengono dei vettori "puliti".

4.7.3. Blocco (4): Operazioni di pulizia dei dati raw dai picchi

Il blocco successivo di codice ha come scopo quello di eliminare eventuali picchi, ovvero quei valori puntuali molto diversi dalla media. Si utilizzano funzioni diverse a seconda del tipo di sensore, in particolare si usano Picchi.elimina_picchiA e Picchi.elimina_picchiB, appartenenti al modulo esterno Picchi. Le operazioni appena descritte sono ripetute per tutte le grandezze misurate. Si ottengono dei vettori "puliti".

4.7.4. Blocco (5): Creazione delle directory di salvataggio

Questo blocco crea le cartelle e le sottocartelle nelle quali salvare le immagini di output del codice, così come sono state descritte nel capitolo 4.5. Se queste cartelle sono già esistenti (generate da una precedente esecuzione del programma), vengono eliminate assieme al loro contenuto, e solamente in seguito create di nuovo.

4.7.5. Blocco (6): Grafici

Le righe di codice contenute nel blocco (6) hanno lo scopo di generare dei grafici che permettano di confrontare ciascuna grandezza come è stata misurata dal sensore, con la stessa al termine della pulizia; la funzione usata è Grafici.raw_vs_puliti, la quale, per ogni grandezza, genera un'immagine contenente due grafici: nel primo è rappresentata la serie di dati grezzi, nel secondo la serie pulita. Le immagini ottenute vengono salvate nella cartella "1) Dati ripuliti".

4.7.6. Blocco (7): Creazione file di output con dati puliti

Questo blocco del codice, per ogni sensore, crea e salva le serie di dati pulite in un file in formato Comma Separated Value avente stesso layout del file contenente i dati grezzi misurati dal sensore. Analogamente, il nome di questo file è costituito dal codice identificativo del sensore, cui segue "_pulito"; per il sensore con codice identificativo "EL_C23-T2-N" questo file sarà "EL_C23-T2-N_pulito.csv". La cartella in cui viene salvato è quella relativa al singolo sensore.

4.8. Seconda parte: analisi dei dati e svincolamento da temperatura (o umidità)

Il codice prosegue analizzando le serie di dati così ripulite da eventuali anomalie; vengono eseguiti calcoli di tipo statistico, e vengono calcolate le derive istantanee e differite, in modo tale da ottenere, infine, delle misurazioni svincolate dalla temperatura per sensori di tipo B, e delle misurazioni svincolate sia da temperatura che da umidità per sensori di tipo A.

I paragrafi successivi spiegano approfonditamente la parte di codice che riguarda i sensori in esame, ovvero quelli di tipo B; tutte le istruzioni inserite in un ciclo if cui si accede solo se la tipologia di sensore è la A, verranno spiegate sommariamente al capitolo 4.9.

4.8.1. Blocco (8): Definizione dei data-set e acquisizione dei parametri scelti dall'utente

Vengono definiti i vettori rappresentanti la serie temporale (*data*), la serie di rotazioni (*rotX*) e quella delle temperature (*temp*), ponendoli uguali a quelli ottenuti in output dalle operazioni precedenti, ovvero *data_pulita*, *rotX_pulita* e *temperatura_pulita*. Questo passaggio è fondamentalmente un'operazione superflua, ma utile per lavorare con variabili dal nome più immediato; inoltre, le variabili diventano, da liste quali erano (class "list"), dei vettori di Numpy (class "numpy.array").

La lista, che è una sequenza mutabile di oggetti, è caratterizzata dalla possibilità di contenere elementi eterogenei (è un oggetto dynamic-type), ma questa flessibilità ha un costo in termini di memoria: infatti, ogni elemento della lista deve contenere informazioni sul proprio tipo, il conteggio di riferimento ed altre; in pratica, ogni elemento è un oggetto Python a tutti gli effetti. Anche quando la lista contiene elementi omogenei, molte informazioni sono ridondanti. Per questo, può essere conveniente memorizzare i dati in un oggetto fixed-type, come è numpy.array.

Viene quindi ricalcolato il numero di misurazioni dopo la pulizia effettuata nella prima parte del codice (*N*), calcolando la lunghezza di uno qualsiasi dei data-set appena definiti.

```
1.  #   (8) ACQUISIZIONE DEI PARAMETRI PER GLI ALGORITMI DI SCORRELAZIONE DEI VALORI DA TEMPERATURA E UMIDITA'
2.
3.  # 8.1 Creazione dei vettori che verranno manipolati negli algoritmi di scorrelazione dei dati
```

```
4.
5.  data = np.array(data_pulita)            #Vettore delle date per entrambi i sensori
6.  rotX = np.array(rotX_pulita)            #Vettore delle rotazioni X per entrambi i sensori
7.  temp = np.array(temperatura_pulita)     #Vettore delle temperature per entrambi i sensori
8.
9.  if tipo_sensore == 'sensoreA':          #Il sensore di tipo A ha due grandezze in più
10.
11.     rotY = np.array(rotY_pulita)        #Vettore delle rotazioni Y per il sensore di tipo A
12.     umid = np.array(umidità_pulita)     #Vettore delle umidità per il sensore di tipo A
13.
14.
15. #8.2 Calcolo dell lunghezza delle serie di dati dopo la pulizia dai plateau
16.
17. N = len(data)
```

Figura 52: Blocchi (8.1) e (8.2) del codice

Viene, poi, aperto di nuovo "file di input utente.txt" con gli stessi comandi presentati al paragrafo 4.6.1; di nuovo vengono lette, una per volta, tutte le righe del file, ma ora si assume come divisore tra i diversi elementi della riga uno spazio vuoto (" ").
Si ricavano con semplici istruzioni condizionali definite da if ed elif i parametri necessari per l'esecuzione delle parti successive di codice:

- primo intervallo temporale su cui calcolare le medie mobili di rotazione (*int1*);
- secondo intervallo temporale su cui calcolare le medie mobili di rotazione (*int2*);
- terzo intervallo temporale su cui calcolare le medie mobili di rotazione (*int3*);
- tempo di campionamento del sensore (*tc*);
- estremo sinistro dell'intervallo di valori che può assumere la deriva istantanea (*Ai*);
- estremo destro dell'intervallo di valori che può assumere la deriva istantanea (*Bi*);
- massimo errore che si assume valido per la definizione della deriva istantanea (*err_i*);
- estremo sinistro dell'intervallo di valori che può assumere la deriva differita (*Ad*);
- estremo destro dell'intervallo di valori che può assumere la deriva differita (*Bd*);
- massimo errore che si assume valido per la definizione della deriva differita (*err_d*);
- indice della posizione iniziale dell'intervallo di valori dei data-set di rotazione e temperatura che si usa per calcolare i coefficienti di correlazione (*C*).

Poi viene chiuso il file di testo che era stato aperto, e si calcola in maniera automatica a partire dal numero delle misurazioni (*N*) l'indice della posizione finale dell'intervallo di valori dei data-set di rotazione e temperatura che si usa per calcolare i coefficienti di correlazione (*D*).
Infine, vengono calcolati il numero di misurazioni che sono contenute nei tre intervalli temporali su cui fare le medie mobili (*val1*, *val2*, *val3*), e le loro lunghezze corrispettive in ore e minuti (*ore1* e *minuti1*, *ore2* e *minuti2*, *ore3* e *minuti3*).

```
#8.3 Acquisizione da file dei parametri necessari per gli algoritmi di scorrelazione

file1 =open(os.getcwd()+"\File di input utente.txt",'r')       #Apertura File di input utente.txt

numero_righe = file1.read().count('\n')                #Conteggio del numero di righe del File di input utente.txt

for n in range(0,numero_righe):

    lettura = getline(os.getcwd()+"\File di input utente.txt", n+1)     #Lettura della riga
    riga2 = lettura.split(' ')                                          #Elemento divisore della riga

    if riga2[0] == 'estremo_sinistro_intervallo_deriva_istantanea[mrad/°C]:':

        Ai = float(riga2[1])            #Acquisizione dell'estremo sinistro dell'intervallo di tentativo per il calcolo della deriva istantanea

    elif riga2[0] == 'estremo_destro_intervallo_deriva_istantanea[mrad/°C]:':

        Bi = float(riga2[1])            #Acquisizione dell'estremo destro dell'intervallo di tentativo per il calcolo della deriva istantanea

    elif riga2[0] == 'massimo_errore_deriva_istantanea[mrad/°C]:':

        err_i = float(riga2[1])         #Acquisizione della tolleranza nell'algoritmo di ottimizzazione del valore di deriva istantanea

    elif riga2[0] == 'estremo_sinistro_intervallo_deriva_differita[mrad/°C]:':

        Ad = float(riga2[1])            #Acquisizione dell'estremo sinistro dell'intervallo di tentativo per il calcolo della deriva differita

    elif riga2[0] == 'estremo_destro_intervallo_deriva_differita[mrad/°C]:':

        Bd = float(riga2[1])            #Acquisizione dell'estremo destro dell'intervallo di tentativo per il calcolo della deriva differita

    elif riga2[0] == 'massimo_errore_deriva_differita[mrad/°C]:':

        err_d = float(riga2[1])         #Acquisizione della tolleranza nell'algoritmo di ottimizzazione del valore di deriva differita

    elif riga2[0] == "indice_posizione_iniziale_dell'intervallo_di_correlazione:":

        C = int(riga2[1])               #Acquisizione dell'indice di posizione iniziale dell'intervallo di correlazione (per escludere i nan iniziali)

    elif riga2[0] == "intervallo1[s]:":

        int1 = int(riga2[1])            #Acquisizione dell'intervallo 1 su cui effettuare la media mobile in secondi

    elif riga2[0] == "intervallo2[s]:":

        int2 = int(riga2[1])            #Acquisizione dell'intervallo 2 su cui effettuare la media mobile in secondi

    elif riga2[0] == "intervallo3[s]:":

        int3 = int(riga2[1])            #Acquisizione dell'intervallo 3 su cui effettuare la media mobile in secondi

    elif riga2[0] == "tempo_campionamento[s]:":

        tc = int(riga2[1])              #Acquisizione del tempo di campionamento in secondi


D = N - 15                  #Calcolo dell'indice di posizione iniziale dell'intervallo di correlazione (per escludere i nan finali)

```

```
59.  file1.close()                #Chiusura del file
60.
61.  val1 = int( int1 / tc )      #Calcolo dei valori contenuti nell'intervallo 1 in cui effettuare la media mobile
62.  val2 = int( int2 / tc )      #Calcolo dei valori contenuti nell'intervallo 2 in cui effettuare la media mobile
63.  val3 = int( int3 / tc )      #Calcolo dei valori contenuti nell'intervallo 3 in cui effettuare la media mobile
64.
65.  ore1 = int (int1 // 3600)                                                    #Calcolo della durata in ore dell'interval
     lo 1 in cui effettuare la media mobile
66.  minuti1 = int ( round( (int1 / 3600 - int(int1 // 3600)) * 60, 0))      #Calcolo della durata in minuti dell'inter
     vallo 1 in cui effettuare la media mobile
67.
68.  ore2 = int (int2 // 3600)                                                    #Calcolo della durata in ore dell'interval
     lo 2 in cui effettuare la media mobile
69.  minuti2 = int ( round( (int2 / 3600 - int(int2 // 3600)) * 60, 0))      #Calcolo della durata in minuti dell'inter
     vallo 2 in cui effettuare la media mobile
70.
71.  ore3 = int (int3 // 3600)                                                    #Calcolo della durata in ore dell'interval
     lo 3 in cui effettuare la media mobile
72.  minuti3 = int ( round( (int3 / 3600 - int(int3 // 3600)) * 60, 0))      #Calcolo della durata in minuti dell'inter
     vallo 3 in cui effettuare la media mobile
```

Figura 53: Blocco (8.3) del codice

Le modalità con cui i valori assegnati ai parametri vanno inseriti nel "file di input utente.txt" sono descritte in esso, così come le unità di misura con cui si devono inserire. Segue un ingrandimento della parte di "file di input utente.txt" richiamata in queste righe di codice.

```
(4) DATI PER MEDIE MOBILI: INSERIRE L'INTERVALLO TEMPORALE SU CUI ANDARE AD ESEGUIRE LA MEDIA MOBILE E IL TEMPO
DI CAMPIONAMENTO DEL SENSORE IN SECONDI (il tempo di campionamento deve essere un sottomultiplo degli intervalli)

intervallo1[s]: 8400
intervallo2[s]: 10800
intervallo3[s]: 13200
tempo_campionamento[s]: 1200

(5) DATI PER IL CALCOLO DELLE DERIVE E DEL COEFFICIENTE DI CORRELAZIONE

estremo_sinistro_intervallo_deriva_istantanea[mrad/°C]: -0.2
estremo_destro_intervallo_deriva_istantanea[mrad/°C]: 0.2
massimo_errore_deriva_istantanea[mrad/°C]: 0.00005
estremo_sinistro_intervallo_deriva_differita[mrad/°C]: -0.2
estremo_destro_intervallo_deriva_differita[mrad/°C]: 0.2
massimo_errore_deriva_differita[mrad/°C]: 0.00005
indice_posizione_iniziale_dell'intervallo_di_correlazione: 320
```

Figura 54: Parte di "file di input utente.txt" usata nel blocco (8.3)

Le righe seguenti del codice richiamano il modulo esterno personalizzato Excel.

```
# 8.4 Creazione di un file Excel per scrivere i risultati

import Excel    #Importazione del modulo esterno che si occupa della creazione dei file Excel per i risultati

if i == 0:      #Se è il primo sensore...

    exc, foglio_ris, foglio_ccraw, foglio_cck, foglio_cckk = Excel.crea_layout(tipo_sensore, numero_sensori, ore1, minuti1, ore2, minuti2, ore3, minuti3)   #...Creazione del file Excel e del suo layout

Excel.crea(i, tipo_sensore, numero_sensori, foglio_ris, foglio_ccraw, foglio_cck, foglio_cckk)        #Scrittura delle intestazioni alle tabelle
```

Figura 55: Blocco (8.4) del codice

La prima funzione, che viene eseguita solo nel caso in cui il sensore i-esimo sia il primo, è Excel.crea_layout; questa ha come scopo la creazione del file Excel dei risultati, la definizione dei suoi diversi fogli, e la definizione delle intestazioni delle tabelle in cui si inseriranno poi i valori ottenuti. Gli argomenti richiesti sono:

- il tipo di sensore (*tipo_sensore*);
- il numero totale di sensori (*numero_sensori*);
- la durata in ore e minuti dei tre intervalli su cui effettuare le medie mobili (*ore1*, *minuti1*, *ore2*, *minuti2*, *ore3*, *minuti3*).

Essa restituisce in output la variabile contenente il file Excel creatosi, e le quattro variabili identificative dei corrispettivi fogli.

```
import openpyxl as xl

from openpyxl.styles import Font, PatternFill, Border, Side, NamedStyle, Alignment

stile1grigio = NamedStyle(name="stile1grigio")
stile1grigio.font = Font(bold = True)
stile1grigio.border = Border(left=Side(style='medium', color='FF000000'), right=Side(style='medium', color='FF000000'), top=Side(style='medium', color='FF000000'), bottom=Side(style='medium', color='FF000000'))
stile1grigio.fill = PatternFill(start_color='D1D0CE', end_color='D1D0CE', fill_type='solid')
stile1grigio.alignment = Alignment(horizontal='center', wrapText=True)

stile1azzurro = NamedStyle(name="stile1azzurro")
stile1azzurro.font = Font(bold = True)
stile1azzurro.border = Border(left=Side(style='medium', color='FF000000'), right=Side(style='medium', color='FF000000'), top=Side(style='medium', color='FF000000'), bottom=Side(style='medium', color='FF000000'))
stile1azzurro.fill = PatternFill(start_color='d9e2f3', end_color='d9e2f3', fill_type='solid')
stile1azzurro.alignment = Alignment(horizontal='center')

stile1verde = NamedStyle(name="stile1verde")
stile1verde.font = Font(bold = True)
stile1verde.border = Border(left=Side(style='medium', color='FF000000'), right=Side(style='medium', color='FF000000'), top=Side(style='medium', color='FF000000'), bottom=Side(style='medium', color='FF000000'))
stile1verde.fill = PatternFill(start_color='e2efd9', end_color='e2efd9', fill_type='solid')
stile1verde.alignment = Alignment(horizontal='center', wrapText=True)

#FUNZIONE EXCEL(1) --> Questa funzione crea il file Excel dei risultati e ne predispone il layout
def crea_layout(tipo_sensore, numero_sensori, ore1, minuti1, ore2, minuti2, ore3, minuti3):

```

```
    #Creazione del file Excel dei risultati e rinominazione dei fogli
    exc = xl.Workbook()
    foglio_ris = exc.active
    foglio_ris.title = "risultati"
    foglio_ccraw = exc.create_sheet("coefficienti correlazione raw")
    foglio_cck = exc.create_sheet("coefficienti correlazione k")
    foglio_cckk = exc.create_sheet("coefficienti correlazione kk")

    #Impostazione della larghezza delle colonne
    lettere = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', 'AA', 'AB', 'AC', 'AD', 'AE']

    for col in range(1, 14):
        foglio_ris.column_dimensions[lettere[col]].width=20
    for col in range(0, 2):
        foglio_ccraw.column_dimensions[lettere[col]].width=16
        foglio_cck.column_dimensions[lettere[col]].width=16
        foglio_cckk.column_dimensions[lettere[col]].width=16
    for col in range(2, 31):
        foglio_ccraw.column_dimensions[lettere[col]].width=12
        foglio_cck.column_dimensions[lettere[col]].width=12
        foglio_cckk.column_dimensions[lettere[col]].width=12


    #Scrittura intestazioni delle colonne foglio risultati
    foglio_ris.cell(1, 2, 'CODICE SENSORE').style = stile1azzurro
    foglio_ris.cell(1, 3, 'MEDIA RAW' + '\n' + '[mrad]').style = stile1grigio
    foglio_ris.cell(1, 4, 'SQM RAW').style = 'stile1grigio'
    foglio_ris.cell(1, 5, 'SQM DIFFERENZA' + '\n' + 'RAW-MR({}h {}m)'.format(ore2, minuti2)).style = 'stile1grigio'
    foglio_ris.cell(1, 6, 'MEDIA K' + '\n' + '[mrad]').style = 'stile1grigio'
    foglio_ris.cell(1, 7, 'SQM K').style = 'stile1grigio'
    foglio_ris.cell(1, 8, 'SQM DIFFERENZA' + '\n' + 'K-MR({}h {}m)'.format(ore2, minuti2)).style = 'stile1grigio'
    foglio_ris.cell(1, 9, 'MEDIA KK' + '\n' + '[mrad]').style = 'stile1grigio'
    foglio_ris.cell(1, 10, 'SQM KK').style = 'stile1grigio'
    foglio_ris.cell(1, 11, 'SQM DIFFERENZA' + '\n' + 'KK-MR({}h {}m)'.format(ore2, minuti2)).style = 'stile1grigio'
    foglio_ris.cell(1, 12, 'DERIVA ISTANTANEA' + '\n' + '[mrad/°C]').style = 'stile1grigio'
    foglio_ris.cell(1, 13, 'DERIVA DIFFERITA' + '\n' + '[mrad/°C]').style = 'stile1grigio'
    foglio_ris.cell(1, 14, 'RITARDO per calcolo deriva differita').style = 'stile1grigio'


    #Unione celle in cui scrivere il tipo di rotazione (per rotazione X)
    foglio_ris.merge_cells(start_row=2, start_column=1, end_row=numero_sensori + 1, end_column=1)

    #Scrittura del tipo di rotazione (per 'Rotazioni X')
    foglio_ris.cell(2, 1, 'rotazioni X').alignment = Alignment(horizontal='center', vertical='center', textRotation=90)

    #Scrittura intestazioni delle colonne foglio delle rotazioni raw
    foglio_ccraw.cell(1, 1, 'CODICE SENSORE').style = 'stile1azzurro'
    foglio_ccraw.cell(1, 2).style = 'stile1grigio'
    foglio_ccraw.cell(1, 3, 'temp. istant.').style = stile1verde
    foglio_ccraw.cell(1, 4, 'MT (1h)').style = 'stile1verde'
    foglio_ccraw.cell(1, 5, 'MT (1h 20m)').style = 'stile1verde'
    foglio_ccraw.cell(1, 6, 'MT (1h 40m)').style = 'stile1verde'
    foglio_ccraw.cell(1, 7, 'MT (2h)' + '\n' + '(da -1h a 1h)').style = 'stile1verde'
    foglio_ccraw.cell(1, 8, 'MT (2h)' + '\n' + '(da -2h a 0h)').style = 'stile1verde'
    foglio_ccraw.cell(1, 9, 'MT (2h)' + '\n' + '(da -3h a 1h)').style = 'stile1verde'
    foglio_ccraw.cell(1, 10, 'MT (2h)' + '\n' + '(da -4h a 2h)').style = 'stile1verde'
    foglio_ccraw.cell(1, 11, 'MT (2h)' + '\n' + '(da -5h a 3h)').style = 'stile1verde'
    foglio_ccraw.cell(1, 12, 'MT (2h)' + '\n' + '(da -6h a 4h)').style = 'stile1verde'
    foglio_ccraw.cell(1, 13, 'MT (2h)' + '\n' + '(da -7h a 5h)').style = 'stile1verde'
    foglio_ccraw.cell(1, 14, 'MT (2h)' + '\n' + '(da -8h a 6h)').style = 'stile1verde'
    foglio_ccraw.cell(1, 15, 'MT (2h)' + '\n' + '(da -9h a 7h)').style = 'stile1verde'
```

```
87.     foglio_ccraw.cell(1, 16, 'MT (2h)' + '\n' + '(da -10h a 8h)').style = 'stile1verde'
88.     foglio_ccraw.cell(1, 17, 'MT (2h)' + '\n' + '(da -11h a 9h)').style = 'stile1verde'
89.     foglio_ccraw.cell(1, 18, 'MT (2h)' + '\n' + '(da -12h a 10h)').style = 'stile1verde'
90.     foglio_ccraw.cell(1, 19, 'MT (2h)' + '\n' + '(da -13h a 11h)').style = 'stile1verde'
91.     foglio_ccraw.cell(1, 20, 'MT (2h)' + '\n' + '(da -14h a 12h)').style = 'stile1verde'
92.     foglio_ccraw.cell(1, 21, 'MT (2h)' + '\n' + '(da -15h a 13h)').style = 'stile1verde'
93.     foglio_ccraw.cell(1, 22, 'MT (2h)' + '\n' + '(da -16h a 14h)').style = 'stile1verde'
94.     foglio_ccraw.cell(1, 23, 'MT (2h)' + '\n' + '(da -17h a 15h)').style = 'stile1verde'
95.     foglio_ccraw.cell(1, 24, 'MT (2h)' + '\n' + '(da -18h a 16h)').style = 'stile1verde'
96.     foglio_ccraw.cell(1, 25, 'MT (2h)' + '\n' + '(da -19h a 17h)').style = 'stile1verde'
97.     foglio_ccraw.cell(1, 26, 'MT (2h)' + '\n' + '(da -20h a 18h)').style = 'stile1verde'
98.     foglio_ccraw.cell(1, 27, 'MT (2h)' + '\n' + '(da -21h a 19h)').style = 'stile1verde'
99.     foglio_ccraw.cell(1, 28, 'MT (2h)' + '\n' + '(da -22h a 20h)').style = 'stile1verde'
100.    foglio_ccraw.cell(1, 29, 'MT (2h)' + '\n' + '(da -23h a 21h)').style = 'stile1verde'
101.    foglio_ccraw.cell(1, 30, 'MT (2h)' + '\n' + '(da -24h a 22h)').style = 'stile1verde'
102.    foglio_ccraw.cell(1, 31, 'MT (2h)' + '\n' + '(da -25h a 23h)').style = 'stile1verde'
103.
104.    #Scrittura intestazioni delle colonne foglio delle rotazioni raw
105.    foglio_cck.cell(1, 1, 'CODICE SENSORE').style = 'stile1azzurro'
106.    foglio_cck.cell(1, 2).style = 'stile1grigio'
107.    foglio_cck.cell(1, 3, 'temp. istant.').style = 'stile1verde'
108.    foglio_cck.cell(1, 4, 'MT (1h)').style = 'stile1verde'
109.    foglio_cck.cell(1, 5, 'MT (1h 20m)').style = 'stile1verde'
110.    foglio_cck.cell(1, 6, 'MT (1h 40m)').style = 'stile1verde'
111.    foglio_cck.cell(1, 7, 'MT (2h)' + '\n' + '(da -1h a 1h)').style = 'stile1verde'
112.    foglio_cck.cell(1, 8, 'MT (2h)' + '\n' + '(da -2h a 0h)').style = 'stile1verde'
113.    foglio_cck.cell(1, 9, 'MT (2h)' + '\n' + '(da -3h a 1h)').style = 'stile1verde'
114.    foglio_cck.cell(1, 10, 'MT (2h)' + '\n' + '(da -4h a 2h)').style = 'stile1verde'
115.    foglio_cck.cell(1, 11, 'MT (2h)' + '\n' + '(da -5h a 3h)').style = 'stile1verde'
116.    foglio_cck.cell(1, 12, 'MT (2h)' + '\n' + '(da -6h a 4h)').style = 'stile1verde'
117.    foglio_cck.cell(1, 13, 'MT (2h)' + '\n' + '(da -7h a 5h)').style = 'stile1verde'
118.    foglio_cck.cell(1, 14, 'MT (2h)' + '\n' + '(da -8h a 6h)').style = 'stile1verde'
119.    foglio_cck.cell(1, 15, 'MT (2h)' + '\n' + '(da -9h a 7h)').style = 'stile1verde'
120.    foglio_cck.cell(1, 16, 'MT (2h)' + '\n' + '(da -10h a 8h)').style = 'stile1verde'
121.    foglio_cck.cell(1, 17, 'MT (2h)' + '\n' + '(da -11h a 9h)').style = 'stile1verde'
122.    foglio_cck.cell(1, 18, 'MT (2h)' + '\n' + '(da -12h a 10h)').style = 'stile1verde'
123.    foglio_cck.cell(1, 19, 'MT (2h)' + '\n' + '(da -13h a 11h)').style = 'stile1verde'
124.    foglio_cck.cell(1, 20, 'MT (2h)' + '\n' + '(da -14h a 12h)').style = 'stile1verde'
125.    foglio_cck.cell(1, 21, 'MT (2h)' + '\n' + '(da -15h a 13h)').style = 'stile1verde'
126.    foglio_cck.cell(1, 22, 'MT (2h)' + '\n' + '(da -16h a 14h)').style = 'stile1verde'
127.    foglio_cck.cell(1, 23, 'MT (2h)' + '\n' + '(da -17h a 15h)').style = 'stile1verde'
128.    foglio_cck.cell(1, 24, 'MT (2h)' + '\n' + '(da -18h a 16h)').style = 'stile1verde'
129.    foglio_cck.cell(1, 25, 'MT (2h)' + '\n' + '(da -19h a 17h)').style = 'stile1verde'
130.    foglio_cck.cell(1, 26, 'MT (2h)' + '\n' + '(da -20h a 18h)').style = 'stile1verde'
131.    foglio_cck.cell(1, 27, 'MT (2h)' + '\n' + '(da -21h a 19h)').style = 'stile1verde'
132.    foglio_cck.cell(1, 28, 'MT (2h)' + '\n' + '(da -22h a 20h)').style = 'stile1verde'
133.    foglio_cck.cell(1, 29, 'MT (2h)' + '\n' + '(da -23h a 21h)').style = 'stile1verde'
134.    foglio_cck.cell(1, 30, 'MT (2h)' + '\n' + '(da -24h a 22h)').style = 'stile1verde'
135.    foglio_cck.cell(1, 31, 'MT (2h)' + '\n' + '(da -25h a 23h)').style = 'stile1verde'
136.
137.    #Scrittura intestazioni delle colonne foglio delle rotazioni raw
138.    foglio_cckk.cell(1, 1, 'CODICE SENSORE').style = 'stile1azzurro'
139.    foglio_cckk.cell(1, 2).style = 'stile1grigio'
140.    foglio_cckk.cell(1, 3, 'temp. istant.').style = 'stile1verde'
141.    foglio_cckk.cell(1, 4, 'MT (1h)').style = 'stile1verde'
142.    foglio_cckk.cell(1, 5, 'MT (1h 20m)').style = 'stile1verde'
143.    foglio_cckk.cell(1, 6, 'MT (1h 40m)').style = 'stile1verde'
144.    foglio_cckk.cell(1, 7, 'MT (2h)' + '\n' + '(da -1h a 1h)').style = 'stile1verde'
145.    foglio_cckk.cell(1, 8, 'MT (2h)' + '\n' + '(da -2h a 0h)').style = 'stile1verde'
146.    foglio_cckk.cell(1, 9, 'MT (2h)' + '\n' + '(da -3h a 1h)').style = 'stile1verde'
147.    foglio_cckk.cell(1, 10, 'MT (2h)' + '\n' + '(da -4h a 2h)').style = 'stile1verde'
148.    foglio_cckk.cell(1, 11, 'MT (2h)' + '\n' + '(da -5h a 3h)').style = 'stile1verde'
149.    foglio_cckk.cell(1, 12, 'MT (2h)' + '\n' + '(da -6h a 4h)').style = 'stile1verde'
150.    foglio_cckk.cell(1, 13, 'MT (2h)' + '\n' + '(da -7h a 5h)').style = 'stile1verde'
151.    foglio_cckk.cell(1, 14, 'MT (2h)' + '\n' + '(da -8h a 6h)').style = 'stile1verde'
152.    foglio_cckk.cell(1, 15, 'MT (2h)' + '\n' + '(da -9h a 7h)').style = 'stile1verde'
```

```
153.     foglio_cckk.cell(1, 16, 'MT (2h)' + '\n' + '(da -10h a 8h)').style = 'stile1verde'
154.     foglio_cckk.cell(1, 17, 'MT (2h)' + '\n' + '(da -11h a 9h)').style = 'stile1verde'
155.     foglio_cckk.cell(1, 18, 'MT (2h)' + '\n' + '(da -12h a 10h)').style = 'stile1verde'
156.     foglio_cckk.cell(1, 19, 'MT (2h)' + '\n' + '(da -13h a 11h)').style = 'stile1verde'
157.     foglio_cckk.cell(1, 20, 'MT (2h)' + '\n' + '(da -14h a 12h)').style = 'stile1verde'
158.     foglio_cckk.cell(1, 21, 'MT (2h)' + '\n' + '(da -15h a 13h)').style = 'stile1verde'
159.     foglio_cckk.cell(1, 22, 'MT (2h)' + '\n' + '(da -16h a 14h)').style = 'stile1verde'
160.     foglio_cckk.cell(1, 23, 'MT (2h)' + '\n' + '(da -17h a 15h)').style = 'stile1verde'
161.     foglio_cckk.cell(1, 24, 'MT (2h)' + '\n' + '(da -18h a 16h)').style = 'stile1verde'
162.     foglio_cckk.cell(1, 25, 'MT (2h)' + '\n' + '(da -19h a 17h)').style = 'stile1verde'
163.     foglio_cckk.cell(1, 26, 'MT (2h)' + '\n' + '(da -20h a 18h)').style = 'stile1verde'
164.     foglio_cckk.cell(1, 27, 'MT (2h)' + '\n' + '(da -21h a 19h)').style = 'stile1verde'
165.     foglio_cckk.cell(1, 28, 'MT (2h)' + '\n' + '(da -22h a 20h)').style = 'stile1verde'
166.     foglio_cckk.cell(1, 29, 'MT (2h)' + '\n' + '(da -23h a 21h)').style = 'stile1verde'
167.     foglio_cckk.cell(1, 30, 'MT (2h)' + '\n' + '(da -24h a 22h)').style = 'stile1verde'
168.     foglio_cckk.cell(1, 31, 'MT (2h)' + '\n' + '(da -25h a 23h)').style = 'stile1verde'
169. 
170.     #Scrittura delle intestazioni delle righe (rotazioni su cui si calcoleranno i coefficienti di correlazione) nel foglio de
     lle rotazioni raw per rotazione X
171.     foglio_ccraw.cell(2,2, 'rot. istant.').style = stile1grigio
172.     foglio_ccraw.cell(3,2, 'MR({} h {} min)'.format(ore1, minuti1)).style = 'stile1grigio'
173.     foglio_ccraw.cell(4,2, 'MR({} h {} min)'.format(ore2, minuti2)).style = 'stile1grigio'
174.     foglio_ccraw.cell(5,2, 'MR({} h {} min)'.format(ore3, minuti3)).style = 'stile1grigio'
175. 
176.     #Scrittura delle intestazioni delle righe (rotazioni su cui si calcoleranno i coefficienti di correlazione) nel foglio de
     lle rotazioni k per rotazione X
177.     foglio_cck.cell(2,2, 'rot. istant. k').style = 'stile1grigio'
178.     foglio_cck.cell(3,2, 'MR_k({} h {} min)'.format(ore1, minuti1)).style = 'stile1grigio'
179.     foglio_cck.cell(4,2, 'MR_k({} h {} min)'.format(ore2, minuti2)).style = 'stile1grigio'
180.     foglio_cck.cell(5,2, 'MR_k({} h {} min)'.format(ore3, minuti3)).style = 'stile1grigio'
181. 
182.     #Scrittura delle intestazioni delle righe (rotazioni su cui si calcoleranno i coefficienti di correlazione) nel foglio de
     lle rotazioni kk per rotazione X
183.     foglio_cckk.cell(2,2, 'rot. istant. kk').style = 'stile1grigio'
184.     foglio_cckk.cell(3,2, 'MR_kk({} h {} min)'.format(ore1, minuti1)).style = 'stile1grigio'
185.     foglio_cckk.cell(4,2, 'MR_kk({} h {} min)'.format(ore2, minuti2)).style = 'stile1grigio'
186.     foglio_cckk.cell(5,2, 'MR_kk({} h {} min)'.format(ore3, minuti3)).style = 'stile1grigio'
187. 
188.     #Unione celle in cui scrivere il nome del sensore nel foglio delle rotazioni raw per rotazione X
189.     foglio_ccraw.merge_cells(start_row=2, start_column=1, end_row=5, end_column=1)
190. 
191.     #Unione celle in cui scrivere il nome del sensore nel foglio delle rotazioni k per rotazione X
192.     foglio_cck.merge_cells(start_row=2, start_column=1, end_row=5, end_column=1)
193. 
194.     #Unione celle in cui scrivere il nome del sensore nel foglio delle rotazioni kk per rotazione X
195.     foglio_cckk.merge_cells(start_row=2, start_column=1, end_row=5, end_column=1)
196. 
197.     if tipo_sensore == 'sensoreA':
198. 
199.         #Unione celle in cui scrivere il tipo di rotazione (per rotazione Y)
200.         foglio_ris.merge_cells(start_row=numero_sensori + 4, start_column=1, end_row = 2 * numero_sensori +
     3, end_column=1)
201. 
202.         #Scrittura del tipo di rotazione ('Rotazioni Y' )
203.         foglio_ris.cell(numero_sensori + 4, 1, 'rotazioni Y').alignment = Alignment(horizontal='center', ver
     tical='center', textRotation=90)
204. 
205.         #Unione celle in cui scrivere il nome del file nel foglio 1 per rotazione Y
206.         foglio_ccraw.merge_cells(start_row=4*numero_sensori+4, start_column=1, end_row = 4*numero_sensori+7,
      end_column=1)
207. 
208.         #Unione celle in cui scrivere il nome del file nel foglio 2 per rotazione Y
209.         foglio_cck.merge_cells(start_row=4*numero_sensori+4, start_column=1, end_row = 4*numero_sensori+7, e
     nd_column=1)
210. 
211.         #Unione celle in cui scrivere il nome del file nel foglio 3 per rotazione Y
```

```
        foglio_cckk.merge_cells(start_row=4*numero_sensori+4, start_column=1, end_row = 4*numero_sensori+7, end_column=1)

    return exc, foglio_ris, foglio_ccraw, foglio_cck,foglio_cckk
```

Figura 56: Modulo esterno Excel, funzione Excel.crea_layout

La seconda funzione richiamata è Excel.crea, la quale ha come argomenti:

- l'indice del sensore analizzato nel ciclo attuale (*i*),
- il tipo del sensore analizzato (*tipo_sensore*);
- il numero di sensori totali (*numero_sensori*);
- il nome del sensore analizzato nel ciclo attuale (*nome_file[i]*);
- i vari fogli aperti del file Excel dei risultati (*foglio_ris* , *foglio_ccraw*, *foglio_cck*, *foglio_cckk*).

Essa scrive nel file Excel creato il nome del sensore corrente alla giusta riga, la stessa in cui poi si scriveranno i risultati relativi.

```
#FUNZIONE EXCEL(2) --> Questa funzione scrive sul file Excel dei risultati i nomi dei sensori
def crea(i, tipo_sensore, numero_sensori, nome_file, foglio_ris, foglio_ccraw, foglio_cck, foglio_cckk):

    #Scrittura nome i-esimo sensore per rotazione X
    foglio_ris.cell(i + 2, 2, nome_file).style = 'stile1azzurro'

    if i>0:

        #Unione celle in cui scrivere il nome del sensore nel foglio delle rotazioni raw per rotazione X
        foglio_ccraw.merge_cells(start_row=4*i+2, start_column=1, end_row=4*i+5, end_column=1)

        #Unione celle in cui scrivere il nome del sensore nel foglio delle rotazioni k per rotazione X
        foglio_cck.merge_cells(start_row=4*i+2, start_column=1, end_row=4*i+5, end_column=1)

        #Unione celle in cui scrivere il nome del sensore nel foglio delle rotazioni kk per rotazione X
        foglio_cckk.merge_cells(start_row=4*i+2, start_column=1, end_row=4*i+5, end_column=1)

        for j in range(4):

            #Scrittura delle intestazioni delle righe (rotazioni su cui si calcoleranno i coefficienti di correlazione) nel foglio delle rotazioni raw per rotazione X
            foglio_ccraw.cell(4*i+2+j, 2, foglio_ccraw.cell(2+j,2).value).style = 'stile1grigio'

            #Scrittura delle intestazioni delle righe (rotazioni su cui si calcoleranno i coefficienti di correlazione) nel foglio delle rotazioni k per rotazione X
            foglio_cck.cell(4*i+2+j, 2, foglio_cck.cell(2+j,2).value).style = 'stile1grigio'

            #Scrittura delle intestazioni delle righe (rotazioni su cui si calcoleranno i coefficienti di correlazione) nel foglio delle rotazioni kk per rotazione X
            foglio_cckk.cell(4*i+2+j, 2, foglio_cckk.cell(2+j,2).value).style = 'stile1grigio'


    #Scrittura del nome del sensore nel foglio 1 per rotazione X
    foglio_ccraw.cell(4*i+2, 1, nome_file).style = 'stile1azzurro'

    #Scrittura del nome del sensore nel foglio 2 per rotazione X
    foglio_cck.cell(4*i+2, 1, nome_file).style = 'stile1azzurro'

    #Scrittura del nome del sensore nel foglio 3 per rotazione X
```

```
37.     foglio_cckk.cell(4*i+2, 1, nome_file).style = 'stile1azzurro'
38.
39.     if tipo_sensore == 'sensoreA':
40.
41.         #Scrittura nome i-esimo sensore per rotazione Y
42.         foglio_ris.cell(numero_sensori + i + 4, 2, nome_file).style = 'stile1azzurro'
43.
44.         if i>=0:
45.
46.             #Unione celle in cui scrivere il nome del file nel foglio 1 per rotazione Y
47.             foglio_ccraw.merge_cells(start_row=4*numero_sensori+4+i*5, start_column=1, end_row = 4*numero_se
    nsori+7+i*4, end_column=1)
48.
49.             #Unione celle in cui scrivere il nome del file nel foglio 2 per rotazione Y
50.             foglio_cck.merge_cells(start_row=4*numero_sensori+4+i*5, start_column=1, end_row = 4*numero_sens
    ori+7+i*4, end_column=1)
51.
52.             #Unione celle in cui scrivere il nome del file nel foglio 3 per rotazione Y
53.             foglio_cckk.merge_cells(start_row=4*numero_sensori+4+i*5, start_column=1, end_row = 4*numero_sen
    sori+7+i*4, end_column=1)
54.
55.             for j in range(4):
56.
57.                #Scrittura delle rotazioni su cui si calcoleranno i coefficienti di correlazione nel foglio 1 per rotazione Y
58.                 foglio_ccraw.cell(4*numero_sensori+6+i*4+j, 2, foglio_ccraw.cell(2+j,2).value).style = 'stil
    e1grigio'
59.
60.                 #Scrittura delle rotazioni su cui si calcoleranno i coefficienti di correlazione nel foglio 2 per rotazione
     Y
61.                 foglio_cck.cell(4*numero_sensori+6+i*4+j, 2, foglio_cck.cell(2+j,2).value).style = 'stile1gr
    igio'
62.
63.                 #Scrittura delle rotazioni su cui si calcoleranno i coefficienti di correlazione nel foglio 3 per rotazione
     Y
64.                 foglio_cckk.cell(4*numero_sensori+6+i*4+j, 2, foglio_cckk.cell(2+j,2).value).style = 'stile1
    grigio'
65.
66.         #Scrittura del nome del file nel foglio 1 per rotazione Y
67.         foglio_ccraw.cell(numero_sensori*4+4*i+4, 1, nome_file).style = 'stile1azzurro'
68.
69.         #Scrittura del nome del file nel foglio 2 per rotazione Y
70.         foglio_cck.cell(numero_sensori*4+4*i+4, 1, nome_file).style = 'stile1azzurro'
71.
72.         #Scrittura del nome del file nel foglio 3 per rotazione Y
73.         foglio_cckk.cell(numero_sensori*4+4*i+4, 1, nome_file).style = 'stile1azzurro'
```

Figura 57: Modulo esterno Excel, funzione Excel.crea

L'insieme successivo di istruzioni è utilizzato per acquisire i dati di rotazione e temperatura a ponte scarico; come anticipato nel paragrafo 4.4, queste informazioni sono contenute nel file "dati_inizialiB.csv", in cui la riga i-esima è formata da tre elementi: il nome del sensore i-esimo, la sua misurazione di rotazione ponte scarico e la temperatura corrispondente. Il formato del file è tale per cui si possono importare nel codice direttamente dei vettori che rappresentano le colonne del file in formato Comma

Separated Value (csv). In particolare, quindi, si genera il vettore *rot0*, in cui l'i-esimo elemento rappresenta la rotazione a ponte scarico misurata dall'i-esimo sensore, e *temp0*, in cui l l'i-esimo elemento rappresenta la temperatura corrispondente alla misurazione di ponte scarico percepita dall'i-esimo sensore.

Innanzitutto, quindi, si deve importare il file di input usando il comando pandas.read_csv, il quale richiede come argomento il percorso del file csv da aprire; poi, i singoli vettori vengono creati richiamandone l'intestazione, ovvero il valore che viene scritto sulla prima riga del file di input.

```
1.  # 8.5 Acquisizione da file dei dati a ponte scarico
2.
3.  if tipo_sensore == 'sensoreA':
4.
5.      file0 = pd.read_csv(os.getcwd()+"\dati_iniziali_A.csv")              #Lettura del file contenente i dati a pont
    e scarico per i sensori A
6.
7.      rot0Y = np.array(file0['rotazione ponte scarico Y[mrad]'])[i]       #Vettore delle rotazioni Y a ponte scarico
    per il sensore di tipo A
8.      umid0 = np.array(file0['umidità ponte scarico[%]'])[i]              #Vettore delle umidità a ponte scarico per
    il sensore di tipo A
9.
10. if tipo_sensore == 'sensoreB':
11.
12.     file0 = pd.read_csv(os.getcwd()+"\dati_iniziali_B.csv")              #Lettura del file contenente i dati a pont
    e scarico per i sensori B
13.
14. rot0X = np.array(file0['rotazione ponte scarico X[mrad]'])[i]           #Vettore delle rotazioni X a ponte scarico
    per entrambi i sensori
15. temp0 = np.array(file0['temperatura ponte scarico[°C]'])[i]             #Vettore delle temperature a ponte scarico
    per entrambi i sensori
```

Figura 58: Blocco (8.5) del codice

I blocchi (8.6), (8.7) e (8.8) sono utili più che altro per sensori di tipo A, in quanto permettono di definire quante volte vadano ripetute le righe di codice successive, ovvero rispetto a quante grandezze vadano svincolati i dati di rotazione. Infatti, nel caso di sensori di tipo B, per cui lo svincolamento deve avvenire solamente rispetto ai dati di temperatura, la parte seguente di codice verrà letta solo una volta, mentre per sensori di tipo A essa verrà eseguita due volte: il primo ciclo permette di rendere indipendenti i dati dalla temperatura, e il secondo dall'umidità. In relazione a ciò, sono inseriti in questi blocchi anche i comandi che permettono di eliminare valori non numerici (Not A Number, oppure nan) dalle serie di rotazioni grezze; questo problema non sussiste al primo ciclo, ma al secondo: infatti, la serie finale del primo ciclo corretta dalle temperature, che

presenta valori non numerici nei suoi primi elementi, rappresenta la serie grezza per il secondo ciclo.

La parte di codice seguente, dunque, è stata inserita in un ciclo che si ripete tante volte quante sono le grandezze da cui svincolare i dati. Per sensori di tipo B, la variabile che indica quale ciclo si stia percorrendo (*cicli_pulizia*), è sempre pari a 0.

4.8.2. Blocco (9): Calcoli sulle rotazioni raw

Questo blocco del codice calcola alcuni parametri statistici relativi alla serie di rotazioni grezze, quali la media e lo scarto quadratico medio, poi calcola tre serie di medie mobili centrate sui tre intervalli di tempo inseriti in input (nel caso in esame, 2 ore e 20 minuti, 3 ore, 3 ore e 40 minuti), e calcola la serie delle differenze tra la serie originale e la serie di medie mobili centrate sul secondo intervallo temporale definito, oltre che lo scarto quadratico medio di quest'ultima serie. Infine, sono presenti le istruzioni per scrivere i dati significativi ricavati (media, scarto quadratico medio della serie raw e scarto quadratico medio della serie delle differenze) sul file Excel dei risultati.

Per quanto riguarda la prima parte, ovvero per tutte le operazioni di calcolo, viene richiamata la funzione Statistica.calcoli_rotazioni, che, come si capisce, appartiene al modulo Statistica. Essa riceve in input la serie di rotazioni grezze, e il numero di valori contenuti nel primo, nel secondo e nel terzo intervallo temporale su cui si vogliono calcolare le medie mobili. L'output di tale funzione è costituito da scalari e da vettori:

- media (rot_medX);
- scarto quadratico medio (*sqmX*);
- differenza tra serie originale e serie di medie mobili centrate calcolate sul secondo intervallo temporale (*diff_mr2X*);
- scarto quadratico medio della serie al punto precedente (*sqm_diff*);
- serie di medie mobili centrate calcolate sul primo intervallo di valori definito dall'utente (*mr1X*);
- serie di medie mobili centrate calcolate sul secondo intervallo di valori definito dall'utente (*mr2X*);
- serie di medie mobili centrate calcolate sul terzo intervallo di valori definito dall'utente (*mr3X*).

Innanzitutto, viene calcolato il numero di valori che devono essere presenti a sinistra e a destra del valore centrale dell'intervallo di valori con cui si calcolano le medie mobili, per il primo, il secondo e il terzo intervallo temporale definiti dall'utente (*sinistra1*, *sinistra2*, *sinistra3*, *destra1*, *destra2*, *destra3*). Vengono usate formulazioni diverse se il numero totale di valori contenuti nell'intervallo (*val1*, *val2*, *val3*) è pari oppure dispari.
Si ricalcola la lunghezza della serie immessa come argomento (*N*, variabile interna alla funzione Statistica.calcoli_rotazioni), e il numero di Not A Number (nan) presenti in essa (saranno comunque concentrati all'inizio)(*n_nan*).
Il valor medio e lo scarto quadratico medio vengono calcolati rispettivamente con i metodi mean e std, entrambi applicabili ad elementi di classe "numpy.array".
I primi elementi delle serie di medie mobili , per definizione, non possono essere calcolati in quanto la finestra di valori su cui si calcola la media mobile deve iniziare, almeno, dal primo elemento della serie di dati grezzi. Essendo la finestra centrata, sono presenti un numero di Not A Number esattamente pari al numero di valori presenti a sinistra del valore centrale della finestra. Dunque, per definire le serie di medie mobili, si calcolano prima di tutto le serie a partire dal loro primo valore non nullo, grazie al metodo mean applicato al giusto intervallo di valori, poi si genera un vettore di Not A Number di lunghezza pari a *sinistra1* (o *sinistra2* o *sinistra3*) tramite la funzione numpy.full, inserendo come argomenti la lunghezza e il valore con cui si vuole riempire l'array, ed infine questi due vettori vengono allineati con la funzione numpy.concatenate.

```
1.  #FUNZIONE STATISTICA(1) --> Questa funzione calcola i parametri statistici delle rotazioni
2.  def calcoli_rotazioni(x, val1, val2, val3):
3.
4.      if val1%2 == 0:
5.          sinistra1 = int( val1 / 2 )
6.          destra1 = int( val1 / 2 + 1 )
7.      else:
8.          sinistra1 = int( val1 /2 - 0.5 )
9.          destra1 = int( val1 / 2 + 0.5 )
10.
11.     if val2%2 == 0:
12.         sinistra2 = int( val2 / 2 )
13.         destra2 = int( val2 / 2 + 1 )
14.     else:
15.         sinistra2 = int( val2 /2 - 0.5 )
16.         destra2 = int( val2 / 2 + 0.5 )
17.
18.     if val3%2 == 0:
19.         sinistra3 = int( val3 / 2 )
20.         destra3 = int( val3 / 2 + 1 )
21.     else:
22.         sinistra3 = int( val3 /2 - 0.5 )
23.         destra3 = int( val3 / 2 + 0.5 )
24.
25.     N = len(x)                                                    #Lunghezza vettore
```

```
26.     n_nan = np.isnan(x)[np.isnan(x)== True].size                    #Conteggio dei nan nel vettore
27.
28.     #Calcolo valore medio delle rotazioni da dopo i nan (raw, pulite o doppiamente pulite)
29.
30.     x_med=x[n_nan:].mean()
31.
32.     #Calcolo scarto quadratico medio delle rotazioni da dopo i nan (raw, pulite o doppiamente pulite)
33.
34.     sqm=x[n_nan:].std()
35.
36.     #Calcolo media mobile su val1 misure delle rotazioni da dopo i nan (raw, pulite o doppiamente pulite)
37.
38.     mr1=np.array([x[i-
    sinistra1:i+destra1].mean() for i in range(n_nan+sinistra1,N)])          #Medie dal primo valore non nan
39.     a=np.full(n_nan+sinistra1, np.nan)                                   #Creazione del vettore dei primi nan
40.     mr1=np.concatenate([a, mr1])                                         #Unione dei due vettori
41.
42.     #Calcolo media mobile su val2 misure delle rotazioni da dopo i nan (raw, pulite o doppiamente pulite)
43.
44.     mr2=np.array([x[i-
    sinistra2:i+destra2].mean() for i in range(n_nan+sinistra2,N)])          #Medie dal primo valore non nan
45.     a=np.full(n_nan+sinistra2, np.nan)                                   #Creazione del vettore dei primi nan
46.     mr2=np.concatenate([a, mr2])                                         #Unione dei due vettori
47.
48.     #Calcolo media mobile su val3 misure delle rotazioni da dopo i nan (raw, pulite o doppiamente pulite)
49.
50.     mr3=np.array([x[i-
    sinistra3:i+destra3].mean() for i in range(n_nan+sinistra3,N)])          #Medie dal primo valore non nan
51.     a=np.full(n_nan+sinistra3, np.nan)                                   #Creazione del vettore dei primi nan
52.     mr3=np.concatenate([a, mr3])                                         #Unione dei due vettori
53.
54.     #Calcolo differenza tra segnale originale e media mobile sul secondo intervallo delle rotazioni da dopo i nan (raw, pulit
    e o doppiamente pulite)
55.
56.     diff_mr2=x-mr2
57.
58.     #Calcolo scarto quadratico medio della differenza calcolata
59.
60.     sqm_diff=diff_mr2[n_nan:].std()
61.
62.     #Restituzione delle grandezze calcolate
63.
64.     return x_med, sqm, diff_mr2, sqm_diff, mr2, mr1, mr3
```

Figura 59: Modulo esterno Statistica, funzione Statistica.calcoli_rotazioni

Per quanto riguarda invece il salvataggio di dati sul file Excel aperto nel blocco (8), esaminato al paragrafo 4.8.1, esso avviene attraverso un metodo appartenente al modulo che permette di gestire file Excel, Openpyxl. Il metodo è cell, richiede in argomento la riga e la colonna della cella da individuare, e facoltativamente il valore da scriverci; questo è un metodo applicabile alle variabili con cui sono stati individuati i fogli del file Excel dei risultati.

```
1.  #   (9) CALCOLI SULLE ROTAZIONI RAW
2.
3.  import Statistica   #Importazione del modulo esterno che si occupa del calcolo dei parametri necessari alla scorrelazione
```

```
4.
5.  #Calcolo di parametri statistici e di serie di medie mobili per la rotazione X nel blocco esterno per entrambi i sensori
6.  rot_medX, sqmX, diff_mr2X, sqm_diffX, mr1X, mr2X, mr3X = Statistica.calcoli_rotazioni(rotX, val1, val2, val3
    )
7.
8.  #Salvataggio dei valori calcolati nel file Excel precedemente aperto per le grandezze comuni a entrambi i sensori
9.
10. foglio_ris.cell(i+2, 3, round(rot_medX, 3)).style = stile2     #Media delle rotazioni X
11. foglio_ris.cell(i+2, 4, round(sqmX, 3)).style = 'stile2'       #Scarto quadratico medio delle rotazioni X
12. foglio_ris.cell(i+2, 5, round(sqm_diffX, 3)).style = 'stile2'  #Scarto quadratico medio della differenza tra la seri
    e di rotazioni X istantanee e la serie di medie mobili calcolata sull'intervallo 2
13.
14.
15. if tipo_sensore == 'sensoreA':
16.
17.     #Calcolo di parametri statistici e di serie di medie mobili per la rotazione Y nel blocco esterno per sensori di tipo A
18.     rot_medY, sqmY, diff_mr2Y, sqm_diffY, mr1Y, mr2Y, mr3Y = Statistica.calcoli_rotazioni(rotY, val1, val2,
    val3)
19.
20.     #Salvataggio dei valori calcolati nel file Excel precedemente aperto (Foglio 1) per le grandezze proprie del sensore di t
    ipo A
21.
22.     foglio_ris.cell(numero_sensori + i + 4, 3, round(rot_medY, 3)).style = 'stile2'      #Media delle rotazioni
     Y
23.     foglio_ris.cell(numero_sensori + i + 4, 4, round(sqmY, 3)).style = 'stile2'          #Scarto quadratico med
    io delle rotazioni Y
24.     foglio_ris.cell(numero_sensori + i + 4, 5, round(sqm_diffY, 3)).style = 'stile2'     #Scarto quadratico med
    io della differenza tra la serie di rotazioni Y istantanee e la serie di medie mobili calcolata sull'intervallo 2
```

Figura 60: Blocco (9) del codice

4.8.3. Blocco (10): Calcoli su temperatura e umidità

Questa parte di codice è finalizzata a generare le serie di temperature dette al capitolo 3.1:

- media mobile su 1 ora antecedente;
- media mobile su 1 ora e 20 minuti antecedenti;
- media mobile su 1 ora e 40 minuti antecedenti;
- media mobile su 2 ore antecedenti;
- media mobile su 2 ore antecedenti, calcolate con ritardi del valore centrale da 0 a 24 ore rispetto all'istante considerato.

```
1.  #   (10) CALCOLI SU UMIDITA' E TEMPERATURA
2.
3.  if tipo_sensore == 'sensoreA':   #Se il sensore è di tipo A, vengono eseguite due scorrelazioni
4.
5.      if cicli_pulizia == 0:       #Al primo ciclo, si calcolano i parametri per la scorrelazione rispetto alla temperatura
6.
7.          #Calcolo di medie mobili sulla temperatura nel blocco esterno per sensori A
8.          mt3, mt4, mt5, mt6, mt6_r1, mt6_r2, mt6_r3, mt6_r4, mt6_r5, mt6_r6, mt6_r7, mt6_r8, mt6_r9, mt6_r10,
     mt6_r11, mt6_r12, mt6_r13, mt6_r14, mt6_r15, mt6_r16, mt6_r17, mt6_r18, mt6_r19, mt6_r20, mt6_r21, mt6_r22,
     mt6_r23 = Statistica.calcoli_temp_umidA(temp)
9.
10.     if cicli_pulizia == 1:         #Al secondo ciclo, si calcolano i parametri per la scorrelazione rispetto all'umidità
11.
12.         #Calcolo di medie mobili sull'umidità nel blocco esterno per sensori A
```

```
13.          mu3, mu4, mu5, mu6, mu6_r1, mu6_r2, mu6_r3, mu6_r4, mu6_r5, mu6_r6, mu6_r7, mu6_r8, mu6_r9, mu6_r10,
     mu6_r11, mu6_r12, mu6_r13, mu6_r14, mu6_r15, mu6_r16, mu6_r17, mu6_r18, mu6_r19, mu6_r20, mu6_r21, mu6_r22,
     mu6_r23 = Statistica.calcoli_temp_umidA(umid)
14.
15.  if tipo_sensore == 'sensoreB':    #Se il sensore è di tipo B, viene eseguita una scorrelazione
16.
17.      #Calcolo di medie mobili sulla temperatura nel blocco esterno per sensori B
18.      mt3, mt4, mt5, mt6, mt6_r1, mt6_r2, mt6_r3, mt6_r4, mt6_r5, mt6_r6, mt6_r7, mt6_r8, mt6_r9, mt6_r10, mt6
     _r11, mt6_r12, mt6_r13, mt6_r14, mt6_r15, mt6_r16, mt6_r17, mt6_r18, mt6_r19, mt6_r20, mt6_r21, mt6_r22, mt6
     _r23 = Statistica.calcoli_temp_umidB(temp)
```

Figura 61: Blocco (10) del codice

Queste vengono calcolate per mezzo della funzione Statistica.calcoli_temp_umidB, la quale è usata solo per sensori di tipo B (ma esiste il suo corrispettivo per sensori di tipo A), in quanto le medie mobili vengono calcolate su un numero di valori coerente con il tempo di campionamento dei sensori B (ad esempio, essendo il tempo di campionamento pari a 20 minuti, la media mobile su un'ora viene calcolata su 3 valori; ciò non è vero in generale).

L'unico argomento fornito in input alla funzione è la serie *temp_umid*, sempre costituita da temperature per sensori di tipo B.

Siccome la finestra temporale su cui andare a calcolare la media è anteriore all'istante considerato, il primo valore della serie delle medie diverso da Not A Number è quello subito successivo a una lunghezza pari a tale finestra. Perciò, dapprima vengono creati i vettori a partire dal loro primo elemento diverso da Not A Number, grazie al metodo mean applicato al giusto intervallo di valori , poi si crea un vettore provvisorio contenente il giusto numero di Not A Number (pari al numero di valori contenuti nella finestra temporale scelta) tramite la funzione numpy.full, inserendo come argomenti la lunghezza e il valore con cui si vuole riempire l'array, e infine questi due vettori vengono allineati con la funzione numpy.concatenate. Il numero di Not A Number aumenta nel caso di medie ritardate.

```
1.  #FUNZIONE STATISTICA(2) --
    > Questa funzione calcola i parametri di temperatura necessari per il calcolo delle derive per il sensore di
     tipo B
2.  def calcoli_temp_umidB(temp_umid):
3.
4.
5.
6.      #Calcolo media mobile su 3 misure precedenti di temperature da dopo i nan (raw, pulite o doppiamente pulite) del sensore
    B
7.
```

```
8.      mt3=np.array([temp_umid[i-
    3:i].mean() for i in range(3,N)])                    #Medie dal primo valore non nan
9.      a=np.full(3, np.nan)                             #Creazione del vettore dei primi nan
10.     mt3=np.concatenate([a, mt3])                     #Unione dei due vettori
11.
12.     #Calcolo media mobile su 4 misure precedenti di temperature da dopo i nan (raw, pulite o doppiamente pulite) del sensore
    B
13.
14.     mt4=np.array([temp_umid[i-
    4:i].mean() for i in range(4,N)])                    #Medie dal primo valore non nan
15.     a=np.full(4, np.nan)                             #Creazione del vettore dei primi nan
16.     mt4=np.concatenate([a, mt4])                     #Unione dei due vettori
17.
18.     #Calcolo media mobile su 5 misure precedenti di temperature da dopo i nan (raw, pulite o doppiamente pulite) del sensore
    B
19.
20.     mt5=np.array([temp_umid[i-
    5:i].mean() for i in range(5,N)])                    #Medie dal primo valore non nan
21.     a=np.full(5, np.nan)                             #Creazione del vettore dei primi nan
22.     mt5=np.concatenate([a, mt5])                     #Unione dei due vettori
23.
24.     #Calcolo media mobile su 7 misure precedenti di temperature da dopo i nan (raw, pulite o doppiamente pulite) del sensore
    B
25.
26.     mt7C=np.array([temp_umid[i-
    3:i+4].mean() for i in range(3,N)])                  #Medie dal primo valore non nan
27.     a=np.full(3, np.nan)                             #Creazione del vettore dei primi nan
28.     mt7C=np.concatenate([a, mt7C])                   #Unione dei due vettori
29.
30.     #Calcolo media mobile su 6 misure prima con ritardo 0h (da 2h a 0h prima) di temperature dopo i nan (raw, pulite o doppia
    mente pulite) del sensore B
31.
32.     mt6=np.array([temp_umid[i-
    6:i].mean() for i in range(6,N)])                    #Medie dal primo valore non nan
33.     a=np.full(6, np.nan)                             #Creazione del vettore dei primi nan
34.     mt6=np.concatenate([a, mt6])                     #Unione dei due vettori
35.
36.     #Calcolo media mobile su 6 misure prima con ritardo 1h (da 3h a 1h prima) di temperature dopo i nan (raw, pulite o doppia
    mente pulite) del sensore B
37.
38.     mt6_r1=np.array([temp_umid[i-9:i-
    4].mean() for i in range(9,N)])                      #Medie dal primo valore non nan
39.     a=np.full(9, np.nan)                             #Creazione del vettore dei primi nan
40.     mt6_r1=np.concatenate([a, mt6_r1])               #Unione dei due vettori
41.
42.     #Calcolo media mobile su 6 misure prima con ritardo 2h (da 4h a 2h prima) di temperature dopo i nan (raw, pulite o doppia
    mente pulite) del sensore B
43.
44.     mt6_r2=np.array([temp_umid[i-12:i-
    7].mean() for i in range(12,N)])                     #Medie dal primo valore non nan
45.     a=np.full(12, np.nan)                            #Creazione del vettore dei primi nan
46.     mt6_r2=np.concatenate([a, mt6_r2])               #Unione dei due vettori
47.
48.     #Calcolo media mobile su 6 misure prima con ritardo 3h (da 5h a 3h prima) di temperature dopo i nan (raw, pulite o doppia
    mente pulite) del sensore B
49.
50.     mt6_r3=np.array([temp_umid[i-15:i-
    10].mean() for i in range(15,N)])                     #Medie dal primo valore non nan
51.     a=np.full(15, np.nan)                            #Creazione del vettore dei primi nan
52.     mt6_r3=np.concatenate([a, mt6_r3])               #Unione dei due vettori
53.
54.     #Calcolo media mobile su 6 misure prima con ritardo 4h (da 6h a 4h prima) di temperature dopo i nan (raw, pulite o doppia
    mente pulite) del sensore B
55.
56.     mt6_r4=np.array([temp_umid[i-18:i-
    13].mean() for i in range(18,N)])                     #Medie dal primo valore non nan
```

```
57.     a=np.full(18, np.nan)                                #Creazione del vettore dei primi nan
58.     mt6_r4=np.concatenate([a, mt6_r4])                   #Unione dei due vettori
59.
60.     #Calcolo media mobile su 6 misure prima con ritardo 5h (da 7h a 5h prima) di temperature dopo i nan (raw, pulite o doppia
    mente pulite) del sensore B
61.
62.     mt6_r5=np.array([temp_umid[i-21:i-
    16].mean() for i in range(21,N)])                        #Medie dal primo valore non nan
63.     a=np.full(21, np.nan)                                #Creazione del vettore dei primi nan
64.     mt6_r5=np.concatenate([a, mt6_r5])                   #Unione dei due vettori
65.
66.     #Calcolo media mobile su 6 misure prima con ritardo 6h (da 8h a 6h prima) di temperature dopo i nan (raw, pulite o doppia
    mente pulite) del sensore B
67.
68.     mt6_r6=np.array([temp_umid[i-24:i-
    19].mean() for i in range(24,N)])                        #Medie dal primo valore non nan
69.     a=np.full(24, np.nan)                                #Creazione del vettore dei primi nan
70.     mt6_r6=np.concatenate([a, mt6_r6])                   #Unione dei due vettori
71.
72.     #Calcolo media mobile su 6 misure prima con ritardo 7h (da 9h a 7h prima) di temperature dopo i nan (raw, pulite o doppia
    mente pulite) del sensore B
73.
74.     mt6_r7=np.array([temp_umid[i-27:i-
    22].mean() for i in range(27,N)])                        #Medie dal primo valore non nan
75.     a=np.full(27, np.nan)                                #Creazione del vettore dei primi nan
76.     mt6_r7=np.concatenate([a, mt6_r7])                   #Unione dei due vettori
77.
78.     #Calcolo media mobile su 6 misure prima con ritardo 8h (da 10h a 8h prima) di temperature dopo i nan (raw, pulite o doppi
    amente pulite) del sensore B
79.
80.     mt6_r8=np.array([temp_umid[i-30:i-
    25].mean() for i in range(30,N)])                        #Medie dal primo valore non nan
81.     a=np.full(30, np.nan)                                #Creazione del vettore dei primi nan
82.     mt6_r8=np.concatenate([a, mt6_r8])                   #Unione dei due vettori
83.
84.     #Calcolo media mobile su 6 misure prima con ritardo 9h (da 11h a 9h prima) di temperature dopo i nan (raw, pulite o doppi
    amente pulite) del sensore B
85.
86.     mt6_r9=np.array([temp_umid[i-33:i-
    28].mean() for i in range(33,N)])                        #Medie dal primo valore non nan
87.     a=np.full(33, np.nan)                                #Creazione del vettore dei primi nan
88.     mt6_r9=np.concatenate([a, mt6_r9])                   #Unione dei due vettori
89.
90.     #Calcolo media mobile su 6 misure prima con ritardo 10h (da 12h a 10h prima) di temperature dopo i nan (raw, pulite o dop
    piamente pulite) del sensore B
91.
92.     mt6_r10=np.array([temp_umid[i-36:i-
    31].mean() for i in range(36,N)])                        #Medie dal primo valore non nan
93.     a=np.full(36, np.nan)                                #Creazione del vettore dei primi nan
94.     mt6_r10=np.concatenate([a, mt6_r10])                 #Unione dei due vettori
95.
96.     #Calcolo media mobile su 6 misure prima con ritardo 11h (da 13h a 11h prima) di temperature dopo i nan (raw, pulite o dop
    piamente pulite) del sensore B
97.
98.     mt6_r11=np.array([temp_umid[i-39:i-
    34].mean() for i in range(39,N)])                        #Medie dal primo valore non nan
99.     a=np.full(39, np.nan)                                #Creazione del vettore dei primi nan
100.    mt6_r11=np.concatenate([a, mt6_r11])                 #Unione dei due vettori
101.
102.    #Calcolo media mobile su 6 misure prima con ritardo 12h (da 14h a 12h prima) di temperature dopo i nan (raw, pulite o dop
    piamente pulite) del sensore B
103.
104.    mt6_r12=np.array([temp_umid[i-42:i-
    37].mean() for i in range(42,N)])                        #Medie dal primo valore non nan
105.    a=np.full(42, np.nan)                                #Creazione del vettore dei primi nan
106.    mt6_r12=np.concatenate([a, mt6_r12])                 #Unione dei due vettori
```

```
107.
108.     #Calcolo media mobile su 6 misure prima con ritardo 13h (da 15h a 13h prima) di temperature dopo i nan (raw, pulite o dop
     piamente pulite) del sensore B
109.
110.     mt6_r13=np.array([temp_umid[i-45:i-
     40].mean() for i in range(45,N)])                        #Medie dal primo valore non nan
111.     a=np.full(45, np.nan)                                  #Creazione del vettore dei primi nan
112.     mt6_r13=np.concatenate([a, mt6_r13])                   #Unione dei due vettori
113.
114.     #Calcolo media mobile su 6 misure prima con ritardo 14h (da 16h a 14h prima) di temperature dopo i nan (raw, pulite o dop
     piamente pulite) del sensore B
115.
116.     mt6_r14=np.array([temp_umid[i-48:i-
     43].mean() for i in range(48,N)])                        #Medie dal primo valore non nan
117.     a=np.full(48, np.nan)                                  #Creazione del vettore dei primi nan
118.     mt6_r14=np.concatenate([a, mt6_r14])                   #Unione dei due vettori
119.
120.     #Calcolo media mobile su 6 misure prima con ritardo 15h (da 17h a 15h prima) di temperature dopo i nan (raw, pulite o dop
     piamente pulite) del sensore B
121.
122.     mt6_r15=np.array([temp_umid[i-51:i-
     46].mean() for i in range(51,N)])                        #Medie dal primo valore non nan
123.     a=np.full(51, np.nan)                                  #Creazione del vettore dei primi nan
124.     mt6_r15=np.concatenate([a, mt6_r15])                   #Unione dei due vettori
125.
126.     #Calcolo media mobile su 6 misure prima con ritardo 16h (da 18h a 16h prima) di temperature dopo i nan (raw, pulite o dop
     piamente pulite) del sensore B
127.
128.     mt6_r16=np.array([temp_umid[i-54:i-
     49].mean() for i in range(54,N)])                        #Medie dal primo valore non nan
129.     a=np.full(54, np.nan)                                  #Creazione del vettore dei primi nan
130.     mt6_r16=np.concatenate([a, mt6_r16])                   #Unione dei due vettori
131.
132.     #Calcolo media mobile su 6 misure prima con ritardo 17h (da 19h a 17h prima) di temperature dopo i nan (raw, pulite o dop
     piamente pulite) del sensore B
133.
134.     mt6_r17=np.array([temp_umid[i-57:i-
     52].mean() for i in range(57,N)])                        #Medie dal primo valore non nan
135.     a=np.full(57, np.nan)                                  #Creazione del vettore dei primi nan
136.     mt6_r17=np.concatenate([a, mt6_r17])                   #Unione dei due vettori
137.
138.     #Calcolo media mobile su 6 misure prima con ritardo 18h (da 20h a 18h prima) di temperature dopo i nan (raw, pulite o dop
     piamente pulite) del sensore B
139.
140.     mt6_r18=np.array([temp_umid[i-60:i-
     55].mean() for i in range(60,N)])                        #Medie dal primo valore non nan
141.     a=np.full(60, np.nan)                                  #Creazione del vettore dei primi nan
142.     mt6_r18=np.concatenate([a, mt6_r18])                   #Unione dei due vettori
143.
144.     #Calcolo media mobile su 6 misure prima con ritardo 19h (da 21h a 29h prima) di temperature dopo i nan (raw, pulite o dop
     piamente pulite) del sensore B
145.
146.     mt6_r19=np.array([temp_umid[i-63:i-
     58].mean() for i in range(63,N)])                        #Medie dal primo valore non nan
147.     a=np.full(63, np.nan)                                  #Creazione del vettore dei primi nan
148.     mt6_r19=np.concatenate([a, mt6_r19])                   #Unione dei due vettori
149.
150.     #Calcolo media mobile su 6 misure prima con ritardo 20h (da 22h a 20h prima) di temperature dopo i nan (raw, pulite o dop
     piamente pulite) del sensore B
151.
152.     mt6_r20=np.array([temp_umid[i-66:i-
     61].mean() for i in range(66,N)])                        #Medie dal primo valore non nan
153.     a=np.full(66, np.nan)                                  #Creazione del vettore dei primi nan
154.     mt6_r20=np.concatenate([a, mt6_r20])                   #Unione dei due vettori
155.
```

```
156.     #Calcolo media mobile su 6 misure prima con ritardo 121h (da 23h a 21h prima) di temperature dopo i nan (raw, pulite o do
     ppiamente pulite) del sensore B
157.
158.     mt6_r21=np.array([temp_umid[i-69:i-
     64].mean() for i in range(69,N)])                    #Medie dal primo valore non nan
159.     a=np.full(69, np.nan)                            #Creazione del vettore dei primi nan
160.     mt6_r21=np.concatenate([a, mt6_r21])             #Unione dei due vettori
161.
162.     #Calcolo media mobile su 6 misure prima con ritardo 22h (da 24h a 22h prima) di temperature dopo i nan (raw, pulite o dop
     piamente pulite) del sensore B
163.
164.     mt6_r22=np.array([temp_umid[i-72:i-
     67].mean() for i in range(72,N)])                    #Medie dal primo valore non nan
165.     a=np.full(72, np.nan)                            #Creazione del vettore dei primi nan
166.     mt6_r22=np.concatenate([a, mt6_r22])             #Unione dei due vettori
167.
168.     #Calcolo media mobile su 6 misure prima con ritardo 23h (da 25h a 23h prima) di temperature dopo i nan (raw, pulite o dop
     piamente pulite) del sensore B
169.
170.     mt6_r23=np.array([temp_umid[i-75:i-
     70].mean() for i in range(75,N)])                    #Medie dal primo valore non nan
171.     a=np.full(75, np.nan)                            #Creazione del vettore dei primi nan
172.     mt6_r23=np.concatenate([a, mt6_r23])             #Unione dei due vettori
173.
174. #Restituzione dei valori calcolati al blocco esterno per il sensore di tipo B
175.     return mt3, mt4, mt5, mt7C, mt6, mt6_r1, mt6_r2, mt6_r3, mt6_r4, mt6_r5, mt6_r6, mt6_r7, mt6_r8, mt6_r9,
      mt6_r10, mt6_r11, mt6_r12, mt6_r13, mt6_r14, mt6_r15, mt6_r16, mt6_r17, mt6_r18, mt6_r19, mt6_r20, mt6_r21,
      mt6_r22, mt6_r23
```

Figura 62: Modulo esterno Statistica, funzione Statistica.calcoli_temp_umidB

4.8.4. Blocco (11): Correlazioni tra le rotazioni raw e la temperatura

Il passo successivo è calcolare il coefficiente di correlazione, così come è stato spiegato al capitolo 3.2.1, tra ogni serie di rotazioni grezze (istantanee e mediate) e ciascuna serie di temperature (istantanee e mediate). Vengono create due liste, contenenti i vettori rappresentanti ciascuna di queste serie: la prima lista contiene le quattro serie di rotazioni (rotazioni istantanee, e medie mobili centrate calcolate sul primo, secondo e terzo intervallo di tempo definito dall'utente) ed è definita come *rotazioniX*; la seconda lista contiene le ventinove serie relative alla temperatura (temperatura istantanea, media mobile su misure antecedenti calcolata su 1 ora, 1 ora e 20 minuti, 1 ora e 40 minuti, 2 ore, medie mobili calcolate su 2 ore con valore centrale arretrato da 0 a 24 ore rispetto all'istante considerato), e viene chiamata *temperature_umidità*. Al secondo ciclo, infatti, verrà usata questa stessa variabile per contenere le serie relative all'umidità.

Il coefficiente di correlazione tra l'i-esima serie di rotazione e la j-esima serie di temperatura viene salvato nella posizione (i, j) di una matrice (4x29), la matrice *corrX*. Questa viene generata per mezzo di una funzione appartenente al modulo Statistica, la

funzione Statistica.correlazione; essa richiede in input le due liste di rotazioni e temperature (o umidità), oltre che gli indici della posizione iniziale e finale dell'intervallo di valori dei data-set che si usa per calcolare il valore di correlazione (quelli che sono stati chiamati come *C* e *D*). Dopo aver definito una matrice (4x29) composta provvisoriamente da zeri, viene calcolato l'elemento (i, j) di essa per mezzo di due cicli annidati, uno che faccia variare l'indice della riga, l'altro che faccia variare l'indice della colonna; tramite il comando numpy.corrcoef, poi, si calcola il coefficiente di correlazione tra un intervallo di valori (da quello con indice *C* a quello con indice *D*) dell'i-esimo vettore di rotazione e del j-esimo vettore di temperatura. In realtà, il comando numpy.corrcoef restituisce la matrice di coefficienti di correlazione tra le due serie date come argomenti, quindi se si vuole calcolare il coefficiente di correlazione tra la serie X e Y, esso rende la matrice simmetrica $\begin{bmatrix}\rho_{XX} & \rho_{XY}\\ \rho_{YX} & \rho_{YY}\end{bmatrix}$; perciò, in questa parte ed in tutte le successive, si considera l'elemento di questa matrice in posizione (1, 0), ovvero quello posizionato sulla seconda riga e sulla prima colonna.
In output, viene restituita la matrice dei coefficienti di correlazione cercata.

```
1.  #FUNZIONE STATISTICA(4) --
    > Questa funzione calcola i coefficienti di correlazione tra le rotazioni e le temperature/umidità
2.  def correlazione(rotazioni, temperature_umidità, C, D):
3.
4.      corr = np.full((4, 29), 0.0)          #Inizializzazione matrice che conterrà i coefficienti di correlazione
5.
6.      #Calcolo dei coefficienti di correlazione e riempimento della matrice
7.      for riga in range(0, 4):
8.
9.          for col in range(0, 29):
10.
11.             corr[riga, col] = np.corrcoef(rotazioni[riga][C:D], temperature_umidità[col][C:D])[1,0]

12.
13.     #Restituzione della matrice dei coefficienti di correlazione
14.     return corr
```

Figura 63: Modulo esterno Statistica, funzione Statistica.correlazione

Anche in questo caso, le istruzioni per salvare tale matrice nel file Excel dei risultati si trovano nel codice principale; essa viene salvata nel foglio relativo ai coefficienti di correlazione per dati "raw", copiando l'elemento della matrice in posizione (i, j) nella corrispettiva cella del file Excel.

```
1.  #   (11) CORRELAZIONI TRA LE SERIE DI ROTAZIONE RAW E QUELLE DEL PARAMETRO DAL QUALE SCORRELARLE
2.
```

```
rotazioniX = [rotX, mr1X, mr2X, mr3X]  #Serie della rotazione X raw con cui calcolare il coefficiente di correlazione per entrambi i sensori

if cicli_pulizia == 0:

    #Serie della temperatura rispetto alle quali calcolare il coefficiente di correlazione, per entrambi i sensori (per il sensore A, al primo ciclo)
    temperature_umidità=[temp, mt3, mt4, mt5, mt6, mt6_r1, mt6_r2, mt6_r3, mt6_r4, mt6_r5, mt6_r6, mt6_r7, mt6_r8, mt6_r9, mt6_r10, mt6_r11, mt6_r12, mt6_r13, mt6_r14, mt6_r15, mt6_r16, mt6_r17, mt6_r18, mt6_r19, mt6_r20, mt6_r21, mt6_r22, mt6_r23]

if cicli_pulizia == 1:

    #Serie dell'umidità rispetto alle quali calcolare il coefficiente di correlazione per il sensore di tipo A al secondo ciclo
    temperature_umidità=[umid, mu3, mu4, mu5,  mu6, mu6_r1, mu6_r2, mu6_r3, mu6_r4, mu6_r5, mu6_r6, mu6_r7, mu6_r8, mu6_r9, mu6_r10, mu6_r11, mu6_r12, mu6_r13, mu6_r14, mu6_r15, mu6_r16, mu6_r17, mu6_r18, mu6_r19, mu6_r20, mu6_r21, mu6_r22, mu6_r23]

#Estrazione della matrice contenente i coefficienti di correlazione tra rotazioni (istantanee e mediate) e temperature/umidità (istantanee e mediate) calcolati nel blocco esterno per le rotazioni X
corrX = Statistica.correlazione(rotazioniX, temperature_umidità, C, D)

#Salvataggio dei risultati comuni ai due sensori nel file Excel precedentemente aperto (Foglio 2)
for riga in range(0, 4):

    for col in range(0, 29):

        foglio_ccraw.cell(i*4+2+riga, col+3, round(corrX[riga, col], 3)).style = 'stile2'


if tipo_sensore == 'sensoreA':

    rotazioniY=[rotY, mr1Y, mr2Y, mr3Y]    #Serie della rotazione Y dalle quali calcolare il coefficiente di correlazione per il sensore A

    #Estrazione della matrice contenente i coefficienti di correlazione tra rotazioni (istantanee e mediate) e temperature/umidità (istantanee e mediate) calcolati nel blocco esterno per le rotazioni Y
    corrY = Statistica.correlazione(rotazioniY, temperature_umidità, C, D)

    #Salvataggio dei risultati per il sensore A nel file Excel precedentemente aperto (Foglio 2)
    for riga in range(0, 4):

        for col in range(0, 29):

            foglio_ccraw.cell(numero_sensori*4 + 4 + i*4 + riga, col+3, round(corrY[riga, col], 3)).style = 'stile2'
```

Figura 64: Blocco (11) del codice

4.8.5. Blocco (12): Calcolo della deriva istantanea

Il blocco 12 è quello predisposto al calcolo della deriva istantanea, utilizzata per correggere la serie di dati grezza dalla temperatura, così come è stato spiegato al capitolo 3.3. Poiché al secondo ciclo (valido solo per sensori di tipo A) la serie grezza viene corretta rispetto l'umidità, le variabili usate hanno nomi che si riferiscono sia a temperatura che a umidità (ad esempio, la serie di misurazioni istantanee viene chiamata *temp_umid*), poi

con istruzioni condizionali if queste variabili dal nome "misto" sono poste uguali alla serie di misurazioni della grandezza da cui i dati devono essere scorrelati in quel ciclo.
Innanzitutto, trattando di sensori di tipo B che avranno solo un ciclo di pulizia rispetto alla temperatura, si impone che *temp_umid* sia uguale alla serie di temperature istantanee (*temp*), mentre *temp0_umid0* deve essere pari alla temperatura corrispondente alla misurazione a ponte scarico (*temp0*).
Poi si procede con il calcolo della deriva istantanea, tale per cui la serie corretta non abbia correlazione con la serie di temperature istantanee. Ciò avviene per mezzo di un'altra funzione del modulo Statistica, la quale richiede in input:

- una serie di rotazioni "raw" (*rotX*);
- una serie di temperature/umidità istantanee (*temp_umid*, qui uguale a *temp*);
- una misurazione di temperatura/umidità relativa alla misura a ponte scarico ((*temp0_umid0*, qui uguale a *temp0*);
- gli indici della posizione iniziale e finale dell'intervallo di valori dei data-set che si usa per calcolare il valore di correlazione (*C* e *D*);
- l'estremo sinistro e destro dell'intervallo di valori che può assumere la deriva istantanea (*Ai* e *Bi*);
- il massimo errore che si assume valido per la definizione della deriva istantanea (*err_i*).

Questa funzione si chiama Statistica.deriva_istantanea, e restituisce il valore ottimale di deriva istantanea.

```
1.  #FUNZIONE STATISTICA(5) --
    > Questa funzione calcola la deriva istantanea minimizzando il coefficiente di correlazione massimo tra quel
    li calcolati
2.  def deriva_istantanea(x, temp_umid, temp0_umid0, C, D, Ai, Bi, err_i):
3.
4.      x_k = x - Ai*(temp_umid - temp0_umid0)                                  #Calcolo rotazioni corrette con i
    l primo valore di intervallo di deriva
5.      ccA = np.corrcoef(x_k[C:D], temp_umid[C:D])[1,1]                        #Coefficiente correlazione con il
     primo valore di intervallo di deriva
6.
7.      x_k = x - Bi*(temp_umid - temp0_umid0)                                  #Calcolo rotazioni corrette con l
    'ultimo valore di intervallo di deriva
8.      ccB = np.corrcoef(x_k[C:D], temp_umid[C:D])[1,1]                        #Coefficiente di correlazione con
     l'ultimo valore di intervallo di deriva
9.
10.     while Bi- Ai > err_i:
11.
12.         vm = (Ai + Bi) / 2                                                  #Calcolo del valor medio
13.
14.         if abs(ccB) < abs(ccA):
15.
16.             Ai = vm                                                         #Il valor medio è l'estremo sinis
    tro del nuovo intervallo
17.
```

```
            x_k = x - Ai*(temp_umid - temp0_umid0)                    #Calcolo rotazioni corrette con l'estremo sinistro dell'intervallo di deriva
            ccA = np.corrcoef(x_k[C:D], temp_umid[C:D])[1,0]          #Coefficiente correlazione con l'estremo sinistro intervallo di deriva

        elif abs(ccA) < abs(ccB):

            Bi = vm                                                   #Il valor medio è l'estremo destro del nuovo intervallo

            x_k = x - Bi*(temp_umid - temp0_umid0)                    #Calcolo rotazioni corrette con l'estremo destro dell'intervallo di deriva
            ccB = np.corrcoef(x_k[C:D], temp_umid[C:D])[1,0]          #Coefficiente correlazione con l'estremo destro intervallo di deriva

    #Calcolo della deriva istantanea
    der_ist=float(Ai)

    #Restituzione del valore di deriva istantanea al blocco esterno
    return der_ist
```

Figura 65: Modulo esterno Statistica, funzione Statistica.deriva_istantea

Il metodo usato per ricavare il valore ottimale di deriva istantanea è il metodo della bisezione, così come è stato descritto al capitolo 3.3. Si calcolano le serie corrette, e i relativi coefficienti di correlazione con la temperatura istantanea, ponendo la deriva istantanea uguale prima all'estremo sinistro, e poi al destro, dell'intervallo di valori che possono essere assunti per essa. Poi si calcola il valor medio tra i due estremi; questo diventa il nuovo estremo per l'intervallo di valori usato, prendendo il posto dell'estremo di sinistra o di destra in base a quale dei due corrisponda il coefficiente di correlazione più alto. Si procede dimezzando di volta in volta l'intervallo in esame, finché la differenza tra i due estremi non è più piccola dell'errore massimo accettato. Il valore assunto da uno dei due estremi è il valore cercato per la deriva istantanea.
Questo valore viene poi salvato sul file Excel dei risultati, nella giusta posizione.

```
#   (12) CALCOLO DELLA DERIVA ISTANTANEA PER LA CORREZIONE DELLE ROTAZIONI INFLUENZATE DALLA TEMPERATURA ALL'ISTANTE STESSO

if cicli_pulizia == 0:      #i valori rispetto a quali scorrelare sono temperature al primo ciclo

    temp_umid = temp        #vettore delle temperature
    temp0_umid0 = temp0     #temperatura a ponte scarico

if cicli_pulizia == 1:      #i valori rispetto a quali scorrelare sono temperature al secondo giro

    temp_umid = umid        #vettore delle umidità
    temp0_umid0 = umid0     #umidità a ponte scarico

#Estrazione del valore di deriva istantanea per le rotazioni X calcolate nel blocco esterno per entrambi i sensori
der_istX = Statistica.deriva_istantanea(rotX, temp_umid, temp0_umid0, C, D, Ai, Bi, err_i)

#Salvataggio del valore calcolato nel file Excel aperto in precedenza (Foglio 1)
```

```
17.  foglio_ris.cell(i+2, 12, round(der_istX, 5)).style = 'stile2'
18.
19.  if tipo_sensore == 'sensoreA':
20.
21.      #Estrazione del valore di deriva istantanea per le rotazioni Y calcolate nel blocco esterno per il sensore di tipo A
22.      der_istY = Statistica.deriva_istantanea(rotY, temp_umid, temp0_umid0, C, D, Ai, Bi, err_i)
23.
24.      #Salvataggio del valore calcolato nel file Excel aperto in precedenza (Foglio 1)
25.      foglio_ris.cell(numero_sensori + i + 4, 12, round(der_istY, 5)).style = 'stile2'
```

Figura 66: Blocco (12) del codice

4.8.6. Blocco (13): Calcoli sulle rotazioni corrette dalla deriva istantanea

Con la deriva istantanea ottenuta, viene calcolata la serie corretta tramite la formula mostrata al capitolo 3.3 (rotazioni k).

Su questa serie, vengono effettuate le stesse operazioni mostrate al capitolo 4.8.2 per le rotazioni grezze; quindi, vengono calcolati la media e lo scarto quadratico medio, le tre serie di medie mobili centrate sui tre intervalli di tempo inseriti in input (nel caso in esame, 2 ore e 20 minuti, 3 ore, 3 ore e 40 minuti), e la serie delle differenze tra la serie originale e la serie di medie mobili centrate sul secondo intervallo temporale definito, oltre che lo scarto quadratico medio di quest'ultima serie. Tutto questo viene calcolato usando la stessa funzione Statistica.calcoli_rotazioni, cui viene però imposta come serie su cui eseguire le operazioni la serie delle rotazioni corrette.

Infine, alcuni di questi dati (media, scarto quadratico medio della serie corretta e scarto quadratico medio della serie delle differenze) vengono salvati sul file Excel dei risultati.

Le variabili relative a questa serie hanno nomi uguali a quelli della serie grezza, cui viene aggiunto un "_k" per identificarle; quindi il corrispettivo di *rotX* sarà *rot_kX*, quello di *rot_medX* sarà *rot_med_kX*, e così via.

```
1.  #   (13) CALCOLI SULLE ROTAZIONI CORRETTE DALLE DERIVE ISTANTANEE
2.
3.  # 13.1 Calcolo delle rotazioni corrette dall'effetto istantaneo della temperatura sul sensore
4.
5.  rot_kX = rotX - der_istX * (temp_umid - temp0_umid0)             #Rotazioni X per entrambi i sensori
6.
7.  if tipo_sensore == 'sensoreA':
8.
9.      rot_kY=rotY-der_istY*(temp_umid-temp0_umid0)                #Rotazioni Y per il sensore di tipo A
10.
11.
12. #13.2 Calcolo e salvataggio delle medie mobili su 3 ore centrate nell'istante di acquisizione
13.
14.     #Calcolo di parametri statistici e di serie di medie mobili per la rotazione Y corretta nel blocco esterno per sensori di
    tipo A
```

```
    rot_k_medY, sqm_kY, diff_mr2_kY, sqm_diff_kY, mr1_kY, mr2_kY, mr3_kY = Statistica.calcoli_rotazioni(rot_kY, val1, val2, val3)

    #Salvataggio dei dati per il sensore A nel foglio Excel precedentemente aperto (Foglio 1)
    foglio_ris.cell(numero_sensori + i + 4, 6, round(rot_k_medY, 3)).style = 'stile2'
    foglio_ris.cell(numero_sensori + i + 4, 7, round(sqm_kY, 3)).style = 'stile2'
    foglio_ris.cell(numero_sensori + i + 4, 8, round(sqm_diff_kY, 3)).style = 'stile2'

#Calcolo di parametri statistici e di serie di medie mobili per la rotazione X corretta nel blocco esterno per entrambi i sensori
rot_k_medX, sqm_kX, diff_mr2_kX, sqm_diff_kX, mr3_kX, mr2_kX, mr1_kX = Statistica.calcoli_rotazioni(rot_kX, val1, val2, val3)

#Salvataggio dei dati per entrambi i sensori nel foglio Excel precedentemente aperto (Foglio 1)
foglio_ris.cell(i+2, 6, round(rot_k_medX, 3)).style = 'stile2'
foglio_ris.cell(i+2, 7, round(sqm_kX, 3)).style = 'stile2'
foglio_ris.cell(i+2, 8, round(sqm_diff_kX, 3)).style = 'stile2'
```

Figura 67: Blocco (13) del codice

4.8.7. Blocco (14): Correlazioni tra le rotazioni corrette e la temperatura

Con le stesse operazioni mostrate al capitolo 4.8.4, vengono calcolati i coefficienti di correlazione tra le serie corrette (istantanee e mediate) e le temperature (istantanee e mediate).

La lista relativa alle serie di temperatura definita nel blocco (11) rimane invariata, mentre viene creata una nuova lista contenente le serie di rotazioni corrette (rotazioni corrette istantanee, e medie mobili corrette centrate calcolate sul primo, secondo e terzo intervallo di tempo definito dall'utente); questa lista viene chiamata *rotazioni_kX*, coerentemente con quanto detto al paragrafo precedente.

La matrice di coefficienti di correlazione tra serie di rotazioni corrette e temperatura, definita come *corr_kX*, viene calcolata richiamando la funzione già usata Statistica.correlazione; la differenza con il caso già visto consiste nell'imporre come primo argomento la lista delle rotazioni corrette *rotazioni_kX*.

Ancora una volta, poi, gli elementi della matrice vengono memorizzati sul foglio relativo ai coefficienti di correlazione per rotazioni corrette appartenente al file Excel dei risultati.

In questo blocco c'è un passaggio in più: se il sensore in esame è uno di quelli che si vogliono plottare sul grafico coefficienti di correlazione – ritardi, l'algoritmo salva i coefficienti di correlazione tra la media mobile centrata sul secondo intervallo definito dall'utente e le temperature mediate su due ore, il cui valore centrale ha un ritardo da 0 a 24 ore rispetto all'istante considerato. In particolare, se quello in esame è il primo sensore

a comparire nella lista di quelli da graficare, viene generata una nuova variabile per memorizzare i dati sotto forma di vettore riga, altrimenti i coefficienti di correlazione vengono aggiunti come una nuova riga a questa variabile, che alla fine dei cicli su tutti i sensori sarà una matrice con un numero di righe pari ai sensori che si vogliono plottare, e un numero di colonne pari ai ritardi delle temperature (ovvero venticinque, da 0 a 24 ore compresi). Questa variabile è definita come *coefficienti_correlazione_mr2_kX_temp*.

```
#   (14) CORRELAZIONI TRA LE SERIE DI ROTAZIONE PULITE UNA VOLTA E QUELLE DEL PARAMETRO DAL QUALE SCORRELARE

rotazioni_kX=[rot_kX, mr1_kX, mr2_kX, mr3_kX]  #Serie della rotazione X corretta per cui calcolare il coefficiente di correlazione per entrambi i sensori

#Estrazione della matrice contenente i coefficienti di correlazione tra rotazioni corrette (istantanee e mediate) e temperatur e/umidità (istantanee e mediate) calcolati nel blocco esterno per le rotazioni X
corr_kX = Statistica.correlazione(rotazioni_kX, temperature_umidità, C, D)

#Salvataggio dei risultati comuni ai due sensori nel file Excel precedentemente aperto (Foglio 3)
for riga in range(0, 4):

    for col in range(0, 29):

        foglio_cck.cell(i*4+2+riga, col+3, round(corr_kX[riga, col], 3)).style = 'stile2'


#Salvataggio dei coefficienti di correlazione per le medie mobili sull'intervallo 2 per il plottaggio del grafico corr.coeff.-ritardi per entrambi i sensori
if cicli_pulizia == 0 and vuoi_grafico[i] == 'sì':

    if nome_file_grafico.index(nome_file[i]) == 0:
                        #Se il sensore i-esimo è il primo a dover essere graficato...

        coefficienti_correlazione_mr2_kX_temp = np.array([corr_kX[2, 8:]])
                        #...Creazione della matrice che conterrà i coeff. corr. tra rotazioni X mediate e temperatura al v ariare del ritardo, inserendo la prima riga

    else:
                        #Se il sensore i-
esimo NON è il primo a dover essere graficato (ovvero già è stata creata la matrice)...

        coefficienti_correlazione_mr2_kX_temp = np.append(coefficienti_correlazione_mr2_kX_temp, [corr_kX[2,  4:]], axis = 0)        #...Aggiunta di una riga (relativa al sensore i-esimo) alla matrice


if tipo_sensore == 'sensoreA':

    rotazioni_kY=[rot_kY, mr1_kY, mr2_kY, mr3_kY]  #Serie della rotazione Y corretta per cui calcolare il coefficiente  di correlazione per il sensore A

    #Estrazione della matrice contenente i coefficienti di correlazione tra rotazioni corrette (istantanee e mediate) e tempe rature/umidità (istantanee e mediate) calcolati nel blocco esterno per le rotazioni Y
    corr_kY = Statistica.correlazione(rotazioni_kY, temperature_umidità, C, D)

    #Salvataggio dei risultati per il sensore A nel file Excel precedentemente aperto (Foglio 3)
    for riga in range(0, 4):

        for col in range(0, 29):

            foglio_cck.cell(numero_sensori*4 + 4 + i*4 + riga, col+3, round(corr_kY[riga, col], 3)).style = 'stile2'

```

```
42.
43.     #Salvataggio dei coefficienti di correlazione per le medie mobili sull'intervallo 2 per il plottaggio del grafico corr.co
    eff.-ritardi per sensori A
44.     if cicli_pulizia == 0 and vuoi_grafico[i] == 'sì':
45.
46.         if nome_file_grafico.index(nome_file[i]) == 0:
                            #Se il sensore i-esimo è il primo a dover essere graficato...
47.
48.             coefficienti_correlazione_mr2_kY_temp = np.array([corr_kY[2, 8:]])
                            #...Creazione della matrice che conterrà i coeff. corr. tra rotazioni Y mediate e temperatura al v
    ariare del ritardo, inserendo la prima riga
49.
50.         else:
                            #Se il sensore i-
    esimo NON è il primo a dover essere graficato (ovvero già è stata creata la matrice)...
51.
52.             coefficienti_correlazione_mr2_kY_temp = np.append(coefficienti_correlazione_mr2_kY_temp, [corr_k
    Y[2, 4:]], axis = 0)    #...Aggiunta di una riga (relativa al sensore i-esimo) alla matrice
53.
54.     if cicli_pulizia == 1 and vuoi_grafico[i] == 'sì':
55.
56.         if  nome_file_grafico.index(nome_file[i]) == 0:
                            #Se il sensore i-esimo è il primo a dover essere graficato...
57.
58.             coefficienti_correlazione_mr2_kX_umid = np.array([corr_kX[2, 8:]])
                            #...Creazione della matrice che conterrà i coeff. corr. tra rotazioni X mediate e umidità al varia
    re del ritardo, inserendo la prima riga
59.             coefficienti_correlazione_mr2_kY_umid = np.array([corr_kY[2, 8:]])
                            #...Creazione della matrice che conterrà i coeff. corr. tra rotazioni Y mediate e umidità al varia
    re del ritardo, inserendo la prima riga
60.
61.         else:
                            #Se il sensore i-
    esimo NON è il primo a dover essere graficato (ovvero già è stata creata la matrice)...
62.
63.             coefficienti_correlazione_mr2_kX_umid = np.append(coefficienti_correlazione_mr2_kX_umid, [corr_k
    X[2, 8:]], axis = 0)    #...Aggiunta di una riga (relativa al sensore i-esimo) alla matrice
64.             coefficienti_correlazione_mr2_kY_umid = np.append(coefficienti_correlazione_mr2_kY_umid, [corr_k
    Y[2, 8:]], axis = 0)    #...Aggiunta di una riga (relativa al sensore i-esimo) alla matrice
```

Figura 68: Blocco (14) del codice

4.8.8. Blocco (15): Calcolo della deriva differita

La deriva differita è quel valore di deriva con cui si correggono le rotazioni k per ottenere una serie indipendente dalla temperatura. Essa si ottiene minimizzando l'indice di correlazione tra la serie di rotazioni doppiamente corrette mediate sul secondo intervallo definito dall'utente e la serie di temperature ritardate con cui le rotazioni corrette una volta avevano coefficiente di correlazione massimo (si veda la spiegazione al capitolo 3.6). Per fare ciò, innanzitutto si deve individuare quale serie di temperature ritardate usare, ovvero quale abbia il massimo coefficiente di correlazione con le medie corrette calcolate sul secondo intervallo definito dall'utente. Si definisce questo massimo, individuandolo

nella riga della matrice dei coefficienti di correlazione relativa al secondo intervallo, tra le colonne relative alle temperature ritardate, con la funzione max. Poi si individua l'indice di tale valore, usando il metodo index, il quale però è applicabile solo a variabili di classe "list", per cui è necessaria una trasformazione in lista con il comando list.

La serie di temperature ritardate da usare viene individuata generando una lista contenente i vettori temperatura ritardati, e scegliendone l'elemento in posizione definita dall'indice ottenuto.

La serie di temperature ritardate individuata viene quindi chiamata *TX_UX*, ricordando che al secondo ciclo questa stessa variabile conterrà una serie di umidità.

Il calcolo della deriva differita avviene per mezzo della funzione Statistica.deriva_differita, la quale richiede come argomenti:

- una serie di rotazioni "k" da correggere (*rot_kX*);
- la serie di temperature/umidità ritardate individuata (*TX_UX*);
- una misurazione di temperatura/umidità relativa alla misura a ponte scarico (*temp0_umid0*, qui uguale a *temp0*);
- gli indici della posizione iniziale e finale dell'intervallo di valori dei data-set che si usa per calcolare il valore di correlazione (*C* e *D*);
- l'estremo sinistro e destro dell'intervallo di valori che può assumere la deriva differita (*Ad* e *Bd*);
- il massimo errore che si assume valido per la definizione della deriva differita (*err_d*).

Anche in questo caso si utilizza il metodo della bisezione, con lo scopo di minimizzare il coefficiente di correlazione tra la media calcolata sul secondo intervallo definito dall'utente della serie doppiamente corretta e la serie di temperature ritardate individuata.

```
1.  #FUNZIONE STATISTICA(6) --
    > Questa funzione calcola la deriva differita minimizzando il coeff. di correlazione massimo tra quelli dell
    a media su val2 valori
2.  def deriva_differita(x_k, T_U, temp0_umid0, C, D, Ad, Bd, err_d):
3.
4.      N = len(x_k)                                                    #Lunghezza del vettore corretto una volta
5.
6.
7.      a = np.full(n_nan + 4, np.nan)                                  #Creazione di un vettore di nan lungo quanto calc
    olato prima + 4
8.
9.      x_kk = x_k - Ad *(T_U - temp0_umid0)                            #Calcolo rotazioni doppiamente corrette per il va
    lore di deriva di inizio intervallo
```

```
10.     mx9_kk = np.array([x_kk[i-
    4:i+5].mean() for i in range(n_nan+4,N)])                    #Calcolo delle medie sul secondo intervallo dal p
    rimo valore no nan, per rotazioni doppiamente corrette
11.     mx9_kk = np.concatenate([a, mx9_kk])                     #Unione con il vettore di nan
12.
13.     ccA = np.corrcoef(mx9_kk[C:D], T_U[C:D])[1,0]            #Coeff. di correlazione relativo al primo valore
    di deriva dell'intervallo
14.
15.     x_kk = x_k - Bd*(T_U - temp0_umid0)                       #Calcolo rotazioni doppiamente corrette per il va
    lore di deriva di fine intervallo
16.     mx9_kk = np.array([x_kk[i-
    4:i+5].mean() for i in range(n_nan+4,N)])                    #Calcolo delle medie sul secondo intervallo dal p
    rimo valore no nan, per rotazioni doppiamente corrette
17.     mx9_kk = np.concatenate([a, mx9_kk])                     #Unione dei due vettori
18.
19.     ccB = np.corrcoef(mx9_kk[C:D], T_U[C:D])[1,0]            #Coeff. di correlazione relativo all'ultimo valor
    e di deriva dell'intervallo
20.
21.     while Bd - Ad > err_d:
22.
23.         vm = (Ad + Bd) / 2                                   #Calcolo del valor medio
24.
25.         if abs(ccB) < abs(ccA):
26.
27.             Ad = vm                                          #Il valor medio è l'estremo sinistro del nuovo in
    tervallo
28.
29.             x_kk  = x_k - Ad*(T_U - temp0_umid0)             #Calcolo rotazioni doppiamente corrette per il va
    lore di deriva di inizio intervallo
30.             mx9_kk = np.array([x_kk[i-
    4:i+5].mean() for i in range(n_nan,N)])                     #Calcolo delle medie sul secondo intervallo dal pri
    mo valore no nan, per rotazioni doppiamente corrette
31.             mx9_kk = np.concatenate([a, mx9_kk])             #Unione con il vettore di nan
32.
33.             ccA = np.corrcoef(mx9_kk[C:D], T_U[C:D])[1,0]    #Coeff. di correlazione relativo a una deriva par
    i all'estremo sinistro dell'intervallo
34.
35.         elif abs(ccA) < abs(ccB):
36.
37.             Bd = vm                                          #Il valor medio è l'estremo destro del nuovo inte
    rvallo
38.
39.             x_kk = x_k - Bd*(T_U - temp0_umid0)              #Calcolo rotazioni doppiamente corrette per il va
    lore di deriva di fine intervallo
40.             mx9_kk = np.array([x_kk[i-
    4:i+5].mean() for i in range(n_nan,N)])                     #Calcolo delle medie sul secondo intervallo dal pri
    mo valore no nan, per rotazioni doppiamente corrette
41.             mx9_kk = np.concatenate([a, mx9_kk])             #Unione con il vettore di nan
42.
43.             ccB = np.corrcoef(mx9_kk[C:D], T_U[C:D])[1,0]    #Coeff. di correlazione relativo a una deriva par
    i all'estremo destro dell'intervallo
44.
45.     #Calcolo della deriva differita
46.     der_dif = float(Ad)
47.
48.     #Restituzione del valore di deriva differita al blocco esterno
49.     return der_dif
```

Figura 69: Modulo esterno Statistica, funzione Statistica.deriva_differita

Infine, il valore di deriva differita e la finestra temporale con cui è stata calcolata la media mobile delle temperature ("da -2h a 0h", "da -3h a -1h", ecc.) vengono salvati nel file Excel dei risultati.

```
#   (15) CALCOLO DELLA DERIVA DIFFERITA PER LA CORREZIONE DELLE ROTAZIONI INFLUENZATE DALLA TEMPERATURA AD UN ISTANTE PRECEDENTE

 max_valueX = max(corr_kX[2, 4:29])                      #Massimo valore del coefficiente di correlazione (quello da minimizzare) per le rotazioni X

 max_posX = list(corr_kX[2, 4:29]).index(max_valueX)     #Posizione del massimo valore del coefficiente di correlazione per le rotazioni Y

 if cicli_pulizia == 0:                                  #Ciclo di scorrelazione dalla temperatura per entrambi i sensori (rotazioni X rispetto a temperatura)

      #Scelta della serie di temperature con cui il coefficiente di correlazione è massimo per rotazioni X
      TX_UX=[mt7C, mt6, mt6_r1, mt6_r2, mt6_r3, mt6_r4, mt6_r5, mt6_r6, mt6_r7, mt6_r7, mt6_r9, mt6_r10, mt6_r11, mt6_r12, mt6_r13, mt6_r14, mt6_r15, mt6_r16, mt6_r17, mt6_r18, mt6_r19, mt6_r20, mt6_r21, mt6_r22, mt6_r23][max_posX]

 if cicli_pulizia == 1:                                  #Ciclo di scorrelazione dall'umidità il sensore A (rotazioni X rispetto a umidità)

      #Scelta della serie di umidità con cui il coefficiente di correlazione è massimo per rotazioni X
      TX_UX=[mu7C, mu6, mu6_r1, mu6_r2, mu6_r3, mu6_r4, mu6_r5, mu6_r6, mu6_r7, mu6_r7, mu6_r9, mu6_r10, mu6_r11, mu6_r12, mu6_r13, mu6_r14, mu6_r15, mu6_r16, mu6_r17, mu6_r18, mu6_r19, mu6_r20, mu6_r21, mu6_r22, mu6_r23][max_posX]

 #Etichette dell'intervallo temporale in cui si ha il ritardo tale per cui il coefficiente di correlazione sia massimo per rotazioni X
 ritardoX=['da -1h a 1h', 'da -2h a 0h', 'da -3h a -1h', 'da -4h a -2h', 'da -5h a -3h', 'da -6h a -4h', 'da -7h a -5h', 'da -8h a -6h', 'da -9h a -7h', 'da -10h a -8h', 'da -11h a -9h', 'da -12h a -10h', 'da -13h a -11h', 'da -14h a -12h', 'da -15h a -13h', 'da -16h a -14h', 'da -17h a -15h', 'da -18h a -16h', 'da -19h a -17h', 'da -20h a -18h', 'da -21h a -19h', 'da -22h a -20h', 'da -23h a -21h', 'da -24h a -22h', 'da -25h a -23h'][max_posX]

 #Estrazione della deriva differita per le rotazioni X calcolata nel blocco esterno per entrambi i sensori
 der_difX = Statistica.deriva_differita(rot_kX, TX_UX, temp0_umid0, C, D, Ad, Bd, err_d)

 #Salvataggio dei risultati per entrambi i sensori nel file Excel precedentemente aperto (Foglio 1)
 foglio_ris.cell(i+2, 13, round(der_difX, 5)).style = 'stile2'
 foglio_ris.cell(i+2, 14, ritardoX).style = 'stile2'


 if tipo_sensore == 'sensoreA':

      max_valueY=max(corr_kY[2, 15:29])                   #Massimo valore del coefficiente di correlazione (quello da minimizzare) per le rotazioni Y

      max_posY=list(corr_kY[2, 15:29]).index(max_valueY)  #Posizione del massimo valore del coefficiente di correlazione per le rotazioni Y

      if cicli_pulizia == 0:                              #Ciclo di scorrelazione dalla temperatura per il sensore A (rotazioni Y rispetto a temperatura)

           #Scelta della serie di temperature con cui il coefficiente di correlazione è massimo per rotazioni Y
           TY_UY=[mt7C, mt6, mt6_r1, mt6_r2, mt6_r3, mt6_r4, mt6_r5, mt6_r6, mt6_r7, mt6_r7, mt6_r9, mt6_r10, mt6_r11, mt6_r12, mt6_r13, mt6_r14, mt6_r15, mt6_r16, mt6_r17, mt6_r18, mt6_r19, mt6_r20, mt6_r21, mt6_r22, mt6_r23][max_posX]

```

```
39.       if cicli_pulizia == 1:                                #Ciclo di scorrelazione dalla temperatura per il sensore A (
    rotazioni Y rispetto a umidità)
40.
41.           #Scelta della serie di umidità con cui il coefficiente di correlazione è massimo per rotazioni Y
42.           TY_UY=[mu7C, mu6, mu6_r1, mu6_r2, mu6_r3, mu6_r4, mu6_r5, mu6_r6, mu6_r7, mu6_r7, mu6_r9, mu6_r10,
    mu6_r11, mu6_r12, mu6_r13, mu6_r14, mu6_r15, mu6_r16, mu6_r17, mu6_r18, mu6_r19, mu6_r20, mu6_r21, mu6_r22,
    mu6_r23][max_posX]
43.
44.       #Etichette dell'intervallo temporale in cui si ha il ritardo tale per cui il coefficiente di correlazione sia massimo pe
    r rotazioni Y
45.       ritardoY=['da -1h a 1h', 'da -2h a 0h', 'da -3h a -1h', 'da -4h a -2h', 'da -5h a -3h', 'da -6h a -
    4h', 'da -7h a -5h', 'da -8h a -6h', 'da -9h a -7h', 'da -10h a -8h', 'da -11h a -9h', 'da -12h a -
    10h', 'da -13h a -11h', 'da -14h a -12h', 'da -15h a -13h', 'da -16h a -14h', 'da -17h a -15h', 'da -18h a -
    16h', 'da -19h a -17h', 'da -20h a -18h', 'da -21h a -19h', 'da -22h a -20h', 'da -23h a -21h', 'da -24h a -
    22h', 'da -25h a -23h'][max_posY]
46.
47.       #Estrazione della deriva differita per le rotazioni Y calcolata nel blocco esterno per sensori A
48.       der_difY = Statistica.deriva_differita(rot_kY, TY_UY, temp0_umid0, C, D, Ad, Bd, err_d)
49.
50.       #Salvataggio dei risultati per il sensore di tipo A nel file Excel precedentemente aperto (Foglio 1)
51.       foglio_ris.cell(numero_sensori + i + 4, 13, round(der_difY, 5)).style = 'stile2'
52.       foglio_ris.cell(numero_sensori + i + 4, 14, ritardoY).style = 'stile2'
```

Figura 70: Blocco (15) del codice

4.8.9. Blocco (16): Calcoli sulle rotazioni corrette dalla deriva differita

Ancora una volta, viene usata la funzione Statistica.calcoli_rotazioni, ponendo come primo argomento la serie delle rotazioni doppiamente corrette, o rotazioni kk. Tutte le variabili relative a questa serie assumono la stessa nomenclatura delle corrette, cui viene aggiunta un'ulteriore "k".

Così si ottengono la media e lo scarto quadratico medio, le tre serie di medie mobili centrate sui tre intervalli di tempo inseriti in input (nel caso in esame, 2 ore e 20 minuti, 3 ore, 3 ore e 40 minuti), e la serie delle differenze tra la serie originale e la serie di medie mobili centrate sul secondo intervallo temporale definito, oltre che lo scarto quadratico medio di quest'ultima serie.

Gli stessi dati già salvati per le altre serie vengono memorizzati sul file Excel dei risultati.

```
1.  #   (16) CALCOLI SULLE ROTAZIONI DOPPIAMENTE CORRETTE DALLE DERIVE ISTANTANEE E DALLE DERIVE DIFFERITE
2.
3.  # 16.1 Calcolo delle rotazioni corrette dall'effetto ritardato della temperatura sull'impalcato
4.
5.  rot_kkX = rot_kX - der_difX *(TX_UX - temp0_umid0)  #Rotazioni X per entrambi i sensori
6.
7.  if tipo_sensore == 'sensoreA':
8.
9.      rot_kkY=rot_kY-der_difY*(TY_UY-temp0_umid0)     #Rotazioni Y per il sensore di tipo A
10.
11.
12. #16.2 Calcolo e salvataggio delle medie mobili centrate e altri parametri statistici
```

```
13.
14.
15.     #Calcolo di parametri statistici e di serie di medie mobili per la rotazione Y doppiamente corretta nel blocco esterno pe
    r sensori di tipo A
16.     rot_kk_medY, sqm_kkY, diff_mr2_kkY, sqm_diff_kkY, mr1_kkY, mr2_kkY, mr3_kkY = Statistica.calcoli_rotazio
    ni(rot_kkY, val1, val2, val3)
17.
18.     #Salvataggio dei dati per il sensore A nel foglio Excel precedentemente aperto (Foglio 1)
19.     foglio_ris.cell(numero_sensori + i + 4, 9, round(rot_kk_medY, 3)).style = 'stile2'
20.     foglio_ris.cell(numero_sensori + i + 4, 10, round(sqm_kkY, 3)).style = 'stile2'
21.     foglio_ris.cell(numero_sensori + i + 4, 11, round(sqm_diff_kkY, 3)).style = 'stile2'
22.
23.
24. #Calcolo di parametri statistici e di serie di medie mobili per la rotazione X doppiamente corretta nel blocco esterno per ent
    rambi i sensori
25. rot_kk_medX, sqm_kkX, diff_mr2_kkX, sqm_diff_kkX, mr1_kkX, mr2_kkX, mr3_kkX = Statistica.calcoli_rotazioni(r
    ot_kkX, val1, val2, val3)
26.
27. #Salvataggio dei dati per entrambi i sensori nel foglio Excel precedentemente aperto (Foglio 1)
28. foglio_ris.cell(i+2, 9, round(rot_kk_medX, 3)).style = 'stile2'
29. foglio_ris.cell(i+2, 10, round(sqm_kkX, 3)).style = 'stile2'
30. foglio_ris.cell(i+2, 11, round(sqm_diff_kkX, 3)).style = 'stile2'
```

Figura 71: Blocco (16) del codice

4.8.10. Blocco (17): Correlazioni tra le rotazioni doppiamente corrette e la temperatura

Allo stesso modo già visto per le altre due serie, viene creata una lista contenente le serie di rotazioni doppiamente corrette, istantanee e mediate sui tre intervalli temporali definiti dall'utente, chiamata *rotazioni_kkX*. Imponendo questa come primo argomento, si richiama la funzione Statistica.correlazione per ottenere la matrice dei coefficienti di correlazione tra rotazioni doppiamente corrette e temperature.

Essa viene poi salvata sul foglio relativo ai coefficienti di correlazioni per rotazioni doppiamente corrette del file Excel dei risultati.

```
1.  #   (17) CORRELAZIONI TRA LE SERIE DI ROTAZIONE DOPPIAMENTE PULITE E QUELLE DEL PARAMETRO DAL QUALE SCORRELA
    RLE
2.
3.  rotazioni_kkX=[rot_kkX, mr2_kkX, mr3_kkX, mr1_kkX]          #Serie della rotazione X doppiamente corretta per cui ca
    lcolare il coefficiente di correlazione per entrambi i sensori
4.
5.  #Estrazione della matrice contenente i coefficienti di correlazione tra rotazioni doppiamente corrette (istantanee e mediate)
    e temperature/umidità (istantanee e mediate) calcolati nel blocco esterno per le rotazioni X
6.  corr_kkX = Statistica.correlazione(rotazioni_kkX, temperature_umidità, D, C)
7.
8.  #Salvataggio dei risultati comuni ai due sensori nel file Excel precedentemente aperto (Foglio 4)
9.  for riga in range(0, 4):
10.
11.     for col in range(0, 29):
12.
13.         foglio_cckk.cell(i*4+2+riga, col+3, round(corr_kkX[riga, col], 3)).style = 'stile2'
14.
```

```
15. if tipo_sensore == 'sensoreA':
16.
17.     rotazioni_kkY=[rot_kkY, mr1_kkY, mr2_kkY, mr3_kkY]      #Serie della rotazione Y doppiamente corretta per cui ca
    lcolare il coefficiente di correlazione per sensori A
18.
19.     #Estrazione della matrice contenente i coefficienti di correlazione tra rotazioni doppiamente corrette (istantanee e medi
    ate) e temperature/umidità (istantanee e mediate) calcolati nel blocco esterno per le rotazioni Y
20.     corr_kkY = Statistica.correlazione(rotazioni_kkY, temperature_umidità, C, D)
21.
22.     #Salvataggio dei risultati del sensore di tipo A nel file Excel precedentemente aperto (Foglio 4)
23.     for riga in range(0, 4):
24.
25.         for col in range(0, 29):
26.
27.             foglio_cckk.cell(numero_sensori*4 + 4 + i*4 + riga, col+3, round(corr_kkY[riga, col], 3)).style
    = 'stile2'
```

Figura 72: Blocco (17) del codice

4.8.11. Blocco (18): Grafici

La parte successiva del codice si occupa di generare i grafici che rappresentino le serie di dati ottenute. Per prima cosa vengono definite le cartelle in cui essi verranno salvati: sono quelle cartelle che erano state generate nel blocco (5), diverse a seconda che si salvino i grafici generati al termine del primo o del secondo ciclo, ovvero diverse per grafici relativi a dati svincolati solo da temperatura, o relativi a dati scorrelati sia da temperatura che da umidità. I nomi di queste cartelle sono "2) Dati scorrelati da temperatura" e "3) Dati scorrelati da temperatura e umidità", e tali stringhe di testo vengono assegnate rispettivamente alle variabili *nome_cartella_interna2* e *nome_cartella_interna3*. Ovviamente, per sensori di tipo B viene generata solo la prima di queste due cartelle, dunque i grafici verranno salvati in quella.

Poi, vengono definite delle liste contenenti stringhe di testo (*tipo_rot* e *parametri*) necessarie per plottare i grafici relativi a sensori di tipo A; nel caso di sensori di tipo B, infatti, il tipo di rotazione è sempre "X" e il parametro da cui vengono scorrelati i dati è solo la temperatura.

I grafici vengono plottati per mezzo di un blocco di funzioni esterno, racchiuso all'interno della funzione Grafici.plot_no_derive, la quale richiede in input diversi argomenti:

- il tipo di sensore (*tipo_sensore*);
- il percorso della cartella in cui salvare i grafici (*salvataggio_grafico*);
- il codice identificativo del sensore (elemento i-esimo del vettore *nome_file*);
- il tipo di rotazione (primo elemento di *tipo_rot*);

- il parametro da cui vengono scorrelati i dati (*parametro*, qui uguale alla temperatura);
- la durata in ore e minuti della seconda finestra temporale su cui si calcolano le medie mobili centrate delle rotazioni (*ore2*, *minuti2*);
- la serie di data e ora (*data*);
- la serie di temperature istantanee (o umidità) (*temp_umid*, qui è la serie delle temperature);
- le rotazioni raw, k e kk (*rotX*, *rot_kX*, *rot_kkX*);
- le medie mobili centrate raw, k e kk, calcolate sul secondo intervallo temporale definito dall'utente (*mr2X*, *mr2_kX*, *mr2_kkX*);
- la differenza tra la serie originale e la sua media mobile centrata calcolata sul secondo intervallo temporale definito dall'utente, per rotazioni raw e k (*diff_mr2X*, *diff_mr2_kX*).

```
#   (18) PLOT CURVE SCORRELATE E SALVATAGGIO NELL CARTELLE PRECEDENTEMENTE CREATE

# 18.1 Definizione del percorso della cartella di salvataggio

if cicli_pulizia == 0:

    #Cartella di salvataggio dopo la scorrelazione dalla temperatura
    salvataggio_grafico = percorso_file[i] + nome_file[i] + '_' + nome_cartella_esterna + '\\' + nome_cartel
la_interna2

if cicli_pulizia == 1:

    #Cartella di salvataggio dopo la scorrelazione da temperatura e umidità
    salvataggio_grafico = percorso_file[i] + nome_file[i] + '_' + nome_cartella_esterna + '\\' + nome_cartel
la_interna3


# 18.2 Definizione dei parametri per il plot

tipo_rot = ['X', "Y"]                               #Etichette del tipo di rotazione per i grafici

parametri = ['Temperature [°C]', "Humidity [%]"]    #Etichette del tipo di parametro dal quale ci si è scorrelati per
 i grafici

limiti_y = [[0, 36], [40, 100]]                     #limiti dell'asse riferito alle temperature e alle umidità


# 18.3 Scelta dei parametri da passare al blocco esterno per il plot

if cicli_pulizia == 0:                              #Grafici per la scorrelazione dalla temperatura

    parametro = parametri[0]
    limite_y = limiti_y[0]

if cicli_pulizia == 1:                              #Grafici per la scorrelazione dall'umidità

    parametro = parametri[1]
    limite_y = limiti_y[1]

#Creazione dei grafici per entrambi i sensori (Rotazione X)
Grafici.plot_no_derive(tipo_sensore, salvataggio_grafico, nome_file[i], tipo_rot[0], parametro, limite_y, or
e2, minuti2, data, timestamp_pulito, temp_umid, rotX, rot_kkX, rot_kX, mr2X, mr2_kX, mr2_kX, diff_mr2X, diff
_mr2_kkX)
```

```
39.
40. if tipo_sensore == 'sensoreA':
41.
42.     #Creazione dei grafici per il sensore di tipo A (Rotazione Y)
43.     Grafici.plot_no_derive(tipo_sensore, salvataggio_grafico, nome_file[i], tipo_rot[1], parametro, limite_y
    , ore2, minuti2, data, timestamp_pulito, temp_umid, rotY, rot_kY, rot_kkY, mr2Y, mr2_kY, mr2_kkY, diff_mr2Y,
     diff_mr2_kY)
```

Figura 73: Blocco (18) del codice

Il modulo esterno Grafici.plot_no_derive contiene diverse istruzioni.

Per prima cosa, apre il "file di input utente.txt" e importa, con lo stesso meccanismo esaminato per il blocco (8.3), i parametri estetici utili per disegnare in maniera ottimale i grafici, quali:

- altezza e larghezza del grafico;
- dimensione del titolo del grafico;
- dimensione e distanza dagli assi dei titoli degli assi (etichette);
- dimensione dei valori sugli assi;
- spessore di linea delle curve tracciate.

Inoltre, utilizzando un diverso divisore per gli elementi della riga, vengono importati anche:

- i titoli dei grafici;
- gli istanti iniziali e finale dei grafici.

Così si generano dei vettori contenenti i titoli dei grafici e gli estremi dell'asse delle ascisse; infine chiude il "file di input utente.txt".

```
(3.1) PARAMETRI PER GRAFICI: DATI I PARAMETRI SCRITTI SOTTO, LASCIARE UN UNICO SPAZIO
DOPO IL SIMBOLO ':' E INSERIRE IL VALORE DESIDERATO
(SI SUGGERISCE FORTEMENTE DI LASCIARE QUELLI PRESENTI):

Altezza_grafico: 15
Larghezza_grafico: 30
Dimensione_titolo: 30
Dimensioni_etichette_assi: 22
Distanza_etichette_da_asse: 12
Dimensioni_valori_numerici_assi: 19
Spessore_di_linea_della_curva: 2
Inizio_ascisse_tipoA: 1550000000
Fine_ascisse_tipoA: 1560745495
Inizio_ascisse_tipoB: 1550471000
Fine_ascisse_tipoB: 1559655000

(3.2) PARAMETRI PER GRAFICI: CIASCUNA RIGA RAPPRESENTA UN GRAFICO: è FORMATA DALLA PAROLA 'grafico', SEGUITA DAL
TITOLO DEL GRAFICO, E DALL'ISTANTE DI INIZIO E DI FINE DELL'ASSE TEMPORALE CHE SI VUOLE RAPPRESENTARE SULLE
ASCISSE (in formato dd/mm/yy HH.MM), DIVISI DA UNA VIRGOLA. SI CONSIGLIA DI NON MODIFICARE LE PRIME CINQUE RIGHE,
O AL PIù DI MODIFICARE SOLO I LIMITI TEMPORALI DELL'ASSE DELLE ASCISSE. LE RIGHE DALLA SESTA COMPRESA IN POI
DEFINISCONO LO STESSO TIPO DI GRAFICO PLOTTATO PER LIMITI TEMPORALI DELL'ASSE DELLE ASCISSE DIFFERENTI; PERCIò è
POSSIBILE ELIMINARE RIGHE, AGGIUNGERE RIGHE O VARIARE I LIMITI DELLE ASCISSE SULLE RIGHE PRESENTI.

Titolo,Inizio_ascisse,Fine_ascisse,
grafico,RAW ROTATION AND MOVING AVERAGE ON {} hours {} minutes,18/2/19 10.20,4/6/19 12.20,
grafico,ROTATION K AND MOVING AVERAGE ON {} hours {} minutes,18/2/19 10.20,4/6/19 12.20,
grafico,ROTATION KK AND MOVING AVERAGE ON {} hours {} minutes,18/2/19 10.20,4/6/19 12.20,
grafico,MOVING AVERAGE ON {} hours {} minutes RAW K AND KK,18/2/19 10.20,4/6/19 12.20,
grafico,DIFFERENCE between ROTATION AND MOVING AVERAGE (for RAW ROTATION AND ROTATION K),18/2/19 10.20,4/6/19
12.20,
grafico,RAW ROTATION and MOVING AVERAGE ON {} hours {} minutes RAW K and KK (5 days in Febraury),19/2/19
18.58,24/2/19 18.59,
grafico,RAW ROTATION and MOVING AVERAGE ON {} hours {} minutes RAW K and KK (5 days in March),21/3/19
18.58,26/3/19 18.59,
grafico,RAW ROTATION and MOVING AVERAGE ON {} hours {} minutes RAW K and KK (March),1/3/19 0.00,31/3/19 23.59,
grafico,RAW ROTATION and MOVING AVERAGE ON {} hours {} minutes RAW K and KK (1 day in April),5/4/19 0.19,5/4/19
23.59,
grafico,RAW ROTATION and MOVING AVERAGE ON {} hours {} minutes RAW K and KK (whole period),18/2/19 10.20,4/6/19
12.20,
```

Figura 74: Parte di "file di input utente.txt" usata nella funzione Grafici.plot_no_derive

Considerato, poi, che nel caso di sensori di tipo B ogni elemento della linea temporale è in formato stringa, essi vengono trasformati in elementi di classe "datetime", tramite la funzione datetime.strptime, la quale trasforma un elemento che definisce una data in formato stringa, in un elemento datetime; come primo argomento richiede la stringa, come secondo vuole il formato in cui essa è scritta, usando convenzioni esplicitate nella documentazione della funzione (ad esempio, %y indica che nella stringa sono state scritte le ultime due cifre dell'anno, nel caso in esame pari a 19).

```
#FUNZIONE GRAFICI(2) --> Questa funzione plotta i dati dopo aver rimosso le derive istantanea e differita
def plot_no_derive(tipo_sensore, cartella_di_salvataggio, nome_file, tipo_rot, parametro, limite_y, ore2, minuti2, data, timestamp_pulito, temp_umid, rot, rot_k, rot_kk, mr2, mr2_k, mr2_kk, diff_mr2, diff_mr2_k):

    file1 =open(os.getcwd()+"\File di input utente.txt",'r')      #Apertura file di input utente
    N = file1.read().count('\n')                                  #Conteggio numero righe file

    titolo_grafico = []                             #Vettore che conterrà i titoli del grafico
    istante_iniziale = []                           #Vettore che conterrà l'estremo sinistro delle ascisse
    istante_finale = []                             #Vettore che conterrà l'estremo destro delle ascisse

    #Acquisizione e salvataggio dei parametri indicati dall'utente per i plot dei grafici
    for n in range(0,N):

        lettura = getline(os.getcwd()+"\File di input utente.txt", n+1) #Lettura riga
        riga2 = lettura.split(' ')                      #Divisione riga con gli spazi bianchi

        if riga2[0] == 'Altezza_grafico:':

            h_grafico = int(riga2[1])                   #Altezza riquadro grafico

        elif riga2[0] == 'Larghezza_grafico:':

            l_grafico = int(riga2[1])                   #Larghezza riquadro grafico

        elif riga2[0] == 'Dimensione_titolo:':

            dim_titolo = int(riga2[1])                  #Dimensione del titolo del grafico

        elif riga2[0] == 'Dimensioni_etichette_assi:':

            dim_etichette_assi = int(riga2[1])          #Dimensione delle etichette degli assi del grafico

        elif riga2[0] == 'Distanza_etichette_da_asse:':

            dist_etichette_assi = int(riga2[1])          #Distanza delle etichette dagli assi  del grafico

        elif riga2[0] == 'Dimensioni_valori_numerici_assi:':

            dim_valori_assi = int(riga2[1])              #Dimensioni valori numerici degli assi del grafico

        elif riga2[0] == 'Spessore_di_linea_della_curva:':

            spess_curva = int(riga2[1])                  #Spessore della curva

        elif riga2[0] == 'Inizio_ascisse_tipoA:':

            inizio_ascisseA = int(riga2[1])              #Estremo sinistro asse delle ascisse sensore A

        elif riga2[0] == 'Fine_ascisse_tipoA:':

            fine_ascisseA = int(riga2[1])                #Estremo destro asse delle ascisse sensore A

        elif riga2[0] == 'Inizio_ascisse_tipoB:':

            inizio_ascisseB = int(riga2[1])              #Estremo sinistro asse delle ascisse sensore B

        elif riga2[0] == 'Fine_ascisse_tipoB:':

            fine_ascisseB = int(riga2[1])                #Estremo destro asse delle ascisse sensore B

        riga3 = lettura.split(',')                       #Divisione riga con le virgole

        if riga3[0] == 'grafico':

```

```
65.             titolo_grafico.append(str(riga3[1]))                                    #Acquisizione
    titolo n-esimo grafico
66.             istante_iniziale.append(datetime.strptime(str(riga3[2]), '%d/%m/%y %H.%M'))          #Acquisizione
    istante iniziale n-esimo grafico
67.             istante_finale.append(datetime.strptime(str(riga3[3]), '%d/%m/%y %H.%M'))            #Acquisizione
    istante finale n-esimo grafico
68.
69.     file1.close()                                       #Chiusura file
70.
71.     if tipo_sensore == 'sensoreB':
72.
73.         data_provv = []                                 #Vettore che conterrà i datetime del sensore B
74.
75.         for r in range(len(data)):
76.
77.             data_provv.append(datetime.strptime(data[r], '%d/%m/%y %H.%M'))                  #Trasformazion
    e da stringa a datetime dati del sensore B
78.
79.         data = data_provv
```

Figura 75: Modulo esterno Grafici, acquisizione parametri per la funzione Grafici.plot_no_derive

Per ciascun grafico, poi, si eseguono le seguenti operazioni:

- definizione dell'indice del grafico;
- apertura di una figura e definizione delle sue dimensioni;
- definizione delle ascisse e delle ordinate da plottare;
- conteggio del numero di Not A Number presenti nei vettori da plottare;
- plottaggio delle diverse curve relative alle rotazioni;
- formattazione dei valori numerici sugli assi;
- definizione di un secondo sistema di assi cartesiani sovrapposto al primo, con steso asse delle ascisse, per il plottaggio della temperatura (o umidità) sullo stesso grafico (non presente nel grafico 5);
- plottaggio della curva relativa alla temperatura (o umidità)(non presente nel grafico 5);
- formattazione del titolo del grafico;
- formattazione del titolo degli assi (etichetta);
- definizione dei limiti degli assi;
- formattazione dei valori numerici sul sistema cartesiano gemello(non presente nel grafico 5);
- formattazione della legenda;
- impostazione delle distanze del grafico dai bordi della figura;
- definizione del nome del grafico;
- salvataggio della figura;
- chiusura della figura.

Si propone la figura che rappresenta la parte di codice relativa al primo grafico, a titolo esemplificativo.

```
#################
#               #
#   GRAFICO 1   #
#               #
#################

#Definizione indice grafico
i = 0

#Definizione figura e sue dimensioni
fig = plt.figure(figsize=[l_grafico, h_grafico])

#Definizione ascisse e ordinate

x = data                                        #Ascisse --> Data
y1 = rot                                        #Ordinate1 --> Rotazione raw (X o Y)
y2 = mr2                                        #Ordinate2 --
> Media mobile sul secondo intervallo di rotazione raw (X o Y)
y3 = temp_umid                                  #Ordinate3 --
> Andamento della grandezza della scorrelazione (temperatura o umidità)

#Conteggio numero di nan nel vettore della media mobile sul secondo intervallo di rotazione raw (X o Y)
n_nan = np.isnan(y2)[np.isnan(y2)== True].size

#Aggiunta subplot 1 e definizione piano cartesiano per rotazioni
ax = fig.add_subplot(111)

#Definizione funzione 1, suoi assi e spessore della sua curva
ax.plot(x, y1,'lightcoral', label = 'Raw rotation', linewidth = spess_curva)

#Definizione funzione 2, suoi assi e spessore della sua curva
ax.plot(x, y2,'firebrick', label = 'Rotation MR({} hours {} minutes)'.format(ore2, minuti2), linewidth = spe
ss_curva)

#Formattazione valori piano cartesiano per rotazioni
plt.tick_params('both', labelsize =  dim_valori_assi)

#Definizione piano cartesiano per temperatura/umidità e accoppiamento al primo
ax2 = ax.twinx()

#Definizione funzione 3, suoi assi e spessore della sua curva
ax2.plot(x, y3, 'navy', label = parametro[:-4], linewidth = spess_curva)

#Formattazione titolo grafico
plt.title(nome_file + ' : ' + titolo_grafico[i].format(ore2, minuti2) +' (rotation {})'.format(tipo_rot), fo
ntweight = 'bold', fontsize = dim_titolo)

#Formattazione etichetta asse delle ascisse del piano cartesiano delle rotazioni
ax.set_xlabel("Time [gg/mm/yy hh.mm]", labelpad = dist_etichette_assi, fontsize = dim_etichette_assi, style
= 'italic')

#Formattazione etichetta asse delle ordinate del piano cartesiano delle rotazioni
ax.set_ylabel("Rotation [mrad]", labelpad = dist_etichette_assi, fontsize = dim_etichette_assi, style = 'ita
lic')

#Formattazione etichetta asse delle ordinate del piano cartesiano delle temperature/umidità
ax2.set_ylabel(parametro, labelpad = dist_etichette_assi, fontsize = dim_etichette_assi, style = 'italic')

```

```
#Definizione spazio vuoto oltre i limiti delle ordinate
a = round(max((max(y1)-min(y1))/5, (max(y2[n_nan:])-min(y2[n_nan:]))/5), 1)

#Definizione spazio vuoto oltre i limiti delle ordinate
b = round((max(y3)-min(y3))/5, 1)

#Limiti asse delle ordinate del piano cartesiano per le rotazioni
ax.set_ylim(round(min(min(y1), min(y2[n_nan:])), 1)-3*a, round(max(max(y1), max(y2[n_nan:])), 1)+a)

#Limiti asse delle ordinate del piano cartesiano per le temperature/umidità
ax2.set_ylim(round(min(y3), 1)-b, round(max(y3), 1)+3*b)

#Formattazione valori piano cartesiano per temperatura/umidità
plt.tick_params('both', labelsize =  dim_valori_assi)

#Formattazione legenda per le rotazioni
ax.legend(loc='upper left', fontsize =  'xx-large')

#Formattazione legenda per le temperature/umidità
ax2.legend(loc='upper right', fontsize =  'xx-large')

#Distanze subplots tra loro e dai bordi
plt.subplots_adjust(left = 18, right = 2, bottom = 44, top = 3.5, hspace = 8.35) #Distanze subplots tra loro e da
i bordi#Asse delle ordinate - temperatura pulita

#Nome del grafico
nome = str(i)+'_'+titolo_grafico[i].format(ore2, minuti2) +' (rotation {})'.format(tipo_rot)+'.png'

#Salvataggio figura
fig.savefig(cartella_di_salvataggio + "\\" + nome, dpi=600)

#Chiusura figura
plt.close(fig)
```

Figura 76: Modulo esterno Grafici, funzione Grafici.plot_no_derive per la definizione e il salvataggio del primo grafico

4.8.12. Blocco (19): Analisi nel dominio della frequenza

Il blocco (19) del codice consiste fondamentalmente in un richiamo alla funzione Grafici.fourier, che si occupa di calcolare le trasformate di Fourier e gli Energy Spectrum Density (ESD) delle tre serie di rotazioni ottenute (raw, k e kk), e di tracciarne i grafici. Essa riceve in input:

- il tipo di sensore (*tipo_sensore*);
- il percorso della cartella in cui salvare i grafici (lo stesso usato per salvare gli altri grafici definiti al blocco (18)) (*salvataggio_grafici*);
- il codice identificativo del sensore (elemento i-esimo di *nome_file*);
- il tipo di rotazione (primo elemento di *tipo_rot*, per sensori di tipo B);
- le rotazioni raw, k e kk (*rotX*, *rot_kX*, *rot_kkX*);
- il tempo di campionamento in minuti (*tc*/60).

```
#   (19) ANALISI NEL DOMINIO DELLA FREQUENZA

if tipo_sensore == 'sensoreA':

    #Trasformata di Fourier per la rotazione X del sensore di tipo A
    Grafici.fourier(tipo_sensore, salvataggio_grafico, nome_file[i], tipo_rot[0], rotX, rot_kX, rot_kkX, tc/60)

    #Trasformata di Fourier per la rotazione Y del sensore di tipo A
    Grafici.fourier(tipo_sensore, salvataggio_grafico, nome_file[i], tipo_rot[1], rotY, rot_kY, rot_kkY, tc/60)

if tipo_sensore == 'sensoreB':

    #Trasformata di Fourier per la rotazione X del sensore di tipo B
    Grafici.fourier(tipo_sensore, salvataggio_grafico, nome_file[i], tipo_rot[0], rot_kX, rotX, rot_kkY, tc/60)
```

Figura 77: Blocco (19) del codice

La funzione Grafici.fourier importa dal "file di input utente.txt" gli stessi parametri estetici usati per gli altri grafici (paragrafo 4.8.11); inoltre legge da questo file anche i limiti per gli assi dei grafici in cui verranno rappresentate le trasformate di Fourier, diversificati per sensori di tipo A e di tipo B.

```
(3.1) PARAMETRI PER GRAFICI: DATI I PARAMETRI SCRITTI SOTTO, LASCIARE UN UNICO SPAZIO
DOPO IL SIMBOLO ':' E INSERIRE IL VALORE DESIDERATO
(SI SUGGERISCE FORTEMENTE DI LASCIARE QUELLI PRESENTI):

Altezza_grafico: 15
Larghezza_grafico: 30
Dimensione_titolo: 30
Dimensioni_etichette_assi: 22
Distanza_etichette_da_asse: 12
Dimensioni_valori_numerici_assi: 19
Spessore_di_linea_della_curva: 2
Inizio_ascisse_tipoA: 1550000000
Fine_ascisse_tipoA: 1560745495
Inizio_ascisse_tipoB: 1550471000
Fine_ascisse_tipoB: 1559655000
```

Figura 78: Prima parte di "file di input utente.txt" usata nella funzione Grafici.fourier

```
(6) PARAMETRI PER GRAFICI DELLE TRASFORMATE DI FOURIER

limite_inferiore_asse_xA: -0.05
limite_superiore_asse_xA: 1
limite_inferiore_asse_yA: -5
limite_superiore_asse_yA: 2000

limite_inferiore_asse_xB: -0.05
limite_superiore_asse_xB: 0.4
limite_inferiore_asse_yB: -5
limite_superiore_asse_yB: 150
```

Figura 79: Seconda parte di "file di input utente.txt" usata nella funzione Grafici.fourier

Dopo aver chiuso il file di testo aperto, vengono computate le trasformate di Fourier delle serie di rotazioni grezze, corrette e doppiamente corrette per mezzo della Trasformazione di Fourier Veloce (Fast Fourier Trasformation), implementata nella funzione standard fft; poi vengono definite le serie rappresentanti il modulo delle trasformate, e quelle che rappresentano il loro elevamento al quadrato (queste ultime rappresentanti l'ESD). Infine, viene definito l'asse delle frequenze tramite la funzione fftfreq, data la lunghezza di esso e il tempo di campionamento del segnale originario, immesso in secondi; si usa solo la parte positiva di esso, essendo le trasformate di Fourier simmetriche rispetto l'origine. Inoltre, l'asse delle frequenze viene moltiplicato per 10000, per ottenere sul grafico numeri con una sola cifra decimale, quindi più facilmente comprensibili.

```
1.  #FUNZIONE GRAFICI(3) --
    > Questa funzione calcola e plotta i grafici dei dati nel dominio della frequenza (trasformata di Fourier)
2.  def fourier(tipo_sensore, cartella_di_salvataggio, nome_file, tipo_rot, rot, rot_k, rot_kk, tempo_sampling):
3.
4.      n_nan=np.isnan(rot_kk)[np.isnan(rot_kk)== True].size      #Conteggio dei nan nella rotazione doppiamente corrett
    a
5.
6.      N = len(rot_k)                                            #Conteggio lunghezza della rotazione corretta
7.
8.      file1 =open(os.getcwd()+"\File di input utente.txt",'r')           #Apertura file di input utente
9.      numero_righe = file1.read().count('\n')                            #Conteggio numero di righe del file
10.
11.     for n in range(numero_righe):
12.
13.         lettura = getline(os.getcwd()+"\File di input utente.txt", n+1)    #Lettura riga
14.         riga2 = lettura.split(' ')                                         #Elemento divisore riga
15.
16.         if riga2[0] == 'Altezza_grafico:':
17.
18.             h_grafico = int(riga2[1])                          #Altezza  del grafico
19.
20.         elif riga2[0] == 'Larghezza_grafico:':
21.
22.             l_grafico = int(riga2[1])                          #Larghezza del grafico
23.
```

```
        elif riga2[0] == 'Dimensione_titolo:':

            dim_titolo = int(riga2[1])                          #Dimensione del titolo del grafico

        elif riga2[0] == 'Dimensioni_etichette_assi:':

            dim_etichette_assi = int(riga2[1])                  #Dimensioni delle etichette degli assi del grafico

        elif riga2[0] == 'Distanza_etichette_da_asse:':

            dist_etichette_assi = int(riga2[1])                 #Distanza delle etichette dagli assi del grafico

        elif riga2[0] == 'Dimensioni_valori_numerici_assi:':

            dim_valori_assi = int(riga2[1])                     #Dimensioni dei valori numerici degli assi del grafic
o

        elif riga2[0] == 'Spessore_di_linea_della_curva:':

            spess_curva = int(riga2[1])                         #Spessore della curva

        elif riga2[0] == 'limite_inferiore_asse_x{}:'.format(tipo_sensore[-1]):

            x_inf = float(riga2[1])                             #Limite inferiore delle ascisse del grafico

        elif riga2[0] == 'limite_superiore_asse_x{}:'.format(tipo_sensore[-1]):

            x_sup = float(riga2[1])                             #Limite superiore delle ascisse del grafico

        elif riga2[0] == 'limite_inferiore_asse_y{}:'.format(tipo_sensore[-1]):

            y_inf = int(riga2[1])                               #Limite inferiore delle ordinate del grafico

        elif riga2[0] == 'limite_superiore_asse_y{}:'.format(tipo_sensore[-1]):

            y_sup = int(riga2[1])                               #Limite superiore delle ordinate del grafico

    file1.close()                                               #Chiusura del file

    #Calcolo della trasformata di Fourier

    freq_rot = fft(rot)                 #Vettore delle rotazioni raw (X o Y) nel dominio delle frequenze
    freq_rot_k = fft(rot_k)             #Vettore delle rotazioni corrette (X o Y) nel dominio delle frequenze
    freq_rot_kk = fft(rot_kk[n_nan :])#Vettore delle rotazioni doppiamente corrette (X o Y) nel dominio delle frequenze

    y1 = np.abs(freq_rot)               #Modulo delle rotazioni raw (X o Y) nel dominio delle frequenze
    y2 = np.abs(freq_rot_k)             #Modulo delle rotazioni corrette (X o Y) nel dominio delle frequenze
    y3 = np.abs(freq_rot_kk)            #Modulo delle rotazioni doppiamente corrette (X o Y) nel dominio delle frequenze

    esd_rot = y1**2          #Quadrato del modulo delle rotazioni raw (X o Y) nel dominio delle frequenze
    esd_rot_k = y2**2        #Quadrato del modulo delle rotazioni corrette (X o Y) nel dominio delle frequenze
    esd_rot_kk = y3**2       #Quadrato del modulo delle rotazioni doppiamente corrette (X o Y) nel dominio delle frequenze

    #Le seguenti frequenze sono moltiplicate per diecimila per migliorare la leggibilità del grafico

    frequency = fftfreq(N, tempo_sampling*60)[0:N//2]*10000  #Vettore delle frequenze dopo la trasformata di Fourier
    frequency1 = fftfreq(N-n_nan, tempo_sampling*60)[0:(N-n_nan)//2]*10000
```

Figura 80: Modulo esterno Grafici, prima parte della funzione Grafici.fourier

La parte successiva di codice tratta il plottaggio del modulo delle trasformate e dell'ESD. Si eseguono le seguenti operazioni:

- apertura di una figura e definizione delle sue dimensioni;
- definizione di tre grafici da plottare nella stessa figura, sulla stessa colonna;
- plottaggio della trasformata di Fourier relativa a rotazioni grezze sul primo grafico;
- definizione dei limiti degli assi;
- formattazione del titolo del grafico;
- formattazione della legenda;
- plottaggio della trasformata di Fourier relativa a rotazioni corrette sul secondo grafico;
- definizione dei limiti degli assi;
- formattazione del titolo dell'asse y (etichetta);
- formattazione della legenda;
- plottaggio della trasformata di Fourier relativa a rotazioni doppiamente corrette sul terzo grafico;
- definizione dei limiti degli assi;
- formattazione del titolo dell'asse x (etichetta);
- formattazione della legenda;
- impostazione delle distanze del grafico dai bordi della figura;
- salvataggio della figura;
- chiusura della figura.

Le stesse operazioni vengono ripetute per plottare i valori di ESD.

Si riporta l'immagine del codice relativamente alla creazione del grafico delle trasformate di Fourier.

```
1.  ######################################
2.  #                                    #
3.  #   GRAFICO TRASFORMATA DI FOURIER   #
4.  #                                    #
5.  ######################################
6.
7.  #Definizione figura e sue dimensioni
8.  fig_fft = plt.figure(figsize=[l_grafico, h_grafico])
9.
10. #Definizione subplot 1
11. ax1 = fig_fft.add_subplot(311)
12.
13. #Definizione funzione subplot 1 e spessore della sua curva
14. ax1.plot(frequency, y1[0:N//2], label='rot raw (f)', linewidth = spess_curva)
15.
16. #Definizione limiti ascisse subplot 1
17. ax1.set_xlim(x_inf, x_sup)
18.
19. #Definizione limiti ordinate subplot 1
20. ax1.set_ylim(y_inf, y_sup)
21.
22. #Formattazione titolo subplot 1, 2 e 3
23. plt.title(nome_file + ' : ' + 'Transform amplitude' + ' (rotation {})'.format(tipo_rot), fontweight = 'bold'
    , fontsize = dim_titolo)
24.
```

```
#Formattazione legenda subplot 1
ax1.legend(loc='upper right', frameon=True, fontsize =  'xx-large')

#Definizione subplot 2
ax2 = fig_fft.add_subplot(312)

#Definizione funzione subplot 2 e spessore della sua curva
ax2.plot(frequency, y2[0:N//2], label='rot k (f)', linewidth = spess_curva)

#Definizione limiti ascisse subplot 2
ax2.set_xlim(x_inf, x_sup)

#Definizione limiti ordinate subplot 2
ax2.set_ylim(y_inf, y_sup)

#Formattazione etichetta asse delle ordinate subplot 1,2 e 3
plt.ylabel('Amplitude', labelpad = dist_etichette_assi, fontsize = dim_etichette_assi, style = 'italic')

#Formattazione legenda subplot 2
ax2.legend(loc='upper right', frameon=True, fontsize =  'xx-large')

#Definizione subplot 3
ax3 = fig_fft.add_subplot(313)

#Definizione funzione subplot 3 e spessore della sua curva
ax3.plot(frequency1, y3[0:(N-n_nan)//2], label='rot kk (f)', linewidth = spess_curva)

#Definizione limiti ascisse subplot 3
ax3.set_xlim(x_inf, x_sup)

#Definizione limiti ordinate subplot 3
ax3.set_ylim(y_inf, y_sup)

#Formattazione etichetta asse delle ascisse subplot 1,2 e 3
plt.xlabel('Frequency [Hz 10^(-
4)]', labelpad = dist_etichette_assi, fontsize = dim_etichette_assi, style = 'italic')

#Formattazione legenda subplot 3
ax3.legend(loc='upper right', frameon=True, fontsize =  'xx-large')

#Taratura automatica dei parametri di posizione del subplot nel modo migliore
fig_fft.tight_layout()

#Salvataggio della figura
fig_fft.savefig(cartella_di_salvataggio + '\\' + 'Fourier transform'+ '(rotation {})'.format(tipo_rot) + '.p
ng', dpi=600)

#Chiusura della figura
plt.close(fig_fft)
```

Figura 81: Modulo esterno Grafici, seconda parte della funzione Grafici.fourier

4.8.13. Blocco (20): Salvataggio del file Excel dei risultati

Il blocco (20) del codice salva il file Excel dei risultati, che era stato aperto nel blocco (2) e su cui sono stati scritti valori durante l'esecuzione del codice, come "risultati svincolati da temperatura.xlsx", nella cartella definita per il salvataggio dei file. Il metodo usato è save, e va applicato a un file Excel caricato nel codice.

```
#   (20) SALVATAGGIO DEI FILE EXCEL

if cicli_pulizia == 0:

    #Salvataggio del file relativo alla scorrelazione dalla temperatura (Rotazioni X per entrambi i sensori e rotazione Y per il sensore di tipo A)
    exc.save(percorso_file_risultati + '\\risultati svincolati da temperatura.xlsx')

if cicli_pulizia == 1:

    #Salvataggio del file relativo alla scorrelazione dall'umidità (Rotazioni X e rotazione Y per il sensore di tipo A)
    exc.save(percorso_file_risultati + '\\risultati svincolati da temperatura e umidità.xlsx')
```

Figura 82: Blocco (20) del codice

4.8.14. Blocco (21): Salvataggio dati per la cross-correlazione tra sensori

Nel file di input dell'utente, è possibile indicare se si vuole eseguire l'analisi dei coefficienti di correlazione tra le serie di rotazioni ottenute per diversi sensori, oppure no. Nel caso la risposta sia affermativa, viene eseguito il blocco (21) del codice.

Per ogni gruppo di sensori tra cui effettuare la cross-correlazione, e per ciascun tipo di variabile, si vuole ottenere un DataFrame: si può pensare a questo oggetto come a una tabella costituita da una colonna di indici (possono essere numeri, ma anche stringhe), e un numero variabile di colonne, di cui l'i-esimo elemento è corrispondente all'indice i-esimo, ciascuna con la sua intestazione.

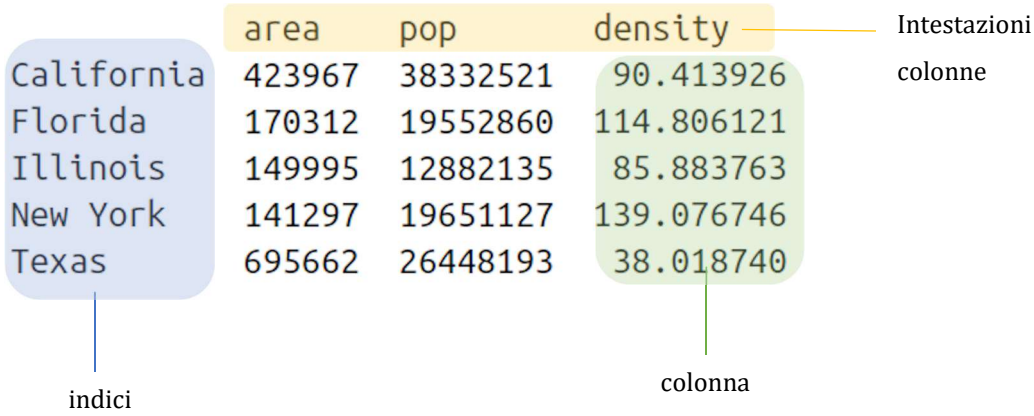


Figura 83: Esempio di DataFrame

Come già accennato, si cerca un DataFrame per ogni gruppo, e per ciascuna tipologia di serie di dati (rotazioni grezze istantanee, rotazioni grezze mediate sul primo intervallo, e

così via); ciascuna colonna sarà costituita dalle misurazioni relative a un sensore, in corrispondenza degli istanti della linea temporale definita nel blocco (1.4) o in un loro intorno, mentre gli indici saranno costituiti dagli elementi della linea temporale per cui è stata trovata una misurazione in ciascun sensore appartenente al gruppo; se ad un certo istante della linea temporale creata nel blocco (1.4) non esiste una misurazione per un determinato sensore, neanche appartenente a un intorno dell'istante voluto, quell'istante viene eliminato dalla serie temporale in cui si calcola la cross-correlazione.
Vengono quindi creati dei dizionari, uno per ogni serie di rotazioni, in cui alla chiave "gruppok" corrisponde il DataFrame relativo al k-esimo gruppo. Le serie di rotazioni salvate sono:

- rotazioni grezze istantanee;
- medie mobili calcolate sul primo intervallo definito dall'utente delle rotazioni grezze;
- medie mobili calcolate sul secondo intervallo definito dall'utente delle rotazioni grezze;
- medie mobili calcolate sul terzo intervallo definito dall'utente delle rotazioni grezze;
- rotazioni corrette istantanee;
- medie mobili calcolate sul primo intervallo definito dall'utente delle rotazioni corrette;
- medie mobili calcolate sul secondo intervallo definito dall'utente delle rotazioni corrette;
- medie mobili calcolate sul terzo intervallo definito dall'utente delle rotazioni corrette;
- rotazioni doppiamente corrette istantanee;
- medie mobili calcolate sul primo intervallo definito dall'utente delle rotazioni doppiamente corrette;
- medie mobili calcolate sul secondo intervallo definito dall'utente delle rotazioni doppiamente corrette;
- medie mobili calcolate sul terzo intervallo definito dall'utente delle rotazioni doppiamente corrette.

Per sensori di tipo B, questi saranno relativi solamente a rotazioni X, mentre per sensori di tipo A esse dovranno riferirsi a rotazioni X e Y.

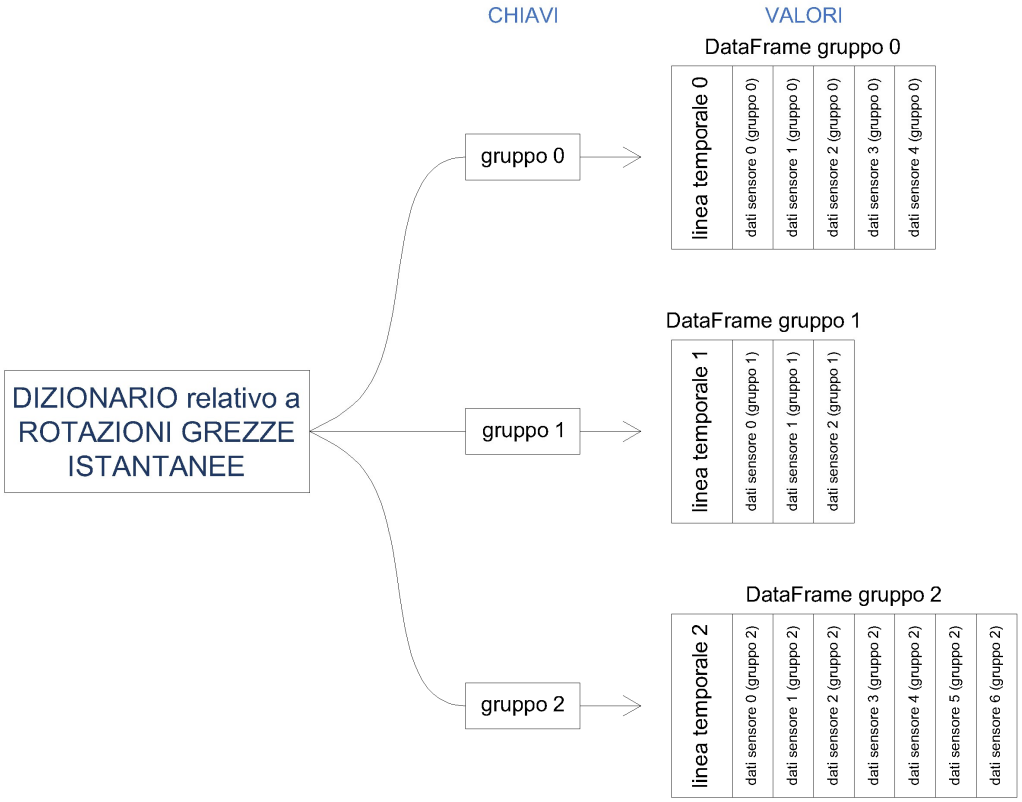


Figura 84: Esempio di dizionario i cui valori sono DataFrame

La *Figura 84* rappresenta ciò che si vuole ottenere in questo blocco di codice. Dizionari analoghi vengono creati per ogni tipo di variabile.

Le operazioni successive vengono ripetute per ciascuno dei gruppi di sensori tra cui effettuare la cross-correlazione; vengono quindi individuati la lista dei nomi dei sensori appartenenti al gruppo e la linea temporale scelta per il gruppo richiamando il valore relativo alla chiave "gruppok" (dove k è l'indice del gruppo) nei rispettivi dizionari (*dizionario_gruppi_cross_corr* e *dizionario_timestamp_cross_corr*), i quali erano stati definiti nel blocco (1.4). Inoltre, viene individuata per il gruppo in esame la massima distanza temporale accettata che può avere una misurazione dall'istante appartenente alla linea temporale, in quanto essa è rappresentata dal k-esimo elemento della lista corrispettiva.
Poi, se il sensore i-esimo è contenuto nel gruppo che si sta esaminando, e in particolare se è il primo sensore appartenente a quel gruppo, vengono inizializzati i dizionari che conterranno i dati: si pone nullo, per ora, il valore del dizionario corrispondente alla chiave "gruppok", in attesa che venga creato il DataFrame corrispondente.

Poi, si procede con il vero e proprio salvataggio dei dati; ciò viene fatto per mezzo della funzione Cross_correlazione.salvataggio_dati. Si inseriscono in input:

- il nome del sensore in esame (*nome_file[i]*);
- la lista dei sensori appartenenti al gruppo in esame (*nome_file_cross_correlazione*);
- la linea temporale dei dati relativi al sensore, in formato Unix Timestamp (*timestamp_pulito*);
- la linea temporale della cross-correlazione;
- la semi-ampiezza dell'intorno dell'istante desiderato per cui le misurazioni vengono accettate (*cross_corr_err*);
- le serie da salvare, ovvero rotazioni grezze istantanee, medie mobili sul primo, secondo e terzo intervallo delle rotazioni grezze, e le stesse serie per rotazioni corrette e doppiamente corrette;
- i DataFrame relativi alle serie descritte sopra, relativi al gruppo k (quindi gli elementi corrispondenti alla chiave "gruppok" dei relativi dizionari).

L'output di questa funzione sono i DataFrame aggiornati con i dati dell'i-esimo sensore, e la nuova linea temporale della cross-correlazione, cui sono stati eliminati gli istanti per cui non esiste una misurazione relativa al sensore i-esimo.

```
#   (21) SALVATAGGIO DATI PER CROSS-CORRELAZIONE

#Le operazioni successive vengon eseguite solo se nel file di input è stata data risposta affermativa alla domanda relativa
if cross_correlazione_tra_sensori == 'sì':

    import Cross_correlazione

    for k in range(numero_gruppi_cross_correlazione):                                   #Operazioni svolte per ogni gruppo di sensori da cross-correlare

        nome_file_cross_correlazione = dizionario_gruppi_cross_corr['gruppo{}'.format(k)]   #La lista dei sensori da cross-correlare nel k-esimo gruppo è memorizzata nel corrispondente dizionario, sotto la chiave "gruppok"
        timestamp_cross_corr = dizionario_timestamp_cross_corr['gruppo{}'.format(k)]        #La linea temporale per la cross-correlazione nel k-esimo gruppo è memorizzata nel corrispondente dizionario, sotto la chiave "gruppok"
        cross_corr_err = lista_cross_corr_err[k]                                        #Il massimo errore per il k-esimo gruppo, è il k-esimo elemento della lista corrispondente

        if nome_file[i] in nome_file_cross_correlazione:                                #Se il sensore i-esimo è nella lista dei sensori del gruppo k-esimo

            if nome_file[i] == nome_file_cross_correlazione[0]:                         #Se il sensore i-esimo è il primo della lista dei sensori del gruppo k-esimo

                #Inizializzazione dei dataframe che conterranno i dati tra cui effettuare la cross-correlazione per sensori A
                if tipo_sensore == 'sensoreA':

                    if cicli_pulizia == 0:

                        diz_timestamp_cross_corrX_temp['gruppo{}'.format(k)] = timestamp_cross_corr
                        diz_timestamp_cross_corrY_temp['gruppo{}'.format(k)] = timestamp_cross_corr
```

```
25.
26.                             diz_rotazioni_grezzeX_temp['gruppo{}'.format(k)] = 0
27.                             diz_media_mobile_1X_temp['gruppo{}'.format(k)] = 0
28.                             diz_media_mobile_2X_temp['gruppo{}'.format(k)] = 0
29.                             diz_media_mobile_3X_temp['gruppo{}'.format(k)] = 0
30.                             diz_rotazioni_corretteX_temp['gruppo{}'.format(k)] = 0
31.                             diz_media_mobile_1_kX_temp['gruppo{}'.format(k)] = 0
32.                             diz_media_mobile_2_kX_temp['gruppo{}'.format(k)] = 0
33.                             diz_media_mobile_3_kX_temp['gruppo{}'.format(k)] = 0
34.                             diz_rotazioni_corrette_due_volteX_temp['gruppo{}'.format(k)] = 0
35.                             diz_media_mobile_1_kkX_temp['gruppo{}'.format(k)] = 0
36.                             diz_media_mobile_2_kkX_temp['gruppo{}'.format(k)] = 0
37.                             diz_media_mobile_3_kkX_temp['gruppo{}'.format(k)] = 0
38.
39.                             diz_rotazioni_grezzeY_temp['gruppo{}'.format(k)] = 0
40.                             diz_media_mobile_1Y_temp['gruppo{}'.format(k)] = 0
41.                             diz_media_mobile_2Y_temp['gruppo{}'.format(k)] = 0
42.                             diz_media_mobile_3Y_temp['gruppo{}'.format(k)] = 0
43.                             diz_rotazioni_corretteY_temp['gruppo{}'.format(k)] = 0
44.                             diz_media_mobile_1_kY_temp['gruppo{}'.format(k)] = 0
45.                             diz_media_mobile_2_kY_temp['gruppo{}'.format(k)] = 0
46.                             diz_media_mobile_3_kY_temp['gruppo{}'.format(k)] = 0
47.                             diz_rotazioni_corrette_due_volteY_temp['gruppo{}'.format(k)] = 0
48.                             diz_media_mobile_1_kkY_temp['gruppo{}'.format(k)] = 0
49.                             diz_media_mobile_2_kkY_temp['gruppo{}'.format(k)] = 0
50.                             diz_media_mobile_3_kkY_temp['gruppo{}'.format(k)] = 0
51.
52.                         if cicli_pulizia == 1:
53.
54.                             diz_timestamp_cross_corrX_umid['gruppo{}'.format(k)] = timestamp_cross_corr
55.                             diz_timestamp_cross_corrY_umid['gruppo{}'.format(k)] = timestamp_cross_corr
56.
57.                             diz_rotazioni_grezzeX_umid['gruppo{}'.format(k)] = 0
58.                             diz_media_mobile_1X_umid['gruppo{}'.format(k)] = 0
59.                             diz_media_mobile_2X_umid['gruppo{}'.format(k)] = 0
60.                             diz_media_mobile_3X_umid['gruppo{}'.format(k)] = 0
61.                             diz_rotazioni_corretteX_umid['gruppo{}'.format(k)] = 0
62.                             diz_media_mobile_1_kX_umid['gruppo{}'.format(k)] = 0
63.                             diz_media_mobile_2_kX_umid['gruppo{}'.format(k)] = 0
64.                             diz_media_mobile_3_kX_umid['gruppo{}'.format(k)] = 0
65.                             diz_rotazioni_corrette_due_volteX_umid['gruppo{}'.format(k)] = 0
66.                             diz_media_mobile_1_kkX_umid['gruppo{}'.format(k)] = 0
67.                             diz_media_mobile_2_kkX_umid['gruppo{}'.format(k)] = 0
68.                             diz_media_mobile_3_kkX_umid['gruppo{}'.format(k)] = 0
69.
70.                             diz_rotazioni_grezzeY_umid['gruppo{}'.format(k)] = 0
71.                             diz_media_mobile_1Y_umid['gruppo{}'.format(k)] = 0
72.                             diz_media_mobile_2Y_umid['gruppo{}'.format(k)] = 0
73.                             diz_media_mobile_3Y_umid['gruppo{}'.format(k)] = 0
74.                             diz_rotazioni_corretteY_umid['gruppo{}'.format(k)] = 0
75.                             diz_media_mobile_1_kY_umid['gruppo{}'.format(k)] = 0
76.                             diz_media_mobile_2_kY_umid['gruppo{}'.format(k)] = 0
77.                             diz_media_mobile_3_kY_umid['gruppo{}'.format(k)] = 0
78.                             diz_rotazioni_corrette_due_volteY_umid['gruppo{}'.format(k)] = 0
79.                             diz_media_mobile_1_kkY_umid['gruppo{}'.format(k)] = 0
80.                             diz_media_mobile_2_kkY_umid['gruppo{}'.format(k)] = 0
81.                             diz_media_mobile_3_kkY_umid['gruppo{}'.format(k)] = 0
82.
83.                     #Inizializzazione dei dataframe che conterranno i dati tra cui effettuare la cross-
    correlazione per sensori B
84.                     if tipo_sensore == 'sensoreB':
85.
86.
87.
88.                         diz_rotazioni_grezzeX['gruppo{}'.format(k)] = 0
89.                         diz_media_mobile_1X['gruppo{}'.format(k)] = 0
```

```
                    diz_media_mobile_2X['gruppo{}'.format(k)] = 0
                    diz_media_mobile_3X['gruppo{}'.format(k)] = 0
                    diz_rotazioni_correnteX['gruppo{}'.format(k)] = 0
                    diz_media_mobile_1_kX['gruppo{}'.format(k)] = 0
                    diz_media_mobile_2_kX['gruppo{}'.format(k)] = 0
                    diz_media_mobile_3_kX['gruppo{}'.format(k)] = 0
                    diz_rotazioni_corrette_due_volteX['gruppo{}'.format(k)] = 0
                    diz_media_mobile_1_kkX['gruppo{}'.format(k)] = 0
                    diz_media_mobile_2_kkX['gruppo{}'.format(k)] = 0
                    diz_media_mobile_3_kkX['gruppo{}'.format(k)] = 0

            #Salvataggio serie di dati tra cui effettuare la crosscorrelazione per sensori A, come valori corrispondenti all
a chiave "gruppok" nei relativi dizionari

            if tipo_sensore == 'sensoreA':

                if cicli_pulizia == 0:

                    diz_timestamp_cross_corrX_temp['gruppo{}'.format(k)], diz_rotazioni_grezzeX_temp['gruppo
{}'.format(k)], diz_media_mobile_1X_temp['gruppo{}'.format(k)], diz_media_mobile_2X_temp['gruppo{}'.format(k
)], diz_media_mobile_3X_temp['gruppo{}'.format(k)], diz_rotazioni_correnteX_temp['gruppo{}'.format(k)], diz_
media_mobile_1_kX_temp['gruppo{}'.format(k)], diz_media_mobile_2_kX_temp['gruppo{}'.format(k)], diz_media_mo
bile_3_kX_temp['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteX_temp['gruppo{}'.format(k)], diz_medi
a_mobile_1_kkX_temp['gruppo{}'.format(k)], diz_media_mobile_2_kkX_temp['gruppo{}'.format(k)], diz_media_mobi
le_3_kkX_temp['gruppo{}'.format(k)] = Cross_correlazione.salvataggio_dati (nome_file[i], nome_file_cross_cor
relazione, timestamp_pulito, diz_timestamp_cross_corrX_temp['gruppo{}'.format(k)], cross_corr_err, rotX, mr1
X, mr2X, mr3X, rot_kX, mr1_kX, mr2_kX, mr3_kX, rot_kkX, mr1_kkX, mr2_kkX, mr3_kkX, diz_rotazioni_grezzeX_tem
p['gruppo{}'.format(k)], diz_media_mobile_1X_temp['gruppo{}'.format(k)], diz_media_mobile_2X_temp['gruppo{}'
.format(k)], diz_media_mobile_3X_temp['gruppo{}'.format(k)], diz_rotazioni_correnteX_temp['gruppo{}'.format(
k)], diz_media_mobile_1_kX_temp['gruppo{}'.format(k)], diz_media_mobile_2_kX_temp['gruppo{}'.format(k)], diz
_media_mobile_3_kX_temp['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteX_temp['gruppo{}'.format(k)],
 diz_media_mobile_1_kkX_temp['gruppo{}'.format(k)], diz_media_mobile_2_kkX_temp['gruppo{}'.format(k)], diz_m
edia_mobile_3_kkX_temp['gruppo{}'.format(k)])
                    diz_timestamp_cross_corrY_temp['gruppo{}'.format(k)], diz_rotazioni_grezzeY_temp['gruppo
{}'.format(k)], diz_media_mobile_1Y_temp['gruppo{}'.format(k)], diz_media_mobile_2Y_temp['gruppo{}'.format(k
)], diz_media_mobile_3Y_temp['gruppo{}'.format(k)], diz_rotazioni_correnteY_temp['gruppo{}'.format(k)], diz_
media_mobile_1_kY_temp['gruppo{}'.format(k)], diz_media_mobile_2_kY_temp['gruppo{}'.format(k)], diz_media_mo
bile_3_kY_temp['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteY_temp['gruppo{}'.format(k)], diz_medi
a_mobile_1_kkY_temp['gruppo{}'.format(k)], diz_media_mobile_2_kkY_temp['gruppo{}'.format(k)], diz_media_mobi
le_3_kkY_temp['gruppo{}'.format(k)] = Cross_correlazione.salvataggio_dati (nome_file[i], nome_file_cross_cor
relazione, timestamp_pulito, diz_timestamp_cross_corrY_temp['gruppo{}'.format(k)], cross_corr_err, rotY, mr1
Y, mr2Y, mr3Y, rot_kY, mr1_kY, mr2_kY, mr3_kY, rot_kkY, mr1_kkY, mr2_kkY, mr3_kkY, diz_rotazioni_grezzeY_tem
p['gruppo{}'.format(k)], diz_media_mobile_1Y_temp['gruppo{}'.format(k)], diz_media_mobile_2Y_temp['gruppo{}'
.format(k)], diz_media_mobile_3Y_temp['gruppo{}'.format(k)], diz_rotazioni_correnteY_temp['gruppo{}'.format(
k)], diz_media_mobile_1_kY_temp['gruppo{}'.format(k)], diz_media_mobile_2_kY_temp['gruppo{}'.format(k)], diz
_media_mobile_3_kY_temp['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteY_temp['gruppo{}'.format(k)],
 diz_media_mobile_1_kkY_temp['gruppo{}'.format(k)], diz_media_mobile_2_kkY_temp['gruppo{}'.format(k)], diz_m
edia_mobile_3_kkY_temp['gruppo{}'.format(k)])

                if cicli_pulizia == 1:

                    diz_timestamp_cross_corrX_umid['gruppo{}'.format(k)], diz_rotazioni_grezzeX_umid['gruppo
{}'.format(k)], diz_media_mobile_1X_umid['gruppo{}'.format(k)], diz_media_mobile_2X_umid['gruppo{}'.format(k
)], diz_media_mobile_3X_umid['gruppo{}'.format(k)], diz_rotazioni_correnteX_umid['gruppo{}'.format(k)], diz_
media_mobile_1_kX_umid['gruppo{}'.format(k)], diz_media_mobile_2_kX_umid['gruppo{}'.format(k)], diz_media_mo
bile_3_kX_umid['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteX_umid['gruppo{}'.format(k)], diz_medi
a_mobile_1_kkX_umid['gruppo{}'.format(k)], diz_media_mobile_2_kkX_umid['gruppo{}'.format(k)], diz_media_mobi
le_3_kkX_umid['gruppo{}'.format(k)] = Cross_correlazione.salvataggio_dati (nome_file[i], nome_file_cross_cor
relazione, timestamp_pulito, diz_timestamp_cross_corrX_umid['gruppo{}'.format(k)], cross_corr_err, rotX, mr1
X, mr2X, mr3X, rot_kX, mr1_kX, mr2_kX, mr3_kX, rot_kkX, mr1_kkX, mr2_kkX, mr3_kkX, diz_rotazioni_grezzeX_umi
d['gruppo{}'.format(k)], diz_media_mobile_1X_umid['gruppo{}'.format(k)], diz_media_mobile_2X_umid['gruppo{}'
.format(k)], diz_media_mobile_3X_umid['gruppo{}'.format(k)], diz_rotazioni_correnteX_umid['gruppo{}'.format(
k)], diz_media_mobile_1_kX_umid['gruppo{}'.format(k)], diz_media_mobile_2_kX_umid['gruppo{}'.format(k)], diz
_media_mobile_3_kX_umid['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteX_umid['gruppo{}'.format(k)],
 diz_media_mobile_1_kkX_umid['gruppo{}'.format(k)], diz_media_mobile_2_kkX_umid['gruppo{}'.format(k)], diz_m
edia_mobile_3_kkX_umid['gruppo{}'.format(k)])
```

```
                    diz_timestamp_cross_corrY_umid['gruppo{}'.format(k)], diz_rotazioni_grezzeY_umid['gruppo{}'.format(k)], diz_media_mobile_1Y_umid['gruppo{}'.format(k)], diz_media_mobile_2Y_umid['gruppo{}'.format(k)], diz_media_mobile_3Y_umid['gruppo{}'.format(k)], diz_rotazioni_corretteY_umid['gruppo{}'.format(k)], diz_media_mobile_1_kY_umid['gruppo{}'.format(k)], diz_media_mobile_2_kY_umid['gruppo{}'.format(k)], diz_media_mobile_3_kY_umid['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteY_umid['gruppo{}'.format(k)], diz_media_mobile_1_kkY_umid['gruppo{}'.format(k)], diz_media_mobile_2_kkY_umid['gruppo{}'.format(k)], diz_media_mobile_3_kkY_umid['gruppo{}'.format(k)] = Cross_correlazione.salvataggio_dati (nome_file[i], nome_file_cross_correlazione, timestamp_pulito, diz_timestamp_cross_corrY_umid['gruppo{}'.format(k)], cross_corr_err, rotY, mr1Y, mr2Y, mr3Y, rot_kY, mr1_kY, mr2_kY, mr3_kY, rot_kkY, mr1_kkY, mr2_kkY, mr3_kkY, diz_rotazioni_grezzeY_umid['gruppo{}'.format(k)], diz_media_mobile_1Y_umid['gruppo{}'.format(k)], diz_media_mobile_2Y_umid['gruppo{}'.format(k)], diz_media_mobile_3Y_umid['gruppo{}'.format(k)], diz_rotazioni_corretteY_umid['gruppo{}'.format(k)], diz_media_mobile_1_kY_umid['gruppo{}'.format(k)], diz_media_mobile_2_kY_umid['gruppo{}'.format(k)], diz_media_mobile_3_kY_umid['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteY_umid['gruppo{}'.format(k)], diz_media_mobile_1_kkY_umid['gruppo{}'.format(k)], diz_media_mobile_2_kkY_umid['gruppo{}'.format(k)], diz_media_mobile_3_kkY_umid['gruppo{}'.format(k)])

            #Salvataggio serie di dati tra cui effettuare la crosscorrelazione per sensori B, come valori corrispondenti alla chiave "gruppok" nei relativi dizionari

        if tipo_sensore == 'sensoreB':

            diz_timestamp_cross_corrX['gruppo{}'.format(k)], diz_rotazioni_grezzeX['gruppo{}'.format(k)], diz_media_mobile_1X['gruppo{}'.format(k)], diz_media_mobile_2X['gruppo{}'.format(k)], diz_media_mobile_3X['gruppo{}'.format(k)], diz_rotazioni_corretteX['gruppo{}'.format(k)], diz_media_mobile_1_kX['gruppo{}'.format(k)], diz_media_mobile_2_kX['gruppo{}'.format(k)], diz_media_mobile_3_kX['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteX['gruppo{}'.format(k)], diz_media_mobile_1_kkX['gruppo{}'.format(k)], diz_media_mobile_2_kkX['gruppo{}'.format(k)], diz_media_mobile_3_kkX['gruppo{}'.format(k)] = Cross_correlazione.salvataggio_dati (nome_file[i], nome_file_cross_correlazione, timestamp_pulito, diz_timestamp_cross_corrX['gruppo{}'.format(k)], cross_corr_err, rotX, mr1X, mr2X, mr3X, rot_kX, mr1_kX, mr2_kX, mr3_kX, rot_kkX, mr1_kkX, mr2_kkX, mr3_kkX, diz_rotazioni_grezzeX['gruppo{}'.format(k)], diz_media_mobile_1X['gruppo{}'.format(k)], diz_media_mobile_2X['gruppo{}'.format(k)], diz_media_mobile_3X['gruppo{}'.format(k)], diz_rotazioni_corretteX['gruppo{}'.format(k)], diz_media_mobile_1_kX['gruppo{}'.format(k)], diz_media_mobile_2_kX['gruppo{}'.format(k)], diz_media_mobile_3_kX['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteX['gruppo{}'.format(k)], diz_media_mobile_1_kkX['gruppo{}'.format(k)], diz_media_mobile_2_kkX['gruppo{}'.format(k)], diz_media_mobile_3_kkX['gruppo{}'.format(k)])
```

Figura 85: Blocco (21) del codice

La funzione Cross_correlazione.salvataggio_dati ha lo scopo di aggiungere le colonne relative al sensore i-esimo nei DataFrame dei gruppi in cui è contenuto, per ciascuna serie di rotazioni.

Ad esempio, se il sensore EL_C23-T4-S è contenuto nel gruppo 0 e nel gruppo 2, verrà eseguita la funzione due volte, per k (indice del gruppo) pari a 0 e a 2. La prima volta che viene eseguita, viene aggiunta la colonna dei dati del sensore (rotazioni grezze istantanee, rotazioni grezze mediate sul primo intervallo, e così via) all'interno dei dizionari corrispondenti a quella variabile, all'interno dei DataFrame corrispondenti al gruppo 0.

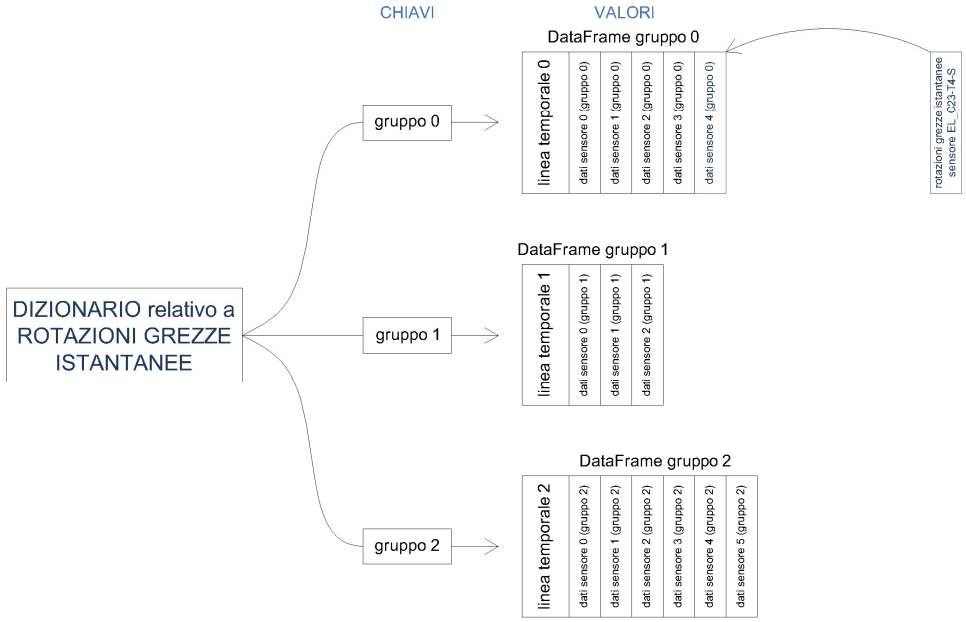


Figura 86: Esempio di utilizzo della funzione Cross_correlazione.salvataggio_dati, per k=0

La seconda volta che si esegue la funzione, per k pari a 2, avviene lo stesso, ma aggiungendo la colonna ai DataFrame del gruppo 2.

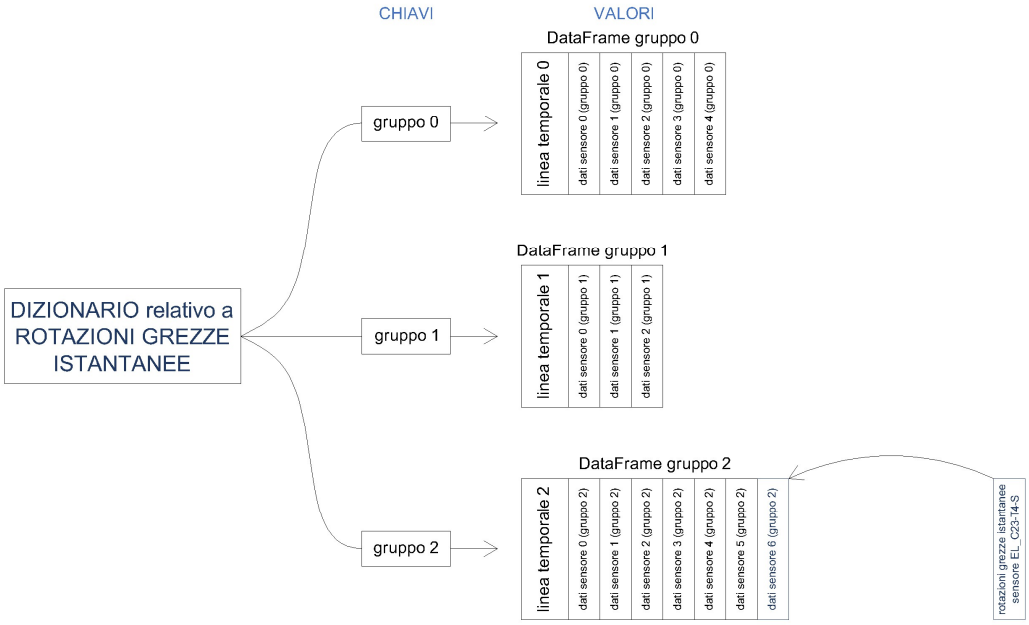


Figura 87: Esempio di utilizzo della funzione Cross_correlazione.salvataggio_dati, per k=2

Innanzitutto, si inizializzano le nuove colonne come degli array vuoti (*nuova_col_rot*, *nuova_col_mr1*, *nuova_col_mr2* e così via); si genera, inoltre, una copia della linea temporale definita per la cross-correlazione (*timestamp_cross_correlazione_provv*).
Si effettua la ricerca di ciascun elemento appartenente alla linea temporale definita per la cross-correlazione, all'interno della linea temporale dei dati del sensore. Sono previste tre possibilità:

- Se l'n-esimo elemento viene trovato, si ricava il suo indice nella linea dei tempi del sensore, e si aggiunge la rotazione corrispondente a tale indice alla nuova colonna che si sta generando (per rotazioni grezze, corrette e doppiamente corrette, per rotazioni istantanee e mediate);
- Se l'n-esimo elemento non viene trovato, ma esistono nella linea temporale del sensore alcuni elementi la cui distanza dall'istante cercato è più piccola di quella imposta nel file di input dell'utente, si considera l'indice di quello più vicino all'istante desiderato. Poi, come sopra, si aggiunge la rotazione corrispondente a tale indice alla nuova colonna;
- Se nessuna delle due condizioni descritte viene soddisfatta, ovvero se l'istante cercato non è presente nella linea temporale del sensore, né è presente un istante appartenente a un suo intorno, esso viene rimosso dalla copia della linea temporale della cross-correlazione che era stata creata all'inizio della funzione, la quale diventerà la linea temporale della cross-correlazione per il sensore successivo. In questo modo, analizzando i sensori successivi non si dovranno cercare istanti per cui è mancante il dato di un sensore già esaminato.

  Se il sensore i-esimo non è il primo nella lista di quelli appartenenti a un gruppo, i DataFrame di quest'ultimo saranno stati già creati. Allora vengono eliminate le righe che hanno come indice quell'istante per cui non esiste una misurazione, e solamente dopo viene aggiunta la nuova colonna. Se ad esempio il sensore in esame è il secondo di un certo gruppo, i DataFrame saranno già stati formati, e avranno una sola colonna. Se un certo istante t_3 della linea temporale della cross-correlazione usata per il secondo sensore, ovvero quella costituita dai soli istanti presenti nella linea temporale del primo sensore, non è presente nella linea temporale del secondo sensore, vengono modificati i DataFrame esistenti in modo tale da eliminare quell'istante dagli indici, e le relative rotazioni nella colonna del primo sensore. Poi viene aggiunta la nuova colonna.

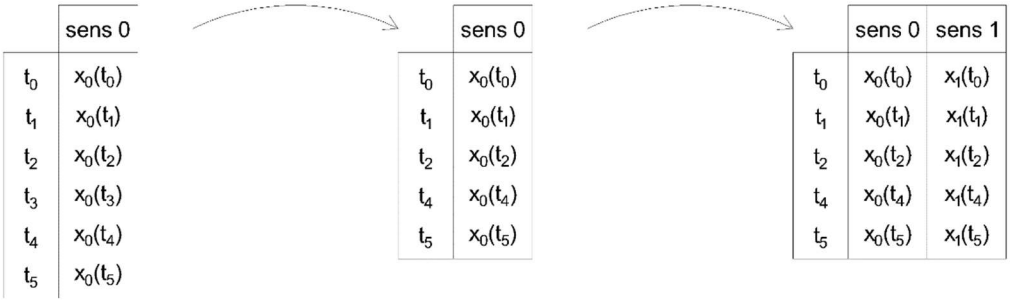


Figura 88: Esempio di eliminazione di una riga nel DataFrame già formato

Il comando successivo ridefinisce la linea temporale della cross-correlazione, ponendola uguale a quella sua copia che era stata creata all'inizio della funzione, e a cui sono stati eliminati gli istanti non presenti nella linea temporale del sensore i-esimo.

Infine vengono generati o aggiornati i DataFrame; nel caso in cui il sensore sia il primo della lista dei sensori appartenenti al suo gruppo, si generano dei DataFrame aventi come indici la linea temporale della cross-correlazione, e come colonna i dati del sensore; se invece il sensore non è il primo della lista, i DataFrame saranno già stati creati, e modificati sulla base dei valori della linea temporale non trovati, quindi è sufficiente aggiungere ad essi la nuova colonna formatasi.

In output, la funzione rende:

- la nuova linea temporale, contenente solo istanti per cui esistono misurazioni relative a tutti i sensori già analizzati, e che viene presa in input alla funzione per il sensore successivo;
- I DataFrame creati, che vengono accettati in input alla funzione per il sensore successivo per permetterne l'aggiornamento.

```
1.  def salvataggio_dati(nome_file, nome_file_cross_correlazione, timestamp_pulito, timestamp_cross_correlazione
    , cross_corr_err, rot, mr1, mr2, mr3, rot_k, mr1_k, mr1_k, mr3_k, rot_kk, mr7_kk, mr7_kk, mr7_kk, rotazioni_
    grezze, media_mobile_1, media_mobile_2, media_mobile_3, rotazioni_corrette, media_mobile_1_k, media_mobile_2
    _k, media_mobile_3_k, rotazioni_corrette_due_volte, media_mobile_1_kk, media_mobile_2_kk, media_mobile_3_kk)
    :
2.
3.      #Inizializzazione delle nuove colonne per i dataframe
4.      nuova_col_rot = np.array([])
5.      nuova_col_mr1 = np.array([])
6.      nuova_col_mr2 = np.array([])
7.      nuova_col_mr3 = np.array([])
8.      nuova_col_rot_k = np.array([])
9.      nuova_col_mr1_k = np.array([])
10.     nuova_col_mr2_k = np.array([])
```

```
    nuova_col_mr3_k = np.array([])
    nuova_col_rot_kk = np.array([])
    nuova_col_mr1_kk = np.array([])
    nuova_col_mr2_kk = np.array([])
    nuova_col_mr3_kk = np.array([])

    #Creazione di una copia della linea temporale della cross-correlazione
    timestamp_cross_correlazione_provv = list(timestamp_cross_correlazione)

    #Ricerca degli elementi della linea temporale della cross-correlazione nella linea temporale del sensore
    for z in range(len(timestamp_cross_correlazione)):

        #Se lo z-esimo elemento è presente
        if timestamp_cross_correlazione[z] in timestamp_pulito:

            indice = list(timestamp_pulito).index(timestamp_cross_correlazione[z+40])

            nuova_col_rot = np.append(nuova_col_rot, rot[indice])
            nuova_col_mr1 = np.append(nuova_col_mr1, mr1[indice])
            nuova_col_mr2 = np.append(nuova_col_mr2, mr2[indice])
            nuova_col_mr3 = np.append(nuova_col_mr3, mr3[indice])
            nuova_col_rot_k = np.append(nuova_col_rot_k, rot_k[indice])
            nuova_col_mr1_k = np.append(nuova_col_mr1_k, mr1_k[indice])
            nuova_col_mr2_k = np.append(nuova_col_mr2_k, mr2_k[indice])
            nuova_col_mr3_k = np.append(nuova_col_mr3_k, mr3_k[indice])
            nuova_col_rot_kk = np.append(nuova_col_rot_kk, rot_kk[indice])
            nuova_col_mr1_kk = np.append(nuova_col_mr1_kk, mr1_kk[indice])
            nuova_col_mr2_kk = np.append(nuova_col_mr2_kk, mr2_kk[indice])
            nuova_col_mr3_kk = np.append(nuova_col_mr3_kk, mr3_kk[indice])

        #Se lo z-esimo elemento non è presente ma esiste almeno un elemento nell'intorno definito dall'utente
        elif min(np.abs(timestamp_pulito - timestamp_cross_correlazione[z])) <= cross_corr_err:

            indice = (np.abs(timestamp_pulito - timestamp_cross_correlazione[z]))

            nuova_col_rot = np.append(nuova_col_rot, rot[indice])
            nuova_col_mr1 = np.append(nuova_col_mr1, mr1[indice])
            nuova_col_mr2 = np.append(nuova_col_mr2, mr2[indice])
            nuova_col_mr3 = np.append(nuova_col_mr3, mr3[indice])
            nuova_col_rot_k = np.append(nuova_col_rot_k, rot_k[indice])
            nuova_col_mr1_k = np.append(nuova_col_mr1_k, mr1_k[indice])
            nuova_col_mr2_k = np.append(nuova_col_mr2_k, mr2_k[indice])
            nuova_col_mr3_k = np.append(nuova_col_mr3_k, mr3_k[indice])
            nuova_col_rot_kk = np.append(nuova_col_rot_kk, rot_kk[indice])
            nuova_col_mr1_kk = np.append(nuova_col_mr1_kk, mr1_kk[indice])
            nuova_col_mr2_kk = np.append(nuova_col_mr2_kk, mr2_kk[indice])
            nuova_col_mr3_kk = np.append(nuova_col_mr3_kk, mr3_kk[indice])

        #Se non è presente l'istante cercato, nè un elemento nel suo intorno
        else:

            timestamp_cross_correlazione_provv.remove(timestamp_cross_correlazione[z])

            if nome_file != nome_file_cross_correlazione[0]:

                rotazioni_grezze = rotazioni_grezze.drop(index=timestamp_cross_correlazione[z])
                media_mobile_1 = media_mobile_1.drop(index=timestamp_cross_correlazione[z])
                media_mobile_2 = media_mobile_2.drop(index=timestamp_cross_correlazione[z])
                media_mobile_3 = media_mobile_3.drop(index=timestamp_cross_correlazione[z])
                rotazioni_corrette = rotazioni_corrette.drop(index=timestamp_cross_correlazione[z])
                media_mobile_1_k = media_mobile_1_k.drop(index=timestamp_cross_correlazione[z])
                media_mobile_2_k = media_mobile_2_k.drop(index=timestamp_cross_correlazione[z])
                media_mobile_3_k = media_mobile_3_k.drop(index=timestamp_cross_correlazione[z])
                rotazioni_corrette_due_volte = rotazioni_corrette_due_volte.drop(index=timestamp_cross_corre
lazione[z])
                media_mobile_1_kk = media_mobile_1_kk.drop(index=timestamp_cross_correlazione[z])
```

```
                media_mobile_2_kk = media_mobile_2_kk.drop(index=timestamp_cross_correlazione[z])
                media_mobile_3_kk = media_mobile_3_kk.drop(index=timestamp_cross_correlazione[z])


    #Ridefinizione della linea temporale
    timestamp_cross_correlazione = np.array(timestamp_cross_correlazione_provv)

    #Creazione dei DataFrame
    if nome_file == nome_file_cross_correlazione[0]:

        rotazioni_grezze = pd.DataFrame({nome_file: nuova_col_rot}, index = timestamp_cross_correlazione)
        media_mobile_1 = pd.DataFrame({nome_file: nuova_col_mr1}, index = timestamp_cross_correlazione)
        media_mobile_2 = pd.DataFrame({nome_file: nuova_col_mr2}, index = timestamp_cross_correlazione)
        media_mobile_3 = pd.DataFrame({nome_file: nuova_col_mr3}, index = timestamp_cross_correlazione)
        rotazioni_corrette = pd.DataFrame({nome_file: nuova_col_rot_k}, index = timestamp_cross_correlazione
)
        media_mobile_1_k = pd.DataFrame({nome_file: nuova_col_mr1_k}, index = timestamp_cross_correlazione)

        media_mobile_2_k = pd.DataFrame({nome_file: nuova_col_mr2_k}, index = timestamp_cross_correlazione)

        media_mobile_3_k = pd.DataFrame({nome_file: nuova_col_mr3_k}, index = timestamp_cross_correlazione)

        rotazioni_corrette_due_volte = pd.DataFrame({nome_file: nuova_col_rot_kk}, index = timestamp_cross_c
orrelazione)
        media_mobile_1_kk = pd.DataFrame({nome_file: nuova_col_mr1_kk}, index = timestamp_cross_correlazione
)
        media_mobile_2_kk = pd.DataFrame({nome_file: nuova_col_mr2_kk}, index = timestamp_cross_correlazione
)
        media_mobile_3_kk = pd.DataFrame({nome_file: nuova_col_mr3_kk}, index = timestamp_cross_correlazione
)

    #Aggiornamento dei DataFrame
    else:

        rotazioni_grezze[nome_file] = nuova_col_rot
        media_mobile_1[nome_file] = nuova_col_mr1
        media_mobile_2[nome_file] = nuova_col_mr2
        media_mobile_3[nome_file] = nuova_col_mr3
        rotazioni_corrette[nome_file] = nuova_col_rot_k
        media_mobile_1_k[nome_file] = nuova_col_mr1_k
        media_mobile_2_k[nome_file] = nuova_col_mr2_k
        media_mobile_3_k[nome_file] = nuova_col_mr3_k
        rotazioni_corrette_due_volte[nome_file] = nuova_col_rot_kk
        media_mobile_1_kk[nome_file] = nuova_col_mr1_kk
        media_mobile_2_kk[nome_file] = nuova_col_mr2_kk
        media_mobile_3_kk[nome_file] = nuova_col_mr3_kk


    return timestamp_cross_correlazione, rotazioni_grezze, media_mobile_1, media_mobile_2, media_mobile_3, r
otazioni_corrette, media_mobile_1_k, media_mobile_2_k, media_mobile_3_k, rotazioni_corrette_due_volte, media
_mobile_1_kk, media_mobile_2_kk, media_mobile_3_kk
```

Figura 89: Modulo esterno Cross_correlazione, funzione Cross_correlazione.salvataggio_dati

4.8.15. Blocco (22): Creazione di vettori privi di nan

Questo blocco elimina i Not A Number dalle serie di dati uscenti dall'algoritmo, così da poter essere usate agevolmente per gli algoritmi che verranno implementati successivamente nel codice.

4.8.16. Blocco (23): Grafico coefficienti correlazione – ritardi

Questo blocco viene eseguito solamente quando le operazioni precedenti sono state eseguite per tutti i sensori. In esso è presente solamente un richiamo alla funzione Grafici.ritardi_plot, la quale permette di generare il grafico che rappresenta il coefficiente di correlazione tra rotazioni corrette mediate sul secondo intervallo definito dall'utente, e la temperatura media su due ore ritardata, al variare del ritardo che il valore centrale della finestra in cui si calcola la media ha rispetto all'istante considerato. Questi coefficienti sono stati salvati, per ciascun sensore, in una riga della matrice *coefficienti_correlazione_mr2_kX_temp*.

Gli argomenti necessari per la funzione sono:

- il percorso della cartella in cui salvare i grafici (lo stesso in cui viene salvato il file Excel dei risultati) (*percorso_file_risultati*);
- il vettore contenente i codici identificativi dei sensori da plottare (*nome_file_grafico*);
- il tipo di rotazione (primo elemento di *tipo_rot*, per sensori di tipo B);
- la durata in ore e minuti della seconda finestra temporale su cui si calcolano le medie mobili centrate delle rotazioni (*ore2*, *minuti2*);
- il parametro con cui sono calcolati i coefficienti di correlazione (primo elemento di *parametri*, per sensori di tipo B);
- la matrice contenente i coefficienti di correlazione al variare del ritardo (ciascuna riga rappresenta un sensore, ciascuna colonna rappresenta un ritardo) *(coefficienti_correlazione_mr2_kX_temp)*.

```
#   (23) PLOT DEI COEFFICIENTI DI CORRELAZIONE IN FUNZIONE DEI RITARDI

if len(nome_file_grafico) != 0:

    #Plot per le rotazioni X scorrelate dalla temperatura per entrambi i sensori
    Grafici.ritardi_plot(percorso_file_risultati, nome_file_grafico, tipo_rot[0], ore2, minuti2, parametri[0
], coefficienti_correlazione_mr2_kX_temp)

```

```
8.      if tipo_sensore == 'sensoreA':
9.
10.         #Plot per le rotazioni X scorrelate dall'umidità per il sensore di tipo A
11.         Grafici.ritardi_plot(percorso_file_risultati, nome_file_grafico, tipo_rot[0],  ore2, minuti2, parame
    tri[1], coefficienti_correlazione_mr2_kX_umid)
12.
13.         #Plot per le rotazioni Y scorrelate dalla temperatura per il sensore di tipo A
14.         Grafici.ritardi_plot(percorso_file_risultati, nome_file_grafico, tipo_rot[1],  ore2, minuti2, parame
    tri[0], coefficienti_correlazione_mr2_kY_temp)
15.
16.         #Plot per le rotazioni Y scorrelate dall'umidità per il sensore di tipo A
17.         Grafici.ritardi_plot(percorso_file_risultati, nome_file_grafico, tipo_rot[1],  ore2, minuti2, parame
    tri[1], coefficienti_correlazione_mr2_kY_umid)
```

Figura 90: Blocco (23) del codice

La funzione Grafici.ritardi_plot importa gli stessi parametri estetici usati per i grafici nel blocco (18); poi si eseguono le seguenti operazioni:

- apertura di una figura e definizione delle sue dimensioni;
- definizione dell'asse delle ascisse (valori da 0 a -24);
- plottaggio dei coefficienti di correlazione relativi al primo sensore;
- definizione di una griglia;
- formattazione della legenda;
- formattazione del titolo del grafico;
- formattazione dei titoli degli assi (etichette);
- formattazione dei valori numerici sugli assi;
- definizione dei limiti dell'asse x;
- formattazione del titolo dell'asse x (etichetta);
- formattazione della posizione degli assi;
- salvataggio della figura;
- chiusura della figura.

```
1.  #FUNZIONE GRAFICI(4) --
    > Questa funzione calcola e plotta i coefficienti di correlazione di tutti i sensori in funzione del ritardo

2.  def ritardi_plot(cartella_di_salvataggio, nome_file, tipo_rot, ore2, minuti2, parametro, coefficienti_correl
    azione_mr2_k):
3.
4.      file1 =open(os.getcwd()+"\File di input utente.txt",'r')       #Apertura file di input utente
5.      numero_righe = file1.read().count('\n')                        #Conteggio numero di righe del file
6.
7.      for n in range(numero_righe):
8.
9.          lettura = getline(os.getcwd()+"\File di input utente.txt", n+1)    #Lettura della riga
10.         riga2 = lettura.split(' ')                                         #Elemento divisore riga
11.
12.         if riga2[0] == 'Altezza_grafico:':
13.
14.             h_grafico = int(riga2[1])                          #Altezza del grafico
15.
```

```
        elif riga2[0] == 'Larghezza_grafico:':

            l_grafico = int(riga2[1])                       #Larghezza del grafico

        elif riga2[0] == 'Dimensione_titolo:':

            dim_titolo = int(riga2[1])                      #Dimensione del titolo del grafico

        elif riga2[0] == 'Dimensioni_etichette_assi:':

            dim_etichette_assi = int(riga2[1])              #Dimensione delle etichette degli assi del grafico

        elif riga2[0] == 'Distanza_etichette_da_asse:':

            dist_etichette_assi = int(riga2[1])             #Distanza delle etichette dagli assi del grafico

        elif riga2[0] == 'Dimensioni_valori_numerici_assi:':

            dim_valori_assi = int(riga2[1])                 #Dimensione dei valori numerici degli assi del grafico

        elif riga2[0] == 'Spessore_di_linea_della_curva:':

            spess_curva = int(riga2[1])                     #Spessore della curva

    file1.close()                                           #Chiusura del file

    #Definizione della figura
    fig_cc=plt.figure(figsize=[l_grafico, h_grafico])

    #Definizione del subplot 1
    ax=fig_cc.add_subplot(111)

    #Definizione asse delle ascisse (ritardi da 0 a 24 ore)
    x=np.flip(np.arange(-24, 1))

    for j in range(np.shape(coefficienti_correlazione_mr2_k)[0]):

        #Estrazione delle ordinate come j-esima riga della matrice in cui sono salvati i coefficienti di correlazione
        y=coefficienti_correlazione_mr2_k[j, :]

        #Plot della curva relativa al j-esimo sensore
        ax.plot(x, y)

    #Inserimento della griglia all'interno del grafico per migliorarne la leggibilità
    plt.grid()

    #Formattazione della legenda del grafico
    plt.legend(labels = nome_file, frameon=True, fontsize =  'xx-large')

    #Formattazione del titolo del grafico
    plt.title('Correlation coefficient between moving average rotation k on {} hours {} minutes'.format(ore2
, minuti2) + '\n' + 'and average {} on 2 hours with delays in range from 0 to -24h'.format(parametro[:-
4]) + ' (rotation {})'.format(tipo_rot), fontweight = 'bold', fontsize = dim_titolo)

    #Formattazione etichetta asse delle ascisse
    plt.xlabel('Delay [hours]', labelpad = dist_etichette_assi, fontsize = dim_etichette_assi, style = 'ital
ic')

    #Formattazione etichetta asse delle ordinate
    plt.ylabel('Correlation coefficient []', labelpad = dist_etichette_assi, fontsize = dim_etichette_assi,
style = 'italic')

    #Definizione dell'intervallo dei valori numerici per l'asse delle ascisse
    plt.xticks(range(-24, 1, 3))

    #Formattazione dei valori numerici sull'asse delle ascisse
```

```
78.     plt.tick_params('both', labelsize =  dim_valori_assi)
79.
80.     #Definizione limiti asse delle ascisse
81.     plt.xlim((-24, 0))
82.
83.     #Posizione legenda
84.     ax.spines['bottom'].set_position('zero')
85.     ax.spines['top'].set_color('none')
86.     ax.spines['right'].set_color('none')
87.
88.     #Salvataggio figura
89.     fig_cc.savefig(cartella_di_salvataggio + '\\' + 'Corr. coeff. with delayed {}'.format(parametro[:-
    4]) + '(rotation {})'.format(tipo_rot) + '.png', dpi=600)
90.
91.     #Chiusura figura
92.     plt.close(fig_cc)
```

Figura 91: Modulo esterno Grafici, funzione Grafici.ritardi_plot

4.8.17. Blocco (24): Cross-correlazione tra sensori

L'ultimo blocco di codice si occupa di calcolare i coefficienti di correlazione tra serie di dati relative a diversi sensori; per questo, viene eseguito soltanto se è stata data risposta affermativa alla relativa domanda nel file di input dell'utente.

Per ciascun gruppo di sensori tra cui effettuare la cross-correlazione, viene identificata la lista dei nomi dei sensori compresi.

Poi, per ciascun gruppo, viene creato il file Excel che conterrà i coefficienti di correlazione cercati, tramite la funzione Cross_correlazione.crea_excel. Inoltre, con questa funzione vengono scritte le intestazioni delle tabelle che verranno riempite con i coefficienti di correlazione tra i sensori. Gli argomenti richiesti sono:

- il tipo di sensore (*tipo_sensore*);
- la lista dei sensori appartenenti al gruppo in esame (*nome_file_cross_correlazione*);
- la durata in ore e minuti dei tre intervalli su cui effettuare le medie mobili (*ore1*, *minuti1*, *ore2*, *minuti2*, *ore3*, minuti3).

Quindi, come accennato già al capitolo 4.5, per ogni gruppo di sensori viene creato un file Excel, su ognuno di questi ultimi sono presenti tre fogli, relativi rispettivamente alle rotazioni grezze, corrette e doppiamente corrette. In ogni foglio sono presenti quattro tabelle per ogni tipo di rotazione (per cui, sensori di tipo B ne avranno quattro riferite alle sole rotazioni X, mentre sensori A ne avranno quattro riferiti a rotazioni X e quattro riferiti a rotazioni Y), riferite rispettivamente a rotazioni istantanee e medie mobili sui tre intervalli temporali definiti dall'utente.

Le righe e le colonne delle tabelle si riferiscono ai diversi sensori, mentre i valori interni rappresentano le matrici dei coefficienti di correlazione; la diagonale rimane vuota. Inoltre, vengono calcolati anche:

- per ogni sensore, la media dei coefficienti di correlazione tra esso e tutti gli altri sensori;
- la media dei coefficienti di correlazione di una famiglia;
- per ogni sensore, lo scarto percentuale tra la media dei coefficienti di correlazione relativi a esso e la media della famiglia.

Si ottiene in output la variabile contenente il file Excel generato e quelle contenenti i tre fogli che lo compongono.

```
1.  import numpy as np
2.  import pandas as pd
3.  import openpyxl as xl
4.
5.  from openpyxl.styles import Font, PatternFill, Border, Side, NamedStyle
6.
7.  stile1 = NamedStyle(name="stile1")
8.  stile1.font = Font(bold = True)
9.  stile1.border = Border(left=Side(style='medium', color='FF000000'), right=Side(style='medium', color='FF0000
    00'), top=Side(style='medium', color='FF000000'), bottom=Side(style='medium', color='FF000000'))
10. stile1.fill = PatternFill(start_color='D1D0CE', end_color='D1D0CE', fill_type='solid')
11.
12. stile2 = NamedStyle(name='stile2')
13. stile2.border = Border(left=Side(style='thin', color='FF000000'), right=Side(style='thin', color='FF000000')
    , top=Side(style='thin', color='FF000000'), bottom=Side(style='thin', color='FF000000'))
14.
15. from math import isnan
16.
17.
18. def crea_excel(tipo_sensore, nome_file_cross_correlazione, ore1, minuti1, ore2, minuti2, ore3, minuti3):
19.
20.     #Conteggio del numero dei sensori appartenenenti al gruppo
21.     N_sens_cross_correlati = len(nome_file_cross_correlazione)
22.
23.     #Creazione del file Excel e dei suoi fogli
24.     exc1 = xl.Workbook()
25.     foglio_ccrawsens = exc1.active
26.     foglio_ccrawsens.title = "coeff. corr. tra sensori (raw)"
27.     foglio_ccksens = exc1.create_sheet("coeff. corr. tra sensori (k)")
28.     foglio_cckksens = exc1.create_sheet("coeff. corr. tra sensori (kk)")
29.
30.     #Scrittura delle intestazioni delle tabelle
31.     for indice_sensore in range(N_sens_cross_correlati):
32.
33.         foglio_ccrawsens.cell(1, 1, 'ROT. ISTANTANEE X').style=stile1
34.         foglio_ccksens.cell(1, 1, 'ROT. ISTANTANEE X').style='stile1'
35.         foglio_cckksens.cell(1, 1, 'ROT. ISTANTANEE X').style='stile1'
36.         foglio_ccrawsens.cell(indice_sensore+2, 1, nome_file_cross_correlazione[indice_sensore]).style='stil
    e1'
37.         foglio_ccksens.cell(indice_sensore+2, 1, nome_file_cross_correlazione[indice_sensore]).style='stile1
    '
38.         foglio_cckksens.cell(indice_sensore+2, 1, nome_file_cross_correlazione[indice_sensore]).style='stile
    1'
39.         foglio_ccrawsens.cell(N_sens_cross_correlati+3, 1, 'MEDIA DEL SENSORE').style='stile1'
40.         foglio_ccksens.cell(N_sens_cross_correlati+3, 1, 'MEDIA DEL SENSORE').style='stile1'
41.         foglio_cckksens.cell(N_sens_cross_correlati+3, 1, 'MEDIA DEL SENSORE').style='stile1'
42.         foglio_ccrawsens.cell(N_sens_cross_correlati+4, 1, 'SCARTO [%]').style='stile1'
```

```
43.         foglio_ccksens.cell(N_sens_cross_correlati+4, 1, 'SCARTO [%]').style='stile1'
44.         foglio_cckksens.cell(N_sens_cross_correlati+4, 1, 'SCARTO [%]').style='stile1'
45.         foglio_ccrawsens.cell(N_sens_cross_correlati+5, 1, 'MEDIA TOTALE').style='stile1'
46.         foglio_ccksens.cell(N_sens_cross_correlati+5, 1, 'MEDIA TOTALE').style='stile1'
47.         foglio_cckksens.cell(N_sens_cross_correlati+5, 1, 'MEDIA TOTALE').style='stile1'
48.         foglio_ccrawsens.cell(1, indice_sensore+2, nome_file_cross_correlazione[indice_sensore]).style='stil
    e1'
49.         foglio_ccksens.cell(1, indice_sensore+2, nome_file_cross_correlazione[indice_sensore]).style='stile1
    '
50.         foglio_cckksens.cell(1, indice_sensore+2, nome_file_cross_correlazione[indice_sensore]).style='stile
    1'
51.
52.         foglio_ccrawsens.cell(1, N_sens_cross_correlati+3, 'ROT. MEDIE X SU {} ore {} minuti'.format(ore1, m
    inuti1)).style='stile1'
53.         foglio_ccksens.cell(1, N_sens_cross_correlati+3, 'ROT. MEDIE X SU {} ore {} minuti'.format(ore1, min
    uti1)).style='stile1'
54.         foglio_cckksens.cell(1, N_sens_cross_correlati+3, 'ROT. MEDIE X SU {} ore {} minuti'.format(ore1, mi
    nuti1)).style='stile1'
55.         foglio_ccrawsens.cell(indice_sensore+2, N_sens_cross_correlati+3, nome_file_cross_correlazione[indic
    e_sensore]).style='stile1'
56.         foglio_ccksens.cell(indice_sensore+2, N_sens_cross_correlati+3, nome_file_cross_correlazione[indice_
    sensore]).style='stile1'
57.         foglio_cckksens.cell(indice_sensore+2, N_sens_cross_correlati+3, nome_file_cross_correlazione[indice
    _sensore]).style='stile1'
58.         foglio_ccrawsens.cell(N_sens_cross_correlati+3, N_sens_cross_correlati+3, 'MEDIA DEL SENSORE').style
    ='stile1'
59.         foglio_ccksens.cell(N_sens_cross_correlati+3, N_sens_cross_correlati+3, 'MEDIA DEL SENSORE').style='
    stile1'
60.         foglio_cckksens.cell(N_sens_cross_correlati+3, N_sens_cross_correlati+3, 'MEDIA DEL SENSORE').style=
    'stile1'
61.         foglio_ccrawsens.cell(N_sens_cross_correlati+4, N_sens_cross_correlati+3, 'SCARTO [%]').style='stile
    1'
62.         foglio_ccksens.cell(N_sens_cross_correlati+4, N_sens_cross_correlati+3, 'SCARTO [%]').style='stile1'

63.         foglio_cckksens.cell(N_sens_cross_correlati+4, N_sens_cross_correlati+3, 'SCARTO [%]').style='stile1
    '
64.         foglio_ccrawsens.cell(N_sens_cross_correlati+5, N_sens_cross_correlati+3, 'MEDIA TOTALE').style='sti
    le1'
65.         foglio_ccksens.cell(N_sens_cross_correlati+5, N_sens_cross_correlati+3, 'MEDIA TOTALE').style='stile
    1'
66.         foglio_cckksens.cell(N_sens_cross_correlati+5, N_sens_cross_correlati+3, 'MEDIA TOTALE').style='stil
    e1'
67.         foglio_ccrawsens.cell(1, N_sens_cross_correlati+4+indice_sensore, nome_file_cross_correlazione[indic
    e_sensore]).style='stile1'
68.         foglio_ccksens.cell(1, N_sens_cross_correlati+4+indice_sensore, nome_file_cross_correlazione[indice_
    sensore]).style='stile1'
69.         foglio_cckksens.cell(1, N_sens_cross_correlati+4+indice_sensore, nome_file_cross_correlazione[indice
    _sensore]).style='stile1'
70.
71.         foglio_ccrawsens.cell(1, 2*N_sens_cross_correlati+5, 'ROT. MEDIE X SU {} ore {} minuti'.format(ore2,
     minuti2)).style='stile1'
72.         foglio_ccksens.cell(1, 2*N_sens_cross_correlati+5, 'ROT. MEDIE X SU {} ore {} minuti'.format(ore2, m
    inuti2)).style='stile1'
73.         foglio_cckksens.cell(1, 2*N_sens_cross_correlati+5, 'ROT. MEDIE X SU {} ore {} minuti'.format(ore2,
    minuti2)).style='stile1'
74.         foglio_ccrawsens.cell(indice_sensore+2, 2*N_sens_cross_correlati+5, nome_file_cross_correlazione[ind
    ice_sensore]).style='stile1'
75.         foglio_ccksens.cell(indice_sensore+2, 2*N_sens_cross_correlati+5, nome_file_cross_correlazione[indic
    e_sensore]).style='stile1'
76.         foglio_cckksens.cell(indice_sensore+2, 2*N_sens_cross_correlati+5, nome_file_cross_correlazione[indi
    ce_sensore]).style='stile1'
77.         foglio_ccrawsens.cell(N_sens_cross_correlati+3, 2*N_sens_cross_correlati+5, 'MEDIA DEL SENSORE').sty
    le='stile1'
78.         foglio_ccksens.cell(N_sens_cross_correlati+3, 2*N_sens_cross_correlati+5, 'MEDIA DEL SENSORE').style
    ='stile1'
```

```
79.         foglio_cckksens.cell(N_sens_cross_correlati+3, 2*N_sens_cross_correlati+5, 'MEDIA DEL SENSORE').styl
    e='stile1'
80.         foglio_ccrawsens.cell(N_sens_cross_correlati+4, 2*N_sens_cross_correlati+5, 'SCARTO [%]').style='sti
    le1'
81.         foglio_ccksens.cell(N_sens_cross_correlati+4, 2*N_sens_cross_correlati+5, 'SCARTO [%]').style='stile
    1'
82.         foglio_cckksens.cell(N_sens_cross_correlati+4, 2*N_sens_cross_correlati+5, 'SCARTO [%]').style='stil
    e1'
83.         foglio_ccrawsens.cell(N_sens_cross_correlati+5, 2*N_sens_cross_correlati+5, 'MEDIA TOTALE').style='s
    tile1'
84.         foglio_ccksens.cell(N_sens_cross_correlati+5, 2*N_sens_cross_correlati+5, 'MEDIA TOTALE').style='sti
    le1'
85.         foglio_cckksens.cell(N_sens_cross_correlati+5, 2*N_sens_cross_correlati+5, 'MEDIA TOTALE').style='st
    ile1'
86.         foglio_ccrawsens.cell(1, 2*N_sens_cross_correlati+6+indice_sensore, nome_file_cross_correlazione[ind
    ice_sensore]).style='stile1'
87.         foglio_ccksens.cell(1, 2*N_sens_cross_correlati+6+indice_sensore, nome_file_cross_correlazione[indic
    e_sensore]).style='stile1'
88.         foglio_cckksens.cell(1, 2*N_sens_cross_correlati+6+indice_sensore, nome_file_cross_correlazione[indi
    ce_sensore]).style='stile1'
89.
90.         foglio_ccrawsens.cell(1, 3*N_sens_cross_correlati+7, 'ROT. MEDIE X SU {} ore {} minuti'.format(ore3,
     minuti3)).style='stile1'
91.         foglio_ccksens.cell(1, 3*N_sens_cross_correlati+7, 'ROT. MEDIE X SU {} ore {} minuti'.format(ore3, m
    inuti3)).style='stile1'
92.         foglio_cckksens.cell(1, 3*N_sens_cross_correlati+7, 'ROT. MEDIE X SU {} ore {} minuti'.format(ore3,
    minuti3)).style='stile1'
93.         foglio_ccrawsens.cell(indice_sensore+2, 3*N_sens_cross_correlati+7, nome_file_cross_correlazione[ind
    ice_sensore]).style='stile1'
94.         foglio_ccksens.cell(indice_sensore+2, 3*N_sens_cross_correlati+7, nome_file_cross_correlazione[indic
    e_sensore]).style='stile1'
95.         foglio_cckksens.cell(indice_sensore+2, 3*N_sens_cross_correlati+7, nome_file_cross_correlazione[indi
    ce_sensore]).style='stile1'
96.         foglio_ccrawsens.cell(N_sens_cross_correlati+3, 3*N_sens_cross_correlati+7, 'MEDIA DEL SENSORE').sty
    le='stile1'
97.         foglio_ccksens.cell(N_sens_cross_correlati+3, 3*N_sens_cross_correlati+7, 'MEDIA DEL SENSORE').style
    ='stile1'
98.         foglio_cckksens.cell(N_sens_cross_correlati+3, 3*N_sens_cross_correlati+7, 'MEDIA DEL SENSORE').styl
    e='stile1'
99.         foglio_ccrawsens.cell(N_sens_cross_correlati+4, 3*N_sens_cross_correlati+7, 'SCARTO [%]').style='sti
    le1'
100.        foglio_ccksens.cell(N_sens_cross_correlati+4, 3*N_sens_cross_correlati+7, 'SCARTO [%]').style='stile
    1'
101.        foglio_cckksens.cell(N_sens_cross_correlati+4, 3*N_sens_cross_correlati+7, 'SCARTO [%]').style='stil
    e1'
102.        foglio_ccrawsens.cell(N_sens_cross_correlati+5, 3*N_sens_cross_correlati+7, 'MEDIA TOTALE').style='s
    tile1'
103.        foglio_ccksens.cell(N_sens_cross_correlati+5, 3*N_sens_cross_correlati+7, 'MEDIA TOTALE').style='sti
    le1'
104.        foglio_cckksens.cell(N_sens_cross_correlati+5, 3*N_sens_cross_correlati+7, 'MEDIA TOTALE').style='st
    ile1'
105.        foglio_ccrawsens.cell(1, 3*N_sens_cross_correlati+8+indice_sensore, nome_file_cross_correlazione[ind
    ice_sensore]).style='stile1'
106.        foglio_ccksens.cell(1, 3*N_sens_cross_correlati+8+indice_sensore, nome_file_cross_correlazione[indic
    e_sensore]).style='stile1'
107.        foglio_cckksens.cell(1, 3*N_sens_cross_correlati+8+indice_sensore, nome_file_cross_correlazione[indi
    ce_sensore]).style='stile1'
108.
109.
110.
111.        if tipo_sensore == 'sensoreA':
112.
113.            foglio_ccrawsens.cell(N_sens_cross_correlati+8, 1, 'ROT. ISTANTANEE Y').style='stile1'
114.            foglio_ccksens.cell(N_sens_cross_correlati+8, 1, 'ROT. ISTANTANEE Y').style='stile1'
115.            foglio_cckksens.cell(N_sens_cross_correlati+8, 1, 'ROT. ISTANTANEE Y').style='stile1'
```

```
116.            foglio_ccrawsens.cell(N_sens_cross_correlati+indice_sensore+9, 1, nome_file_cross_correlazione[i
    ndice_sensore]).style='stile1'
117.            foglio_ccksens.cell(N_sens_cross_correlati+indice_sensore+9, 1, nome_file_cross_correlazione[ind
    ice_sensore]).style='stile1'
118.            foglio_cckksens.cell(N_sens_cross_correlati+indice_sensore+9, 1, nome_file_cross_correlazione[in
    dice_sensore]).style='stile1'
119.            foglio_ccrawsens.cell(2*N_sens_cross_correlati+10, 1, 'MEDIA DEL SENSORE').style='stile1'
120.            foglio_ccksens.cell(2*N_sens_cross_correlati+10, 1, 'MEDIA DEL SENSORE').style='stile1'
121.            foglio_cckksens.cell(2*N_sens_cross_correlati+10, 1, 'MEDIA DEL SENSORE').style='stile1'
122.            foglio_ccrawsens.cell(2*N_sens_cross_correlati+11, 1, 'SCARTO [%]').style='stile1'
123.            foglio_ccksens.cell(2*N_sens_cross_correlati+11, 1, 'SCARTO [%]').style='stile1'
124.            foglio_cckksens.cell(2*N_sens_cross_correlati+11, 1, 'SCARTO [%]').style='stile1'
125.            foglio_ccrawsens.cell(2*N_sens_cross_correlati+12, 1, 'MEDIA TOTALE').style='stile1'
126.            foglio_ccksens.cell(2*N_sens_cross_correlati+12, 1, 'MEDIA TOTALE').style='stile1'
127.            foglio_cckksens.cell(2*N_sens_cross_correlati+12, 1, 'MEDIA TOTALE').style='stile1'
128.            foglio_ccrawsens.cell(N_sens_cross_correlati+8, indice_sensore+2, nome_file_cross_correlazione[i
    ndice_sensore]).style='stile1'
129.            foglio_ccksens.cell(N_sens_cross_correlati+8, indice_sensore+2, nome_file_cross_correlazione[ind
    ice_sensore]).style='stile1'
130.            foglio_cckksens.cell(N_sens_cross_correlati+8, indice_sensore+2, nome_file_cross_correlazione[in
    dice_sensore]).style='stile1'
131.
132.            foglio_ccrawsens.cell(N_sens_cross_correlati+8, N_sens_cross_correlati+3, 'ROT. MEDIE Y SU {} or
    e {} minuti'.format(ore1, minuti1)).style='stile1'
133.            foglio_ccksens.cell(N_sens_cross_correlati+8, N_sens_cross_correlati+3, 'ROT. MEDIE Y SU {} ore
    {} minuti'.format(ore1, minuti1)).style='stile1'
134.            foglio_cckksens.cell(N_sens_cross_correlati+8, N_sens_cross_correlati+3, 'ROT. MEDIE Y SU {} ore
     {} minuti'.format(ore1, minuti1)).style='stile1'
135.            foglio_ccrawsens.cell(N_sens_cross_correlati+indice_sensore+9, N_sens_cross_correlati+3, nome_fi
    le_cross_correlazione[indice_sensore]).style='stile1'
136.            foglio_ccksens.cell(N_sens_cross_correlati+indice_sensore+9, N_sens_cross_correlati+3, nome_file
    _cross_correlazione[indice_sensore]).style='stile1'
137.            foglio_cckksens.cell(N_sens_cross_correlati+indice_sensore+9, N_sens_cross_correlati+3, nome_fil
    e_cross_correlazione[indice_sensore]).style='stile1'
138.            foglio_ccrawsens.cell(2*N_sens_cross_correlati+10, N_sens_cross_correlati+3, 'MEDIA DEL SENSORE'
    ).style='stile1'
139.            foglio_ccksens.cell(2*N_sens_cross_correlati+10, N_sens_cross_correlati+3, 'MEDIA DEL SENSORE').
    style='stile1'
140.            foglio_cckksens.cell(2*N_sens_cross_correlati+10, N_sens_cross_correlati+3, 'MEDIA DEL SENSORE')
    .style='stile1'
141.            foglio_ccrawsens.cell(2*N_sens_cross_correlati+11, N_sens_cross_correlati+3, 'SCARTO [%]').style
    ='stile1'
142.            foglio_ccksens.cell(2*N_sens_cross_correlati+11, N_sens_cross_correlati+3, 'SCARTO [%]').style='
    stile1'
143.            foglio_cckksens.cell(2*N_sens_cross_correlati+11, N_sens_cross_correlati+3, 'SCARTO [%]').style=
    'stile1'
144.            foglio_ccrawsens.cell(2*N_sens_cross_correlati+12, N_sens_cross_correlati+3, 'MEDIA TOTALE').sty
    le='stile1'
145.            foglio_ccksens.cell(2*N_sens_cross_correlati+12, N_sens_cross_correlati+3, 'MEDIA TOTALE').style
    ='stile1'
146.            foglio_cckksens.cell(2*N_sens_cross_correlati+12, N_sens_cross_correlati+3, 'MEDIA TOTALE').styl
    e='stile1'
147.            foglio_ccrawsens.cell(N_sens_cross_correlati+8, N_sens_cross_correlati+4+indice_sensore, nome_fi
    le_cross_correlazione[indice_sensore]).style='stile1'
148.            foglio_ccksens.cell(N_sens_cross_correlati+8, N_sens_cross_correlati+4+indice_sensore, nome_file
    _cross_correlazione[indice_sensore]).style='stile1'
149.            foglio_cckksens.cell(N_sens_cross_correlati+8, N_sens_cross_correlati+4+indice_sensore, nome_fil
    e_cross_correlazione[indice_sensore]).style='stile1'
150.
151.            foglio_ccrawsens.cell(N_sens_cross_correlati+8, 2*N_sens_cross_correlati+5, 'ROT. MEDIE Y SU {}
    ore {} minuti'.format(ore2, minuti2)).style='stile1'
152.            foglio_ccksens.cell(N_sens_cross_correlati+8, 2*N_sens_cross_correlati+5, 'ROT. MEDIE Y SU {} or
    e {} minuti'.format(ore2, minuti2)).style='stile1'
153.            foglio_cckksens.cell(N_sens_cross_correlati+8, 2*N_sens_cross_correlati+5, 'ROT. MEDIE Y SU {} o
    re {} minuti'.format(ore2, minuti2)).style='stile1'
```

```
154.            foglio_ccrawsens.cell(N_sens_cross_correlati+indice_sensore+9, 2*N_sens_cross_correlati+5, nome_
    file_cross_correlazione[indice_sensore]).style='stile1'
155.            foglio_ccksens.cell(N_sens_cross_correlati+indice_sensore+9, 2*N_sens_cross_correlati+5, nome_fi
    le_cross_correlazione[indice_sensore]).style='stile1'
156.            foglio_cckksens.cell(N_sens_cross_correlati+indice_sensore+9, 2*N_sens_cross_correlati+5, nome_f
    ile_cross_correlazione[indice_sensore]).style='stile1'
157.            foglio_ccrawsens.cell(2*N_sens_cross_correlati+10, 2*N_sens_cross_correlati+5, 'MEDIA DEL SENSOR
    E').style='stile1'
158.            foglio_ccksens.cell(2*N_sens_cross_correlati+10, 2*N_sens_cross_correlati+5, 'MEDIA DEL SENSORE'
    ).style='stile1'
159.            foglio_cckksens.cell(2*N_sens_cross_correlati+10, 2*N_sens_cross_correlati+5, 'MEDIA DEL SENSORE
    ').style='stile1'
160.            foglio_ccrawsens.cell(2*N_sens_cross_correlati+11, 2*N_sens_cross_correlati+5, 'SCARTO [%]').sty
    le='stile1'
161.            foglio_ccksens.cell(2*N_sens_cross_correlati+11, 2*N_sens_cross_correlati+5, 'SCARTO [%]').style
    ='stile1'
162.            foglio_cckksens.cell(2*N_sens_cross_correlati+11, 2*N_sens_cross_correlati+5, 'SCARTO [%]').styl
    e='stile1'
163.            foglio_ccrawsens.cell(2*N_sens_cross_correlati+12, 2*N_sens_cross_correlati+5, 'MEDIA TOTALE').s
    tyle='stile1'
164.            foglio_ccksens.cell(2*N_sens_cross_correlati+12, 2*N_sens_cross_correlati+5, 'MEDIA TOTALE').sty
    le='stile1'
165.            foglio_cckksens.cell(2*N_sens_cross_correlati+12, 2*N_sens_cross_correlati+5, 'MEDIA TOTALE').st
    yle='stile1'
166.            foglio_ccrawsens.cell(N_sens_cross_correlati+8, 2*N_sens_cross_correlati+6+indice_sensore, nome_
    file_cross_correlazione[indice_sensore]).style='stile1'
167.            foglio_ccksens.cell(N_sens_cross_correlati+8, 2*N_sens_cross_correlati+6+indice_sensore, nome_fi
    le_cross_correlazione[indice_sensore]).style='stile1'
168.            foglio_cckksens.cell(N_sens_cross_correlati+8, 2*N_sens_cross_correlati+6+indice_sensore, nome_f
    ile_cross_correlazione[indice_sensore]).style='stile1'
169.
170.            foglio_ccrawsens.cell(N_sens_cross_correlati+8, 3*N_sens_cross_correlati+7, 'ROT. MEDIE Y SU {}
    ore {} minuti'.format(ore3, minuti3)).style='stile1'
171.            foglio_ccksens.cell(N_sens_cross_correlati+8, 3*N_sens_cross_correlati+7, 'ROT. MEDIE Y SU {} or
    e {} minuti'.format(ore3, minuti3)).style='stile1'
172.            foglio_cckksens.cell(N_sens_cross_correlati+8, 3*N_sens_cross_correlati+7, 'ROT. MEDIE Y SU {} o
    re {} minuti'.format(ore3, minuti3)).style='stile1'
173.            foglio_ccrawsens.cell(N_sens_cross_correlati+indice_sensore+9, 3*N_sens_cross_correlati+7, nome_
    file_cross_correlazione[indice_sensore]).style='stile1'
174.            foglio_ccksens.cell(N_sens_cross_correlati+indice_sensore+9, 3*N_sens_cross_correlati+7, nome_fi
    le_cross_correlazione[indice_sensore]).style='stile1'
175.            foglio_cckksens.cell(N_sens_cross_correlati+indice_sensore+9, 3*N_sens_cross_correlati+7, nome_f
    ile_cross_correlazione[indice_sensore]).style='stile1'
176.            foglio_ccrawsens.cell(2*N_sens_cross_correlati+10, 3*N_sens_cross_correlati+7, 'MEDIA DEL SENSOR
    E').style='stile1'
177.            foglio_ccksens.cell(2*N_sens_cross_correlati+10, 3*N_sens_cross_correlati+7, 'MEDIA DEL SENSORE'
    ).style='stile1'
178.            foglio_cckksens.cell(2*N_sens_cross_correlati+10, 3*N_sens_cross_correlati+7, 'MEDIA DEL SENSORE
    ').style='stile1'
179.            foglio_ccrawsens.cell(2*N_sens_cross_correlati+11, 3*N_sens_cross_correlati+7, 'SCARTO [%]').sty
    le='stile1'
180.            foglio_ccksens.cell(2*N_sens_cross_correlati+11, 3*N_sens_cross_correlati+7, 'SCARTO [%]').style
    ='stile1'
181.            foglio_cckksens.cell(2*N_sens_cross_correlati+11, 3*N_sens_cross_correlati+7, 'SCARTO [%]').styl
    e='stile1'
182.            foglio_ccrawsens.cell(2*N_sens_cross_correlati+12, 3*N_sens_cross_correlati+7, 'MEDIA TOTALE').s
    tyle='stile1'
183.            foglio_ccksens.cell(2*N_sens_cross_correlati+12, 3*N_sens_cross_correlati+7, 'MEDIA TOTALE').sty
    le='stile1'
184.            foglio_cckksens.cell(2*N_sens_cross_correlati+12, 3*N_sens_cross_correlati+7, 'MEDIA TOTALE').st
    yle='stile1'
185.            foglio_ccrawsens.cell(N_sens_cross_correlati+8, 3*N_sens_cross_correlati+8+indice_sensore, nome_
    file_cross_correlazione[indice_sensore]).style='stile1'
186.            foglio_ccksens.cell(N_sens_cross_correlati+8, 3*N_sens_cross_correlati+8+indice_sensore, nome_fi
    le_cross_correlazione[indice_sensore]).style='stile1'
```

```
            foglio_cckksens.cell(N_sens_cross_correlati+8, 3*N_sens_cross_correlati+8+indice_sensore, nome_file_cross_correlazione[indice_sensore]).style='stile1'

    return exc1, foglio_ccrawsens, foglio_ccksens, foglio_cckksens
```

Figura 92: Modulo esterno Cross_correlazione, funzione Cross_correlazione.crea_excel

Sul codice principale, poi, i DataFrame descritti al capitolo 4.8.14 vengono trasformati in numpy.array aventi due dimensioni, ovvero in matrici le cui colonne contengono le serie di una certa variabile, relative ai diversi sensori. Ciò avviene per mezzo del metodo to_numpy, applicabili a variabili di tipo DataFrame.

Finalmente si procede con il calcolo dei coefficienti di correlazione tra sensori; la funzione Cross_correlazione.cross_corr è atta a questo scopo. Devono essere inseriti i seguenti argomenti:

- il tipo di rotazione (*tipo_rot[0]* per rotazioni X, *tipo_rot[1]* per rotazioni Y);
- le matrici relative alle diverse variabili, relative al gruppo k (quindi gli elementi corrispondenti alla chiave “gruppok” dei relativi dizionari).
- i fogli creati nel nuovo file Excel dei risultati della cross-correlazione.

Viene definito il numero di sensori da cross-correlare, calcolando il numero delle colonne delle matrici appena create; poi si calcola il massimo numero di Not A Number presenti nelle serie di dati, e per semplicità si usa quello come indice iniziale dell’intervallo di valori che si usa per la cross-correlazione (*C*). Viene eliminato un uguale numero di valori anche alla fine della serie di dati, ovvero si definisce anche l’indice dell’ultimo valore da considerare nel calcolo dei coefficienti di correlazione (*D*).

Vengono inizializzate le matrici che conterranno i coefficienti di correlazione; sono matrici con un numero di righe pari al numero delle colonne, pari al numero dei sensori appartenenti al gruppo in esame. Ovviamente saranno matrici simmetriche.

```
def cross_corr(tipo_rot, rotazioni_grezze, media_mobile_1, media_mobile_2, media_mobile_3, rotazioni_corrette, media_mobile_1_k, media_mobile_2_k, media_mobile_3_k, rotazioni_corrette_due_volte, media_mobile_1_kk, media_mobile_2_kk, media_mobile_3_kk, foglio_ccrawsens, foglio_ccksens, foglio_cckksens):

    N_sens_cross_correlati = rotazioni_grezze.shape[1]       #Calcolo dei sensori da crosscorrelare nel gruppo
    n_nan_max = 0
    for k in range(N_sens_cross_correlati):
        n_nan_kesimo = np.isnan(media_mobile_3_kk[:,k])[np.isnan(media_mobile_3_kk[:,k])== True].size
        if n_nan_kesimo > n_nan_max:
            n_nan_max = n_nan_kesimo                         #Calcolo del numero di nan
    C = n_nan_max                                  #Indice del primo elemento della finestra di elementi da correlare
    D = rotazioni_grezze.shape[0] - C              #Indice dell’ultimo elemento della finestra di elementi da correlare

```

```
    #Creazione delle matrici vuote
    corr_raw_sens=np.full((N_sens_cross_correlati, N_sens_cross_correlati), 0.0)
    corr_k_sens=np.full((N_sens_cross_correlati, N_sens_cross_correlati), 0.0)
    corr_kk_sens=np.full((N_sens_cross_correlati, N_sens_cross_correlati), 0.0)

    corr_mr1_sens=np.full((N_sens_cross_correlati, N_sens_cross_correlati), 0.0)
    corr_mr1k_sens=np.full((N_sens_cross_correlati, N_sens_cross_correlati), 0.0)
    corr_mr1kk_sens=np.full((N_sens_cross_correlati, N_sens_cross_correlati), 0.0)

    corr_mr2_sens=np.full((N_sens_cross_correlati, N_sens_cross_correlati), 0.0)
    corr_mr2k_sens=np.full((N_sens_cross_correlati, N_sens_cross_correlati), 0.0)
    corr_mr2kk_sens=np.full((N_sens_cross_correlati, N_sens_cross_correlati), 0.0)

    corr_mr3_sens=np.full((N_sens_cross_correlati, N_sens_cross_correlati), 0.0)
    corr_mr3k_sens=np.full((N_sens_cross_correlati, N_sens_cross_correlati), 0.0)
    corr_mr3kk_sens=np.full((N_sens_cross_correlati, N_sens_cross_correlati), 0.0)

#Riempimento delle matrici; sulle diagonali vengono imposti nan
    for riga in range(0, N_sens_cross_correlati):

        for col in range(0, N_sens_cross_correlati):

            if riga != col:

                corr_raw_sens[riga, col]=np.corrcoef(rotazioni_grezze[:, riga][C:D], rotazioni_grezze[:, col][C:D])[1,0]
                corr_k_sens[riga, col]=np.corrcoef(rotazioni_corrette[:, riga][C:D], rotazioni_corrette[:, col][C:D])[1,0]
                corr_kk_sens[riga, col]=np.corrcoef(rotazioni_corrette_due_volte[:, riga][C:D], rotazioni_corrette_due_volte[:, col][C:D])[1,0]

                corr_mr1_sens[riga, col]=np.corrcoef(media_mobile_1[:, riga][C:D], media_mobile_1[:, col][C:D])[1,0]
                corr_mr1k_sens[riga, col]=np.corrcoef(media_mobile_1_k[:, riga][C:D], media_mobile_1_k[:, col][C:D])[1,0]
                corr_mr1kk_sens[riga, col]=np.corrcoef(media_mobile_1_kk[:, riga][C:D], media_mobile_1_kk[:, col][C:D])[1,0]

                corr_mr2_sens[riga, col]=np.corrcoef(media_mobile_2[:, riga][C:D], media_mobile_2[:, col][C:D])[1,0]
                corr_mr2k_sens[riga, col]=np.corrcoef(media_mobile_2_k[:, riga][C:D], media_mobile_2_k[:, col][C:D])[1,0]
                corr_mr2kk_sens[riga, col]=np.corrcoef(media_mobile_2_kk[:, riga][C:D], media_mobile_2_kk[:, col][C:D])[1,0]

                corr_mr3_sens[riga, col]=np.corrcoef(media_mobile_3[:, riga][C:D], media_mobile_3[:, col][C:D])[1,0]
                corr_mr3k_sens[riga, col]=np.corrcoef(media_mobile_3_k[:, riga][C:D], media_mobile_3_k[:, col][C:D])[1,0]
                corr_mr3kk_sens[riga, col]=np.corrcoef(media_mobile_3_kk[:, riga][C:D], media_mobile_3_kk[:, col][C:D])[1,0]

            else:

                corr_raw_sens[riga, col]=np.nan
                corr_k_sens[riga, col]=np.nan
                corr_kk_sens[riga, col]=np.nan

                corr_mr1_sens[riga, col]=np.nan
                corr_mr1k_sens[riga, col]=np.nan
                corr_mr1kk_sens[riga, col]=np.nan

                corr_mr2_sens[riga, col]=np.nan
                corr_mr2k_sens[riga, col]=np.nan
                corr_mr2kk_sens[riga, col]=np.nan

```

```
                corr_mr3_sens[riga, col]=np.nan
                corr_mr3k_sens[riga, col]=np.nan
                corr_mr3kk_sens[riga, col]=np.nan
```

Figura 93: Modulo esterno Cross_correlazione, prima parte della funzione Cross_correlazione.cross_corr

Di seguito, viene calcolato l'elemento (i, j) di queste matrici per mezzo di due cicli annidati, uno che faccia variare l'indice della riga, l'altro che faccia variare l'indice della colonna; tramite il comando numpy.corrcoef, quindi, si calcola il coefficiente di correlazione tra un intervallo di valori (da quello con indice *C* a quello con indice *D*) della serie di dati relativa all'i-esimo sensore e della serie di dati relativa al j-esimo sensore. In realtà, come già visto, il comando numpy.corrcoef restituisce una matrice di coefficienti di correlazione, perciò si considera l'elemento in posizione (1, 0), ovvero quello posizionato sulla seconda riga e sulla prima colonna. Così facendo, gli elementi sulla diagonale sono tutti pari a 1; questi vengono sovrascritti con Not A Number (nan).
Di seguito viene definita una funzione media che non restituisca errore nel caso vengano trovati elementi Not A Number tra gli oggetti di cui si vuol calcolare il valor medio.
Infine, si procede con la scrittura dei dati sul file Excel, usando due cicli annidati, uno che faccia variare l'indice della riga, e uno che faccia variare l'indice della colonna.

```
    #FUNZIONE MEDIA CHE NON CONSIDERI I NAN
    def mean(X):
        s=0
        n=0
        if len(X.shape)==1:
            for i in range(len(X)):
                if not(isnan(X[i])):
                    s=s+X[i]
                    n=n+1
        elif len(X.shape)==2:
            for i in range(X.shape[0]):
                for j in range(X.shape[1]):
                    if not(isnan(X[i, j])):
                        s=s+X[i, j]
                        n=n+1
        m=s/n
        return(m)


    if tipo_rot == 'X':

        for riga in range(0, N_sens_cross_correlati):

            for col in range(0, N_sens_cross_correlati):

                #TABELLE DELLE ROTAZIONI ISTANTANEE (ROTAZIONI X)
                foglio_ccrawsens.cell(riga+2, col+2, round(corr_raw_sens[riga, col], 3)).style=stile2
```

```
                foglio_ccrawsens.cell(N_sens_cross_correlati+3, col+2, round(mean(corr_raw_sens[:, col]), 3)
).style='stile2'
                foglio_ccrawsens.cell(N_sens_cross_correlati+5, 2, round(mean(corr_raw_sens), 3)).style='sti
le2'
                foglio_ccrawsens.cell(N_sens_cross_correlati+4, col+2, round((mean(corr_raw_sens[:, col])-
mean(corr_raw_sens))/mean(corr_raw_sens), 3)*100).style='stile2'
                foglio_ccksens.cell(riga+2, col+2, round(corr_k_sens[riga, col], 3)).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+3, col+2, round(mean(corr_k_sens[:, col]), 3)).st
yle='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+5, 2, round(mean(corr_k_sens), 3)).style='stile2'

                foglio_ccksens.cell(N_sens_cross_correlati+4, col+2, round((mean(corr_k_sens[:, col])-
mean(corr_k_sens))/mean(corr_k_sens), 3)*100).style='stile2'
                foglio_cckksens.cell(riga+2, col+2, round(corr_kk_sens[riga, col], 3)).style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+3, col+2, round(mean(corr_kk_sens[:, col]), 3)).
style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+5, 2, round(mean(corr_kk_sens), 3)).style='stile
2'
                foglio_cckksens.cell(N_sens_cross_correlati+4, col+2, round((mean(corr_kk_sens[:, col])-
mean(corr_kk_sens))/mean(corr_kk_sens), 3)*100).style='stile2'

                #TABELLE DELLE ROTAZIONI MEDIATE SUL PRIMO INTERVALLO DEFINITO DALL'UTENTE (ROTAZIONI X)
                foglio_ccrawsens.cell(riga+2, N_sens_cross_correlati+col+4, round(corr_mr1_sens[riga, col],
3)).style='stile2'
                foglio_ccrawsens.cell(N_sens_cross_correlati+3, N_sens_cross_correlati+col+4, round(mean(cor
r_mr1_sens[:, col]), 3)).style='stile2'
                foglio_ccrawsens.cell(N_sens_cross_correlati+5, N_sens_cross_correlati+4, round(mean(corr_mr
1_sens), 3)).style='stile2'
                foglio_ccrawsens.cell(N_sens_cross_correlati+4, N_sens_cross_correlati+col+4, round((mean(co
rr_mr1_sens[:, col])-mean(corr_mr1_sens))/mean(corr_mr1_sens), 3)*100).style='stile2'
                foglio_ccksens.cell(riga+2, N_sens_cross_correlati+col+4, round(corr_mr1k_sens[riga, col], 3
)).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+3, N_sens_cross_correlati+col+4, round(mean(corr_
mr1k_sens[:, col]), 3)).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+5, N_sens_cross_correlati+4, round(mean(corr_mr1k
_sens), 3)).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+4, N_sens_cross_correlati+col+4, round((mean(corr
_mr1k_sens[:, col])-mean(corr_mr1k_sens))/mean(corr_mr1k_sens), 3)*100).style='stile2'
                foglio_cckksens.cell(riga+2, N_sens_cross_correlati+col+4, round(corr_mr1kk_sens[riga, col],
 3)).style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+3, N_sens_cross_correlati+col+4, round(mean(corr
_mr1kk_sens[:, col]), 3)).style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+5, N_sens_cross_correlati+4, round(mean(corr_mr1
kk_sens), 3)).style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+4, N_sens_cross_correlati+col+4, round((mean(cor
r_mr1kk_sens[:, col])-mean(corr_mr1kk_sens))/mean(corr_mr1kk_sens), 3)*100).style='stile2'

                #TABELLE DELLE ROTAZIONI MEDIATE SUL SECONDO INTERVALLO DEFINITO DALL'UTENTE (ROTAZIONI X)
                foglio_ccrawsens.cell(riga+2, 2*N_sens_cross_correlati+col+6, round(corr_mr2_sens[riga, col]
, 3)).style='stile2'
                foglio_ccrawsens.cell(N_sens_cross_correlati+3, 2*N_sens_cross_correlati+col+6, round(mean(c
orr_mr2_sens[:, col]), 3)).style='stile2'
                foglio_ccrawsens.cell(N_sens_cross_correlati+5, 2*N_sens_cross_correlati+6, round(mean(corr_
mr2_sens), 3)).style='stile2'
                foglio_ccrawsens.cell(N_sens_cross_correlati+4, 2*N_sens_cross_correlati+col+6, round((mean(
corr_mr2_sens[:, col])-mean(corr_mr2_sens))/mean(corr_mr2_sens), 3)*100).style='stile2'
                foglio_ccksens.cell(riga+2, 2*N_sens_cross_correlati+col+6, round(corr_mr2k_sens[riga, col],
 3)).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+3, 2*N_sens_cross_correlati+col+6, round(mean(cor
r_mr2k_sens[:, col]), 3)).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+5, 2*N_sens_cross_correlati+6, round(mean(corr_mr
2k_sens), 3)).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+4, 2*N_sens_cross_correlati+col+6, round((mean(co
rr_mr2k_sens[:, col])-mean(corr_mr2k_sens))/mean(corr_mr2k_sens), 3)*100).style='stile2'
                foglio_cckksens.cell(riga+2, 2*N_sens_cross_correlati+col+6, round(corr_mr2kk_sens[riga, col
], 3)).style='stile2'
```

```
                foglio_cckksens.cell(N_sens_cross_correlati+3, 2*N_sens_cross_correlati+col+6, round(mean(corr_mr2kk_sens[:, col]), 3)).style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+5, 2*N_sens_cross_correlati+6, round(mean(corr_mr2kk_sens), 3)).style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+4, 2*N_sens_cross_correlati+col+6, round((mean(corr_mr2kk_sens[:, col])-mean(corr_mr2kk_sens))/mean(corr_mr2kk_sens), 3)*100).style='stile2'

                #TABELLE DELLE ROTAZIONI MEDIATE SUL TERZO INTERVALLO DEFINITO DALL'UTENTE (ROTAZIONI X)
                foglio_ccrawsens.cell(riga+2, 3*N_sens_cross_correlati+col+8, round(corr_mr3_sens[riga, col], 3)).style='stile2'
                foglio_ccrawsens.cell(N_sens_cross_correlati+3, 3*N_sens_cross_correlati+col+8, round(mean(corr_mr3_sens[:, col]), 3)).style='stile2'
                foglio_ccrawsens.cell(N_sens_cross_correlati+5, 3*N_sens_cross_correlati+8, round(mean(corr_mr3_sens), 3)).style='stile2'
                foglio_ccrawsens.cell(N_sens_cross_correlati+4, 3*N_sens_cross_correlati+col+8, round((mean(corr_mr3_sens[:, col])-mean(corr_mr3_sens))/mean(corr_mr3_sens), 3)*100).style='stile2'
                foglio_ccksens.cell(riga+2, 3*N_sens_cross_correlati+col+8, round(corr_mr3k_sens[riga, col], 3)).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+3, 3*N_sens_cross_correlati+col+8, round(mean(corr_mr3k_sens[:, col]), 3)).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+5, 3*N_sens_cross_correlati+8, round(mean(corr_mr3k_sens), 3)).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+4, 3*N_sens_cross_correlati+col+8, round((mean(corr_mr3k_sens[:, col])-mean(corr_mr3k_sens))/mean(corr_mr3k_sens), 3)*100).style='stile2'
                foglio_cckksens.cell(riga+2, 3*N_sens_cross_correlati+col+8, round(corr_mr3kk_sens[riga, col], 3)).style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+3, 3*N_sens_cross_correlati+col+8, round(mean(corr_mr3kk_sens[:, col]), 3)).style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+5, 3*N_sens_cross_correlati+8, round(mean(corr_mr3kk_sens), 3)).style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+4, 3*N_sens_cross_correlati+col+8, round((mean(corr_mr3kk_sens[:, col])-mean(corr_mr3kk_sens))/mean(corr_mr3kk_sens), 3)*100).style='stile2'


    if tipo_rot == 'Y':

        for riga in range(0, N_sens_cross_correlati):

            for col in range(0, N_sens_cross_correlati):

                #TABELLE DELLE ROTAZIONI ISTANTANEE (ROTAZIONI Y)
                foglio_ccrawsens.cell(N_sens_cross_correlati+riga+9, col+2, round(corr_raw_sens[riga, col], 3)).style='stile2'
                foglio_ccrawsens.cell(2*N_sens_cross_correlati+10, col+2, round(mean(corr_raw_sens[:, col]), 3)).style='stile2'
                foglio_ccrawsens.cell(2*N_sens_cross_correlati+12, 2, round(mean(corr_raw_sens), 3)).style='stile2'
                foglio_ccrawsens.cell(2*N_sens_cross_correlati+11, col+2, round((mean(corr_raw_sens[:, col])-mean(corr_raw_sens))/mean(corr_raw_sens), 3)*100).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+riga+9, col+2, round(corr_k_sens[riga, col], 3)).style='stile2'
                foglio_ccksens.cell(2*N_sens_cross_correlati+10, col+2, round(mean(corr_k_sens[:, col]), 3)).style='stile2'
                foglio_ccksens.cell(2*N_sens_cross_correlati+12, 2, round(mean(corr_k_sens), 3)).style='stile2'
                foglio_ccksens.cell(2*N_sens_cross_correlati+11, col+2, round((mean(corr_k_sens[:, col])-mean(corr_k_sens))/mean(corr_k_sens), 3)*100).style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+riga+9, col+2, round(corr_kk_sens[riga, col], 3)).style='stile2'
                foglio_cckksens.cell(2*N_sens_cross_correlati+10, col+2, round(mean(corr_kk_sens[:, col]), 3)).style='stile2'
                foglio_cckksens.cell(2*N_sens_cross_correlati+12, 2, round(mean(corr_kk_sens), 3)).style='stile2'
                foglio_cckksens.cell(2*N_sens_cross_correlati+11, col+2, round((mean(corr_kk_sens[:, col])-mean(corr_kk_sens))/mean(corr_kk_sens), 3)*100).style='stile2'

```

```
                #TABELLE DELLE ROTAZIONI MEDIATE SUL PRIMO INTERVALLO DEFINITO DALL'UTENTE (ROTAZIONI Y)
                foglio_ccrawsens.cell(N_sens_cross_correlati+riga+9, N_sens_cross_correlati+col+4, round(corr_mr1_sens[riga, col], 3)).style='stile2'
                foglio_ccrawsens.cell(2*N_sens_cross_correlati+10, N_sens_cross_correlati+col+4, round(mean(corr_mr1_sens[:, col]), 3)).style='stile2'
                foglio_ccrawsens.cell(2*N_sens_cross_correlati+12, N_sens_cross_correlati+4, round(mean(corr_mr1_sens), 3)).style='stile2'
                foglio_ccrawsens.cell(2*N_sens_cross_correlati+11, N_sens_cross_correlati+col+4, round((mean(corr_mr1_sens[:, col])-mean(corr_mr1_sens))/mean(corr_mr1_sens), 3)*100).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+riga+9, N_sens_cross_correlati+col+4, round(corr_mr1k_sens[riga, col], 3)).style='stile2'
                foglio_ccksens.cell(2*N_sens_cross_correlati+10, N_sens_cross_correlati+col+4, round(mean(corr_mr1k_sens[:, col]), 3)).style='stile2'
                foglio_ccksens.cell(2*N_sens_cross_correlati+12, N_sens_cross_correlati+4, round(mean(corr_mr1k_sens), 3)).style='stile2'
                foglio_ccksens.cell(2*N_sens_cross_correlati+11, N_sens_cross_correlati+col+4, round((mean(corr_mr1k_sens[:, col])-mean(corr_mr1k_sens))/mean(corr_mr1k_sens), 3)*100).style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+riga+9, N_sens_cross_correlati+col+4, round(corr_mr1kk_sens[riga, col], 3)).style='stile2'
                foglio_cckksens.cell(2*N_sens_cross_correlati+10, N_sens_cross_correlati+col+4, round(mean(corr_mr1kk_sens[:, col]), 3)).style='stile2'
                foglio_cckksens.cell(2*N_sens_cross_correlati+12, N_sens_cross_correlati+4, round(mean(corr_mr1kk_sens), 3)).style='stile2'
                foglio_cckksens.cell(2*N_sens_cross_correlati+11, N_sens_cross_correlati+col+4, round((mean(corr_mr1kk_sens[:, col])-mean(corr_mr1kk_sens))/mean(corr_mr1kk_sens), 3)*100).style='stile2'

                #TABELLE DELLE ROTAZIONI MEDIATE SUL SECONDO INTERVALLO DEFINITO DALL'UTENTE (ROTAZIONI Y)
                foglio_ccrawsens.cell(N_sens_cross_correlati+riga+9, 2*N_sens_cross_correlati+col+6, round(corr_mr2_sens[riga, col], 3)).style='stile2'
                foglio_ccrawsens.cell(2*N_sens_cross_correlati+10, 2*N_sens_cross_correlati+col+6, round(mean(corr_mr2_sens[:, col]), 3)).style='stile2'
                foglio_ccrawsens.cell(2*N_sens_cross_correlati+12, 2*N_sens_cross_correlati+6, round(mean(corr_mr2_sens), 3)).style='stile2'
                foglio_ccrawsens.cell(2*N_sens_cross_correlati+11, 2*N_sens_cross_correlati+col+6, round((mean(corr_mr2_sens[:, col])-mean(corr_mr2_sens))/mean(corr_mr2_sens), 3)*100).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+riga+9, 2*N_sens_cross_correlati+col+6, round(corr_mr2k_sens[riga, col], 3)).style='stile2'
                foglio_ccksens.cell(2*N_sens_cross_correlati+10, 2*N_sens_cross_correlati+col+6, round(mean(corr_mr2k_sens[:, col]), 3)).style='stile2'
                foglio_ccksens.cell(2*N_sens_cross_correlati+12, 2*N_sens_cross_correlati+6, round(mean(corr_mr2k_sens), 3)).style='stile2'
                foglio_ccksens.cell(2*N_sens_cross_correlati+11, 2*N_sens_cross_correlati+col+6, round((mean(corr_mr2k_sens[:, col])-mean(corr_mr2k_sens))/mean(corr_mr2k_sens), 3)*100).style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+riga+9, 2*N_sens_cross_correlati+col+6, round(corr_mr2kk_sens[riga, col], 3)).style='stile2'
                foglio_cckksens.cell(2*N_sens_cross_correlati+10, 2*N_sens_cross_correlati+col+6, round(mean(corr_mr2kk_sens[:, col]), 3)).style='stile2'
                foglio_cckksens.cell(2*N_sens_cross_correlati+12, 2*N_sens_cross_correlati+6, round(mean(corr_mr2kk_sens), 3)).style='stile2'
                foglio_cckksens.cell(2*N_sens_cross_correlati+11, 2*N_sens_cross_correlati+col+6, round((mean(corr_mr2kk_sens[:, col])-mean(corr_mr2kk_sens))/mean(corr_mr2kk_sens), 3)*100).style='stile2'

                #TABELLE DELLE ROTAZIONI MEDIATE SUL TERZO INTERVALLO DEFINITO DALL'UTENTE (ROTAZIONI Y)
                foglio_ccrawsens.cell(N_sens_cross_correlati+riga+9, 3*N_sens_cross_correlati+col+8, round(corr_mr3_sens[riga, col], 3)).style='stile2'
                foglio_ccrawsens.cell(2*N_sens_cross_correlati+10, 3*N_sens_cross_correlati+col+8, round(mean(corr_mr3_sens[:, col]), 3)).style='stile2'
                foglio_ccrawsens.cell(2*N_sens_cross_correlati+12, 3*N_sens_cross_correlati+8, round(mean(corr_mr3_sens), 3)).style='stile2'
                foglio_ccrawsens.cell(2*N_sens_cross_correlati+11, 3*N_sens_cross_correlati+col+8, round((mean(corr_mr3_sens[:, col])-mean(corr_mr3_sens))/mean(corr_mr3_sens), 3)*100).style='stile2'
                foglio_ccksens.cell(N_sens_cross_correlati+riga+9, 3*N_sens_cross_correlati+col+8, round(corr_mr3k_sens[riga, col], 3)).style='stile2'
                foglio_ccksens.cell(2*N_sens_cross_correlati+10, 3*N_sens_cross_correlati+col+8, round(mean(corr_mr3k_sens[:, col]), 3)).style='stile2'
```

```
                foglio_ccksens.cell(2*N_sens_cross_correlati+12, 3*N_sens_cross_correlati+8, round(mean(corr_mr3k_sens), 3)).style='stile2'
                foglio_ccksens.cell(2*N_sens_cross_correlati+11, 3*N_sens_cross_correlati+col+8, round((mean(corr_mr3k_sens[:, col])-mean(corr_mr3k_sens))/mean(corr_mr3k_sens), 3)*100).style='stile2'
                foglio_cckksens.cell(N_sens_cross_correlati+riga+9, 3*N_sens_cross_correlati+col+8, round(corr_mr3kk_sens[riga, col], 3)).style='stile2'
                foglio_cckksens.cell(2*N_sens_cross_correlati+10, 3*N_sens_cross_correlati+col+8, round(mean(corr_mr3kk_sens[:, col]), 3)).style='stile2'
                foglio_cckksens.cell(2*N_sens_cross_correlati+12, 3*N_sens_cross_correlati+8, round(mean(corr_mr3kk_sens), 3)).style='stile2'
                foglio_cckksens.cell(2*N_sens_cross_correlati+11, 3*N_sens_cross_correlati+col+8, round((mean(corr_mr3kk_sens[:, col])-mean(corr_mr3kk_sens))/mean(corr_mr3kk_sens), 3)*100).style='stile2'
```

Figura 94: Modulo esterno Cross_correlazione, seconda parte della funzione Cross_correlazione.cross_corr

Infine, nel codice principale si stampa il numero di elementi eliminati dalla linea temporale della correlazione creata nel blocco (1.4), anche in percentuale, in modo tale da poter allargare o restringere l'intorno in cui cercare i valori per i diversi sensori. Poi, si salva il file Excel che è stato creato.

Infine, il codice stampa il tempo totale impiegato per processare i dati, computato tramite la funzione time.process_time.

```
#   (24) CROSS-CORRELAZIONE TRA SENSORI

#Le operazioni successive vengon eseguite solo se nel file di input è stata data risposta affermativa alla domanda relativa
if cross_correlazione_tra_sensori == 'sì':

    for k in range(numero_gruppi_cross_correlazione):                                                    #Operazioni svolte per ogni gruppo di sensori da cross-correlare

        nome_file_cross_correlazione = dizionario_gruppi_cross_corr['gruppo{}'.format(k)]            #La lista dei sensori da cross-correlare nel k-esimo gruppo è memorizzata nel corrispondente dizionario, sotto la chiave "gruppok"

        #Creazione di un file Excel per ogni gruppo che conterrà le tabelle di cross-correlazione
        exc1, foglio_ccrawsens, foglio_ccksens, foglio_cckksens = Cross_correlazione.crea_excel(tipo_sensore, nome_file_cross_correlazione, ore1, minuti1, ore2, minuti2, ore3, minuti3)


        if tipo_sensore == 'sensoreA':

            for z in range(2):

                if z == 0:

                    #Creazione delle matrici che costituiscono ogni valore dei dizionari contenenti le serie di dati
                    diz_rotazioni_grezzeX_temp['gruppo{}'.format(k)] = diz_rotazioni_grezzeX_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_1X_temp['gruppo{}'.format(k)] = diz_media_mobile_1X_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_2X_temp['gruppo{}'.format(k)] = diz_media_mobile_2X_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_3X_temp['gruppo{}'.format(k)] = diz_media_mobile_3X_temp['gruppo{}'.format(k)].to_numpy()
```

```
                    diz_rotazioni_correttеX_temp['gruppo{}'.format(k)] = diz_rotazioni_correttеX_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_1_kX_temp['gruppo{}'.format(k)] = diz_media_mobile_1_kX_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_2_kX_temp['gruppo{}'.format(k)] = diz_media_mobile_2_kX_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_3_kX_temp['gruppo{}'.format(k)] = diz_media_mobile_3_kX_temp['gruppo{}'.format(k)].to_numpy()
                    diz_rotazioni_corrette_due_volteX_temp['gruppo{}'.format(k)] = diz_rotazioni_corrette_due_volteX_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_1_kkX_temp['gruppo{}'.format(k)] = diz_media_mobile_1_kkX_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_2_kkX_temp['gruppo{}'.format(k)] = diz_media_mobile_2_kkX_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_3_kkX_temp['gruppo{}'.format(k)] = diz_media_mobile_3_kkX_temp['gruppo{}'.format(k)].to_numpy()

                    diz_rotazioni_grezzeY_temp['gruppo{}'.format(k)] = diz_rotazioni_grezzeY_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_1Y_temp['gruppo{}'.format(k)] = diz_media_mobile_1Y_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_2Y_temp['gruppo{}'.format(k)] = diz_media_mobile_2Y_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_3Y_temp['gruppo{}'.format(k)] = diz_media_mobile_3Y_temp['gruppo{}'.format(k)].to_numpy()
                    diz_rotazioni_correttеY_temp['gruppo{}'.format(k)] = diz_rotazioni_correttеY_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_1_kY_temp['gruppo{}'.format(k)] = diz_media_mobile_1_kY_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_2_kY_temp['gruppo{}'.format(k)] = diz_media_mobile_2_kY_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_3_kY_temp['gruppo{}'.format(k)] = diz_media_mobile_3_kY_temp['gruppo{}'.format(k)].to_numpy()
                    diz_rotazioni_corrette_due_volteY_temp['gruppo{}'.format(k)] = diz_rotazioni_corrette_due_volteY_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_1_kkY_temp['gruppo{}'.format(k)] = diz_media_mobile_1_kkY_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_2_kkY_temp['gruppo{}'.format(k)] = diz_media_mobile_2_kkY_temp['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_3_kkY_temp['gruppo{}'.format(k)] = diz_media_mobile_3_kkY_temp['gruppo{}'.format(k)].to_numpy()

                    #Calcolo dei coefficienti di cross-correlazione tra le serie
                    Cross_correlazione.cross_corr(tipo_rot[0], diz_rotazioni_grezzeX_temp['gruppo{}'.format(k)], diz_media_mobile_1X_temp['gruppo{}'.format(k)], diz_media_mobile_2X_temp['gruppo{}'.format(k)], diz_media_mobile_3X_temp['gruppo{}'.format(k)], diz_rotazioni_correttеX_temp['gruppo{}'.format(k)], diz_media_mobile_1_kX_temp['gruppo{}'.format(k)], diz_media_mobile_2_kX_temp['gruppo{}'.format(k)], diz_media_mobile_3_kX_temp['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteX_temp['gruppo{}'.format(k)], diz_media_mobile_1_kkX_temp['gruppo{}'.format(k)], diz_media_mobile_2_kkX_temp['gruppo{}'.format(k)], diz_media_mobile_3_kkX_temp['gruppo{}'.format(k)], foglio_ccrawsens, foglio_ccksens, foglio_cckksens)
                    Cross_correlazione.cross_corr(tipo_rot[1], diz_rotazioni_grezzeY_temp['gruppo{}'.format(k)], diz_media_mobile_1Y_temp['gruppo{}'.format(k)], diz_media_mobile_2Y_temp['gruppo{}'.format(k)], diz_media_mobile_3Y_temp['gruppo{}'.format(k)], diz_rotazioni_correttеY_temp['gruppo{}'.format(k)], diz_media_mobile_1_kY_temp['gruppo{}'.format(k)], diz_media_mobile_2_kY_temp['gruppo{}'.format(k)], diz_media_mobile_3_kY_temp['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteY_temp['gruppo{}'.format(k)], diz_media_mobile_1_kkY_temp['gruppo{}'.format(k)], diz_media_mobile_2_kkY_temp['gruppo{}'.format(k)], diz_media_mobile_3_kkY_temp['gruppo{}'.format(k)], foglio_ccrawsens, foglio_ccksens, foglio_cckksens)

                    #Stampa del numero di valori  eliminati dalla serie temporale
                    print('{}° gruppo: elementi scartati per rotazioni X (dopo svincolamento da temperatura) : '.format(k+1))
                    print(len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)])-len(diz_timestamp_cross_corrX_temp['gruppo{}'.format(k)]), 'su', len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)]), '(', round((len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)])-len(diz_timestamp_cross_corrX_temp['gruppo{}'.format(k)]))/len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)]), 3)*100, '% )')
```

```
                    print('{}° gruppo: elementi scartati per rotazioni Y (dopo svincolamento da temperatura) : '.format(k+1))
                    print(len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)])-len(diz_timestamp_cross_corrY_temp['gruppo{}'.format(k)]), 'su', len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)]), '(', round((len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)])-len(diz_timestamp_cross_corrY_umid['gruppo{}'.format(k)]))/len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)]), 3)*100, '% )')

                    #Salvataggio dei coefficienti di cross-correlazione tra le serie
                    exc1.save(percorso_file_risultati + '\\Cross_correlazione tra sensori del {}° gruppo (svincolati da temperatura).xlsx'.format(k+1))

                if z == 1:

                    #Creazione delle matrici che costituiscono ogni valore dei dizionari contenenti le serie di dati
                    diz_rotazioni_grezzeX_umid['gruppo{}'.format(k)] = diz_rotazioni_grezzeX_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_1X_umid['gruppo{}'.format(k)] = diz_media_mobile_1X_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_2X_umid['gruppo{}'.format(k)] = diz_media_mobile_2X_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_3X_umid['gruppo{}'.format(k)] = diz_media_mobile_3X_umid['gruppo{}'.format(k)].to_numpy()
                    diz_rotazioni_correttеX_umid['gruppo{}'.format(k)] = diz_rotazioni_correttеX_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_1_kX_umid['gruppo{}'.format(k)] = diz_media_mobile_1_kX_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_2_kX_umid['gruppo{}'.format(k)] = diz_media_mobile_2_kX_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_3_kX_umid['gruppo{}'.format(k)] = diz_media_mobile_3_kX_umid['gruppo{}'.format(k)].to_numpy()
                    diz_rotazioni_corrette_due_volteX_umid['gruppo{}'.format(k)] = diz_rotazioni_corrette_due_volteX_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_1_kkX_umid['gruppo{}'.format(k)] = diz_media_mobile_1_kkX_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_2_kkX_umid['gruppo{}'.format(k)] = diz_media_mobile_2_kkX_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_3_kkX_umid['gruppo{}'.format(k)] = diz_media_mobile_3_kkX_umid['gruppo{}'.format(k)].to_numpy()

                    diz_rotazioni_grezzeY_umid['gruppo{}'.format(k)] = diz_rotazioni_grezzeY_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_1Y_umid['gruppo{}'.format(k)] = diz_media_mobile_1Y_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_2Y_umid['gruppo{}'.format(k)] = diz_media_mobile_2Y_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_3Y_umid['gruppo{}'.format(k)] = diz_media_mobile_3Y_umid['gruppo{}'.format(k)].to_numpy()
                    diz_rotazioni_correttеY_umid['gruppo{}'.format(k)] = diz_rotazioni_correttеY_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_1_kY_umid['gruppo{}'.format(k)] = diz_media_mobile_1_kY_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_2_kY_umid['gruppo{}'.format(k)] = diz_media_mobile_2_kY_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_3_kY_umid['gruppo{}'.format(k)] = diz_media_mobile_3_kY_umid['gruppo{}'.format(k)].to_numpy()
                    diz_rotazioni_corrette_due_volteY_umid['gruppo{}'.format(k)] = diz_rotazioni_corrette_due_volteY_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_1_kkY_umid['gruppo{}'.format(k)] = diz_media_mobile_1_kkY_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_2_kkY_umid['gruppo{}'.format(k)] = diz_media_mobile_2_kkY_umid['gruppo{}'.format(k)].to_numpy()
                    diz_media_mobile_3_kkY_umid['gruppo{}'.format(k)] = diz_media_mobile_3_kkY_umid['gruppo{}'.format(k)].to_numpy()

                    #Calcolo dei coefficienti di cross-correlazione tra le serie
```

```
90.                     Cross_correlazione.cross_corr(tipo_rot[0], diz_rotazioni_grezzeX_umid['gruppo{}'.format(
    k)], diz_media_mobile_1X_umid['gruppo{}'.format(k)], diz_media_mobile_2X_umid['gruppo{}'.format(k)], diz_med
    ia_mobile_3X_umid['gruppo{}'.format(k)], diz_rotazioni_corretteX_umid['gruppo{}'.format(k)], diz_media_mobil
    e_1_kX_umid['gruppo{}'.format(k)], diz_media_mobile_2_kX_umid['gruppo{}'.format(k)], diz_media_mobile_3_kX_u
    mid['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteX_umid['gruppo{}'.format(k)], diz_media_mobile_1_
    kkX_umid['gruppo{}'.format(k)], diz_media_mobile_2_kkX_umid['gruppo{}'.format(k)], diz_media_mobile_3_kkX_um
    id['gruppo{}'.format(k)], foglio_ccrawsens, foglio_ccksens, foglio_cckksens)
91.                     Cross_correlazione.cross_corr(tipo_rot[1], diz_rotazioni_grezzeY_umid['gruppo{}'.format(
    k)], diz_media_mobile_1Y_umid['gruppo{}'.format(k)], diz_media_mobile_2Y_umid['gruppo{}'.format(k)], diz_med
    ia_mobile_3Y_umid['gruppo{}'.format(k)], diz_rotazioni_corretteY_umid['gruppo{}'.format(k)], diz_media_mobil
    e_1_kY_umid['gruppo{}'.format(k)], diz_media_mobile_2_kY_umid['gruppo{}'.format(k)], diz_media_mobile_3_kY_u
    mid['gruppo{}'.format(k)], diz_rotazioni_corrette_due_volteY_umid['gruppo{}'.format(k)], diz_media_mobile_1_
    kkY_umid['gruppo{}'.format(k)], diz_media_mobile_2_kkY_umid['gruppo{}'.format(k)], diz_media_mobile_3_kkY_um
    id['gruppo{}'.format(k)], foglio_ccrawsens, foglio_ccksens, foglio_cckksens)
92.
93.                     #Stampa del numero di valori  eliminati dalla serie temporale
94.                     print('{}° gruppo: elementi scartati per rotazioni X (dopo svincolamento da temperatura
    e umidità): '.format(k+1))
95.                     print(len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)])-
    len(diz_timestamp_cross_corrX_umid['gruppo{}'.format(k)]), 'su', len(dizionario_timestamp_cross_corr['gruppo
    {}'.format(k)]), '(', round((len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)])-
    len(diz_timestamp_cross_corrX_umid['gruppo{}'.format(k)]))/len(dizionario_timestamp_cross_corr['gruppo{}'.fo
    rmat(k)]), 3)*100, '% )')
96.                     print('{}° gruppo: elementi scartati per rotazioni Y (dopo svincolamento da temperatura
    e umidità): '.format(k+1))
97.                     print(len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)])-
    len(diz_timestamp_cross_corrY_umid['gruppo{}'.format(k)]), 'su', len(dizionario_timestamp_cross_corr['gruppo
    {}'.format(k)]), '(', round((len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)])-
    len(diz_timestamp_cross_corrY_umid['gruppo{}'.format(k)]))/len(dizionario_timestamp_cross_corr['gruppo{}'.fo
    rmat(k)]), 3)*100, '% )')
98.
99.                     #Salvataggio dei coefficienti di cross-correlazione tra le serie
100.                    exc1.save(percorso_file_risultati + '\\Cross_correlazione tra sensori del {}° gruppo (sv
    incolati da temperatura e umidità).xlsx'.format(k+1))
101.
102.        if tipo_sensore == 'sensoreB':
103.
104.            #Creazione delle matrici che costituiscono ogni valore dei dizionari contenenti le serie di dati
105.            diz_rotazioni_grezzeX['gruppo{}'.format(k)] = diz_rotazioni_grezzeX['gruppo{}'.format(k)].to_num
    py()
106.            diz_media_mobile_1X['gruppo{}'.format(k)] = diz_media_mobile_1X['gruppo{}'.format(k)].to_numpy()

107.            diz_media_mobile_2X['gruppo{}'.format(k)] = diz_media_mobile_2X['gruppo{}'.format(k)].to_numpy()

108.            diz_media_mobile_3X['gruppo{}'.format(k)] = diz_media_mobile_3X['gruppo{}'.format(k)].to_numpy()

109.            diz_rotazioni_corretteX['gruppo{}'.format(k)] = diz_rotazioni_corretteX['gruppo{}'.format(k)].to
    _numpy()
110.            diz_media_mobile_1_kX['gruppo{}'.format(k)] = diz_media_mobile_1_kX['gruppo{}'.format(k)].to_num
    py()
111.            diz_media_mobile_2_kX['gruppo{}'.format(k)] = diz_media_mobile_2_kX['gruppo{}'.format(k)].to_num
    py()
112.            diz_media_mobile_3_kX['gruppo{}'.format(k)] = diz_media_mobile_3_kX['gruppo{}'.format(k)].to_num
    py()
113.            diz_rotazioni_corrette_due_volteX['gruppo{}'.format(k)] = diz_rotazioni_corrette_due_volteX['gru
    ppo{}'.format(k)].to_numpy()
114.            diz_media_mobile_1_kkX['gruppo{}'.format(k)] = diz_media_mobile_1_kkX['gruppo{}'.format(k)].to_n
    umpy()
115.            diz_media_mobile_2_kkX['gruppo{}'.format(k)] = diz_media_mobile_2_kkX['gruppo{}'.format(k)].to_n
    umpy()
116.            diz_media_mobile_3_kkX['gruppo{}'.format(k)] = diz_media_mobile_3_kkX['gruppo{}'.format(k)].to_n
    umpy()
117.
118.            #Calcolo dei coefficienti di cross-correlazione tra le serie
119.            Cross_correlazione.cross_corr(tipo_rot[0], diz_rotazioni_grezzeX['gruppo{}'.format(k)], diz_medi
    a_mobile_1X['gruppo{}'.format(k)], diz_media_mobile_2X['gruppo{}'.format(k)], diz_media_mobile_3X['gruppo{}'
```

```
    .format(k)], diz_rotazioni_correttex['gruppo{}'.format(k)], diz_media_mobile_1_kX['gruppo{}'.format(k)], diz
    _media_mobile_2_kX['gruppo{}'.format(k)], diz_media_mobile_3_kX['gruppo{}'.format(k)], diz_rotazioni_corrett
    e_due_volteX['gruppo{}'.format(k)], diz_media_mobile_1_kkX['gruppo{}'.format(k)], diz_media_mobile_2_kkX['gr
    uppo{}'.format(k)], diz_media_mobile_3_kkX['gruppo{}'.format(k)], foglio_ccrawsens, foglio_ccksens, foglio_c
    ckksens)
120.
121.            #Stampa del numero di valori  eliminati dalla serie temporale
122.            print('{}° gruppo: elementi scartati per rotazioni X (dopo svincolamento da temperatura): '.form
    at(k+1))
123.            print(len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)])-
    len(diz_timestamp_cross_corrX['gruppo{}'.format(k)]), 'su', len(dizionario_timestamp_cross_corr['gruppo{}'.f
    ormat(k)]), '(', round((len(dizionario_timestamp_cross_corr['gruppo{}'.format(k)])-
    len(diz_timestamp_cross_corrX['gruppo{}'.format(k)]))/len(dizionario_timestamp_cross_corr['gruppo{}'.format(
    k)]), 3)*100, '% )')
124.
125.            #Salvataggio dei coefficienti di cross-correlazione tra le serie
126.            exc1.save(percorso_file_risultati + '\\Cross_correlazione tra sensori del {}° gruppo (svincolati
     da temperatura).xlsx'.format(k+1))
127.
128.
129. print('\nTempo impiegato per il processo:', round(time.process_time()), 'sec') #Tempo di fine e durata dell'esecu
    zione dell'intero algoritmo
```

Figura 95: Blocco (24) del codice

4.9. Cenni su sensori di tipo A

Sensori di tipo A misurano anche rotazioni lungo un asse perpendicolare al primo (rotazioni Y), e l'umidità.

Questo vuol dire che tutte le operazioni eseguite per le rotazioni lungo l'asse X vengono ripetute anche per rotazioni di tipo Y. Inoltre, le istruzioni dal blocco (8.6) fino al (20) vengono ripetute due volte, come già accennato, per permettere lo svincolamento dei dati anche dall'umidità, oltre che dalla temperatura. Perciò le variabili che al primo ciclo erano riferite alla temperatura, al secondo vengono riferite all'umidità.

Vengono memorizzati un numero quadruplo di grafici, in quanto quelli salvati per rotazioni X vengono salvati anche per rotazioni Y, al termine di due diversi cicli anziché alla fine di uno solo.

Poi, si salvano due file Excel dei risultati, uno relativo alla procedura di svincolamento dalla temperatura e uno relativo allo svincolamento dall'umidità (su dati già svincolati da temperatura). In ciascuno di essi, ed in ciascun foglio, vengono generate due tabelle, una sottostante all'altra: la prima relativa a rotazioni X e la seconda relativa a rotazioni Y.

5. CONCLUSIONI

Si vogliono raccogliere di seguito le conclusioni del lavoro svolto.

È stato creato un codice scritto in Python che permetta l'analisi di dati di rotazione; tale algoritmo è stato usato per analizzare le rotazioni misurate da otto inclinometri posizionati su un viadotto autostradale, ma esso è scritto in modo tale da funzionare per un numero generico di sensori di due diverse tipologie, e in modo tale da poter impostare i parametri utilizzati per l'analisi su un file esterno, così da non dover modificare il codice.

Le serie di rotazioni sono state depurate dall'errore causato istantaneamente dalla temperatura sul sensore, ottenendo le serie di rotazioni corrette (capitolo 3.3), sulle quali sono state effettuate operazioni statistiche (capitolo 3.4); poi, queste serie sono state rese indipendenti dall'effetto che la temperatura provoca sull'impalcato, il quale avviene con un certo ritardo rispetto alla variazione di temperatura che l'ha provocato. Infine, la serie di rotazioni ottenuta è quella doppiamente corretta, ed è stata analizzata anch'essa (capitoli 3.6 e 3.7).

Si riportano in *Tabella 17* i parametri statistici più significativi delle tre serie, quella grezza, quella corretta e quella doppiamente corretta.

Tabella 17: Valor medio e scarto quadratico medio delle rotazioni grezze, compensate e doppiamente compensate

CODICE SENSORE	MEDIA RAW [mrad]	SQM RAW [mrad]	MEDIA K [mrad]	SQM K [mrad]	MEDIA KK [mrad]	SQM KK [mrad]
C23-T2-N	-0.072	0.093	-0.017	0.070	-0.026	0.069
C23-T2-S	-0.058	0.072	-0.024	0.062	-0.030	0.062
C23-T3-N	0.006	0.102	-0.037	0.090	-0.054	0.088
C23-T3-S	-0.072	0.086	-0.019	0.063	-0.026	0.062
C23-T4-N	-0.048	0.066	-0.019	0.057	-0.025	0.057
C23-T4-S	-0.065	0.086	-0.031	0.077	-0.031	0.077
C23-T5-N	-0.058	0.077	-0.01	0.062	-0.016	0.061
C23-T5-S	-0.205	0.169	-0.067	0.087	-0.067	0.087
MEDIE	-0,072	0,094	-0,028	0,071	-0,034	0,070

Come già anticipato nei capitoli relativi, si possono trarre le seguenti osservazioni:

1. Passando dai valori raw ai valori k:
    a. la media dei segnali scende da -0.072 mrad a -0.028 mrad, avvicinandosi quindi al valore di zero (ponte scarico) e alla sensibilità degli strumenti;
    b. lo scarto quadratico medio scende in proporzione molto meno dei valori medi da 0.094 mrad a 0.071 mrad;
    c. di conseguenza, i valori k sono più prossimi allo zero, ma più rumorosi dei valori raw.
2. Passando dai valori k ai valori kk:
    a. la media dei segnali si allontana dallo zero (ponte scarico), passando da –0.028 mrad a -0.034 mrad;
    b. lo scarto quadratico medio rimane praticamente costante;
    c. di conseguenza, si riduce leggermente la rumorosità del segnale;
    d. la differenza tra i valori k e kk è molto più bassa di quella tra i valori raw e i valori k, quindi la seconda pulizia potrebbe essere non necessaria.

Inoltre, per valutare la presenza di travi con comportamento anomalo, considerata la simmetria del viadotto e della disposizione dei sensori, è stata effettuata un'analisi del

grado di correlazione tra le serie di rotazioni ottenute per ciascun sensore. Si riportano le medie dei coefficienti di correlazione tra i sensori per ciascuna famiglia.

Tabella 18: Medie dei coefficienti di correlazione tra sensori per famiglia

	ROTAZIONI RAW	ROTAZIONI K	ROTAZIONI KK
ROTAZIONI ISTANTANEE	0.481	0.448	0.450
ROT. MEDIE 7 VALORI	0.600	0.695	0.708
ROT. MEDIE 9 VALORI	0.606	0.713	0.728
ROT. MEDIE 11 VALORI	0.610	0.725	0.741

Si possono trarre le seguenti osservazioni:

1. i valori degli indici di correlazione sono più alti sui valori di media mobile che sui valori istantanei in quanto si elimina l'effetto del traffico (che non è sincrono); difatti, la media totale aumenta nelle famiglie relative alle medie mobili rispetto a quelle relative ai dati istantanei;
2. al crescere del periodo su cui si effettua la media mobile (7, 9 o 11 valori) la correlazione cresce, come si evince dai valori medi dei gruppi;
3. i valori degli indici di correlazione tra rotazioni istantanee sono più alti per le rotazioni raw in quanto l'effetto istantaneo della temperatura è comune a tutti i sensori;
4. le serie k e kk hanno valori di correlazione simili ma la kk sembra presentare risultati migliori;
5. le serie k e kk presentano sulle medie mobili a 9 e 11 valori coefficienti di correlazione tra i sensori compresi tra 60% e 80%;
6. i sensori più dissimili dagli altri sono il T3-N e il T5-S con valori compresi tra 60% e 65%.

Si può quindi concludere che le travi sembrano avere un comportamento omogeneo e non si rilevano trend di anomalia, né nel tempo né nello spazio.

Osservando i grafici, è stato notato, anche nella serie doppiamente corretta, un trend verso il basso durante il periodo di acquisizione dei dati; in altri termini, le rotazioni diventano sempre più negative passando dalle misurazioni di febbraio a quelle di giugno.

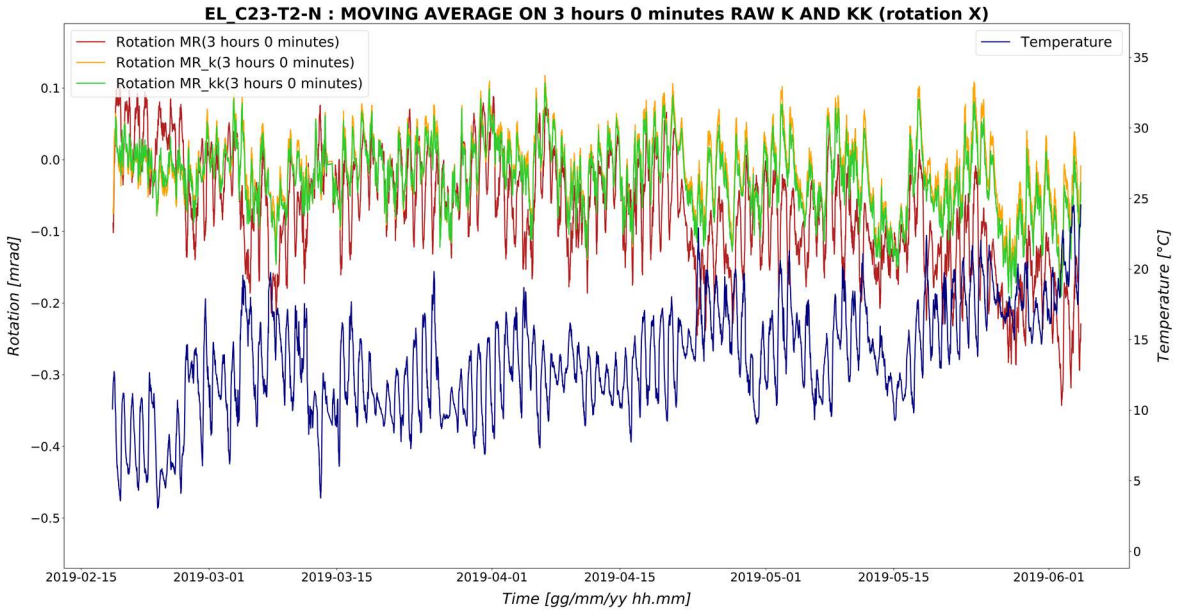


Figura 96: Media mobile su 9 valori della rotazione raw, k e kk

Questo probabilmente indica che le serie di rotazioni sono ancora affette dalla variazione di temperatura stagionale; non è escluso, comunque, che questo fenomeno sia dovuto a una deriva della strumentazione.
Si potrebbe capire quale delle due opzioni è quella esatta avendo i dati relativi a un intero anno, ovvero conoscendo l'andamento delle rotazioni sia passando dalla stagione invernale all'estiva, che viceversa. In questo modo, si potrebbe lavorare per eliminare l'effetto della variazione termica stagionale dalle misurazioni, nel caso sia questa la causa del trend osservato.

Una volta ottenute rotazioni indipendenti dalle variazioni di temperatura (giornaliera e stagionale), si potrebbe effettuare un confronto tra le misurazioni reali ottenute, e i risultati in termini di rotazioni ricavati da un modello teorico della struttura.

Indice delle Figure

Figura 1: Disposizione dei sensori 10
Figura 2: Inclinometri a barra 11
Figura 3: Elementi e dimensioni dell'inclinometro a barra 11
Figura 4: Funzionamento di un accelerometro capacitivo 12
Figura 5: Uso di un accelerometro per valutare inclinazioni 13
Figura 6: Disposizione dei mezzi per prova di carico con mezzi leggeri 16
Figura 7: Disposizione dei mezzi per prova di carico con mezzi pesanti 16
Figura 8: Disposizione dei capisaldi per la lettura degli spostamenti verticali 17
Figura 9: Variazioni di estensione dovute alla fase 1 – mezzi leggeri (με) 19
Figura 10: Variazioni di rotazione dovute alla fase 1 – mezzi leggeri (mrad) 19
Figura 11: Variazioni di estensione dovute alla fase 2 – mezzi pesanti (με) 20
Figura 12: Variazioni di rotazione dovute alla fase 2 – mezzi pesanti (mrad) 20
Figura 13: Variazioni di estensione dovute alle fasi 1 e 2 – mezzi leggeri e pesanti (με) 21
Figura 14: Variazioni di rotazioni dovute alle fasi 1 e 2 – mezzi leggeri e pesanti (mrad) 22
Figura 15: Grafico delle rotazioni 'old' e 'new' per il sensore EL_C23-T2-N 24
Figura 16: Media mobile su 9 valori centrata 28
Figura 17: Media mobile su 6 valori di temperatura precedenti al tempo t_i 30
Figura 18: Media mobile su 6 valori di temperatura precedenti al tempo t_i, con ritardo di un'ora 31
Figura 19: Coefficienti di correlazione tra rotazioni corrette mediate su 180 minuti e temperatura mediata con diversi ritardi (da 0h a -24h) 42
Figura 20: Generico segnale campionato 58
Figura 21: Segnale continuo e treno di impulsi 58
Figura 22: Trasformate di Fourier del segnale continuo e del corrispettivo campionato 59
Figura 23: Ampiezze delle trasformate di Fourier delle rotazioni grezze ('raw'), corrette 'k' e doppiamente corrette 'kk' 60
Figura 24: Ampiezze dell'Energy Spectral Density (ESD) delle rotazioni grezze 'raw', corrette 'k' e doppiamente corrette 'kk' 61
Figura 25: Rotazione raw e sua media mobile su 9 valori 62
Figura 26: Rotazione k e sua media mobile su 9 valori 63
Figura 27: Rotazione kk e sua media mobile su 9 valori 63
Figura 28: Media mobile su 9 valori della rotazione raw, k e kk 64
Figura 29: Differenza tra rotazione e sua media mobile su 9 valori, per rotazioni raw e k 65
Figura 30: Rotazione raw e media mobile su 9 valori delle rotazioni raw, k e kk (19-24/02/2019) 65
Figura 31: Rotazione raw e media mobile su 9 valori delle rotazioni raw, k e kk (21-26/03/2019) 66
Figura 32: Rotazione raw e media mobile su 9 valori delle rotazioni raw, k e kk (01-31/03/2019) 66
Figura 33: Rotazione raw e media mobile su 9 valori delle rotazioni raw, k e kk (05/04/2019) 67
Figura 34: Rotazione raw e media mobile su 9 valori delle rotazioni raw, k e kk (18/02/2019-04/06/2019) 67

Figura 35: Moduli importati ... 70
Figura 36: Moduli esterni creati e loro funzioni ... 73
Figura 37: Esempio di percorso del file EL_C23-T2-N.csv ... 74
Figura 38: Configurazione del file EL_C23-T2-N.csv ... 75
Figura 39: Configurazione della prima parte di "file di input utente.txt" ... 76
Figura 40: Configurazione della seconda parte di "file di input utente.txt" ... 77
Figura 41: Configurazione della terza parte di "file di input utente.txt" ... 78
Figura 42: Configurazione di dati_iniziali_B.csv ... 79
Figura 43: Foglio "risultati" del file "risultati scorrelati da temperatura.xlsx" ... 81
Figura 44: Foglio "coefficienti di correlazione raw" del file "risultati scorrelati da temperatura.xlsx" ... 81
Figura 45: Foglio "coeff. corr. tra sensori (raw)" del file "Cross_correlazione tra sensori del 1° gruppo" ... 82
Figura 46: Parte di "file di input utente.txt" usata nel blocco (1.3) ... 84
Figura 47: Blocchi (1.1), (1.2) e (1.3) del codice ... 86
Figura 48: Parte di "file di input utente.txt" usata nel blocco (1.3) e (1.4) ... 87
Figura 49: Linea temporale per cross-correlazione tra sensori ... 88
Figura 50: Blocchi (1.4), (1.5) e (1.6) del codice ... 90
Figura 51: Ciclo for sul numero dei sensori ... 91
Figura 52: Blocchi (8.1) e (8.2) del codice ... 94
Figura 53: Blocco (8.3) del codice ... 96
Figura 54: Parte di "file di input utente.txt" usata nel blocco (8.3) ... 96
Figura 55: Blocco (8.4) del codice ... 97
Figura 56: Modulo esterno Excel, funzione Excel.crea_layout ... 101
Figura 57: Modulo esterno Excel, funzione Excel.crea ... 102
Figura 58: Blocco (8.5) del codice ... 103
Figura 59: Modulo esterno Statistica, funzione Statistica.calcoli_rotazioni ... 106
Figura 60: Blocco (9) del codice ... 107
Figura 61: Blocco (10) del codice ... 108
Figura 62: Modulo esterno Statistica, funzione Statistica.calcoli_temp_umidB ... 112
Figura 63: Modulo esterno Statistica, funzione Statistica.correlazione ... 113
Figura 64: Blocco (11) del codice ... 114
Figura 65: Modulo esterno Statistica, funzione Statistica.deriva_istantea ... 116
Figura 66: Blocco (12) del codice ... 117
Figura 67: Blocco (13) del codice ... 118
Figura 68: Blocco (14) del codice ... 120
Figura 69: Modulo esterno Statistica, funzione Statistica.deriva_differita ... 122
Figura 70: Blocco (15) del codice ... 124
Figura 71: Blocco (16) del codice ... 125
Figura 72: Blocco (17) del codice ... 126
Figura 73: Blocco (18) del codice ... 128

Figura 74: Parte di "file di input utente.txt" usata nella funzione Grafici.plot_no_derive *129*
Figura 75: Modulo esterno Grafici, acquisizione parametri per la funzione Grafici.plot_no_derive *131*
Figura 76: Modulo esterno Grafici, funzione Grafici.plot_no_derive per la definizione e il salvataggio del primo grafico *133*
Figura 77: Blocco (19) del codice *134*
Figura 78: Prima parte di "file di input utente.txt" usata nella funzione Grafici.fourier *134*
Figura 79: Seconda parte di "file di input utente.txt" usata nella funzione Grafici.fourier *135*
Figura 80: Modulo esterno Grafici, prima parte della funzione Grafici.fourier *136*
Figura 81: Modulo esterno Grafici, seconda parte della funzione Grafici.fourier *138*
Figura 82: Blocco (20) del codice *139*
Figura 83: Esempio di DataFrame *139*
Figura 84: Esempio di dizionario i cui valori sono DataFrame *141*
Figura 85: Blocco (21) del codice *145*
Figura 86: Esempio di utilizzo della funzione Cross_correlazione.salvataggio_dati, per k=0 ..... *146*
Figura 87: Esempio di utilizzo della funzione Cross_correlazione.salvataggio_dati, per k=2 ..... *146*
Figura 88: Esempio di eliminazione di una riga nel DataFrame già formato *148*
Figura 89: Modulo esterno Cross_correlazione, funzione Cross_correlazione.salvataggio_dati *150*
Figura 90: Blocco (23) del codice *152*
Figura 91: Modulo esterno Grafici, funzione Grafici.ritardi_plot *154*
Figura 92: Modulo esterno Cross_correlazione, funzione Cross_correlazione.crea_excel *160*
Figura 93: Modulo esterno Cross_correlazione, prima parte della funzione Cross_correlazione.cross_corr *162*
Figura 94: Modulo esterno Cross_correlazione, seconda parte della funzione Cross_correlazione.cross_corr *166*
Figura 95: Blocco (24) del codice *170*
Figura 96: Media mobile su 9 valori della rotazione raw, k e kk *174*

Indice delle Tabelle

Tabella 1: Valor medio e scarto quadratico medio delle rotazioni grezze 28
Tabella 2: Coefficienti di correlazione tra rotazioni 'raw' (istantanee e mediate) e temperatura istantanea 33
Tabella 3: Coefficienti di correlazione tra rotazioni 'raw' (istantanee e mediate) e temperatura (istantanea e mediata) 34
Tabella 4: Deriva termica istantanea 36
Tabella 5: Valor medio e scarto quadratico medio delle rotazioni grezze e compensate 37
Tabella 6: Coefficienti di correlazione tra rotazioni corrette (istantanee e mediate) e temperatura (istantanea, mediata e mediata con ritardo) 39
Tabella 7: coefficienti di correlazione tra rotazioni corrette (istantanee e mediate) e temperatura (istantanea, mediata e mediata con ritardo) 40
Tabella 8: Coefficienti di correlazione tra rotazioni corrette (istantanee e mediate) e temperatura (istantanea, mediata e mediata con ritardo) 41
Tabella 9: Deriva termica differita 44
Tabella 10: Valor medio e scarto quadratico medio delle rotazioni grezze, compensate e doppiamente compensate 45
Tabella 11: Coefficienti di correlazione tra rotazioni doppiamente corrette (istantanee e mediate) e temperatura (istantanea, mediata e mediata con ritardo) 47
Tabella 12: Coefficienti di correlazione tra rotazioni doppiamente corrette (istantanee e mediate) e temperatura (istantanea, mediata e mediata con ritardo) 48
Tabella 13: Coefficienti di correlazione tra rotazioni doppiamente corrette (istantanee e mediate) e temperatura (istantanea, mediata e mediata con ritardo) 49
Tabella 14: Coefficienti di correlazione tra rotazioni raw 51
Tabella 15: Coefficienti di correlazione tra rotazioni k 53
Tabella 16: Coefficienti di correlazione tra rotazioni kk 55
Tabella 17: Valor medio e scarto quadratico medio delle rotazioni grezze, compensate e doppiamente compensate 172
Tabella 18: Medie dei coefficienti di correlazione tra sensori per famiglia 173

Ringraziamenti

Giunti alla conclusione, mi sento in dovere di ringraziare coloro che mi hanno aiutato nella stesura della tesi, e più in generale quelli che mi hanno accompagnato nel mio percorso universitario.

Vorrei ringraziare il professor Gabriele Bertagnoli, per la sua disponibilità e la leggerezza con cui mi ha permesso di lavorare.
Un grazie immenso lo devo ai miei genitori, che mi hanno costantemente supportato; non mi sono mai sentita abbandonata da loro.
Non posso scordarmi di ringraziare il resto della mia famiglia; riconosco quanto è importante avervi accanto.
Grazie Andrea, per essermi vicino nei momenti felici, ed in quelli difficili.
Per essere stati compagni di viaggio fantastici, per tutti gli istanti che abbiamo condiviso, non posso che ringraziare gli amici di Perugia e quelli di Torino.
Infine, permettetemi di ringraziare me stessa, per averci creduto sempre.