

UNIVERSITÀ DEGLI STUDI DI MILANO  
Facoltà di Scienze e Tecnologie  
*Corso di Laurea in Informatica*

# TEMP IDENTIFICAZIONE DI SATELLITI

**Relatore:** Prof. Giovanni RIGHINI

**Correlatore:**

Tesi di:  
Jonathan Junior AGYEKUM  
Matricola: 935132

Anno Accademico 2022-2023

*dedicato a ...*

# Prefazione

prefazione

## Organizzazione della tesi

La tesi è organizzata come segue:

- nel Capitolo 1 ....

# Ringraziamenti

Grazie.

# Indice

	ii
<b>Prefazione</b>	<b>iii</b>
<b>Ringraziamenti</b>	<b>iv</b>
<b>1 Introduzione</b>	<b>1</b>
1.1 Strumenti utilizzati . . . . .	1
1.2 Ricerca operativa e astronomia . . . . .	1
1.3 Storia dell'identificazione di satelliti . . . . .	1
1.3.1 Rilevamenti diretti e indiretti . . . . .	1
<b>2 Descrizione del problema</b>	<b>2</b>
2.1 Descrizione . . . . .	2
2.2 Campi di applicazione e utilità . . . . .	3
<b>3 Sviluppo del modello</b>	<b>4</b>
3.1 Definizione del modello . . . . .	4
3.1.1 Calcolo della regione pareto-ottima . . . . .	5
3.2 Algoritmo di controllo . . . . .	9
3.2.1 Definizione del passo per il minimo periodo $\beta$ . . . . .	9
3.2.2 Punto di inizializzazione . . . . .	10
3.2.3 Criterio arresto . . . . .	11
3.2.4 Scelta del solutore . . . . .	12
3.2.5 Gestione degli errori . . . . .	14
3.2.6 Selezione della soluzione . . . . .	15
<b>4 Sviluppo dell'algoritmo</b>	<b>47</b>
4.1 Struttura e approccio . . . . .	47
4.1.1 Creazione dei sottoproblemi . . . . .	48
4.1.2 Gestione dei nodi non foglia . . . . .	48

4.1.3	Gestione dei nodi foglia . . . . .	49
-------	------------------------------------	----

# Capitolo 1

## Introduzione

1.1 Strumenti utilizzati

1.2 Ricerca operativa e astronomia

1.3 Storia dell'identificazione di satelliti

1.3.1 Rilevamenti diretti e indiretti

## Capitolo 2

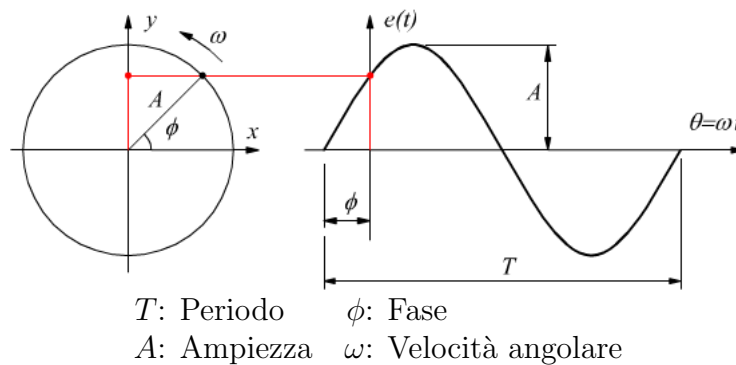
# Descrizione del problema

### 2.1 Descrizione

L'osservazione in modo continuo dell'angolo descritto da  $k$  satelliti intorno al loro pianeta, genera delle sinusoidi, questo apre alla possibilità di una identificazione dei satelliti attraverso l'analisi del loro moto orbitale.

Il legame tra l'orbite dei satelliti e le sinusoidi può essere spiegata considerando le orbite dei satelliti circolari e quindi attraverso le leggi del moto circolare.

Figura 1: Moto circolare uniforme



Dove il periodo corrisponde al tempo di rivoluzione del satellite, l'ampiezza alla massima distanza che intercorre nel periodo tra il satellite e il corrispettivo pianeta,  $e(t)$  lo spostamento del satellite lungo l'orbita. È quindi possibile descrivere la sinusoide risultante tramite l'espressione analitica:  $e(t) = A \sin(\omega t + \phi)$

L'obiettivo è quello di riuscire ad attribuire ai  $k$  satelliti i  $k$  valori di ogni osservazione e di trovare periodo, ampiezza e fase delle  $k$  sinusoidi, supponendo di poter



osservare i  $k$  valori simultaneamente, in  $n$  momenti successivi. Si suppone di non conoscere nulla dei  $k$  satelliti e soprattutto di non poterli distinguere in fase di rilevazione dei valori.

Il problema è scomponibile in due sottoproblemi: uno di interpolazione e uno di assegnamento. La componente di interpolazione è identificata nell'operazione di individuazione della sinusoide che si scosta il meno possibile dai valori. Questo definisce un problema di interpolazione non-lineare e continuo: non-lineare poiché una sinusoide è una funzione periodica non-lineare, definita da una trasformazione elementare della funzione seno, continuo poiché il dominio di una sinusoide e delle sue componenti è definito in  $\mathbb{R}$  o in un suo sottoinsieme.

La componente di assegnamento è individuata nel processo di selezione dei punti tra le diverse osservazioni da interpolare. La funzione di questa componente è quella di individuare  $k$  sinusoidi, generate dall'interpolazione di una assegnazione di valori, che rappresentino i  $k$  satelliti.

## 2.2 Campi di applicazione e utilità

Il principio della tipologia di identificazione che ho studiato è quello di riconoscere satelliti di un pianeta in base alla variazione in un determinato dominio di una caratteristica quantificabile, in questo caso l'angolo, in funzione del tempo. La variazione dev'essere riconoscibile e identificativa del satellite, cioè deve definire un andamento caratteristico di una funzione elementare o una combinazione di essi. Questo principio è illustrato, per esempio dal metodo di transito, metodo di identificazione fotometrico di satelliti extrasolari [4].

La tipologia di identificazione che illustrerò, può essere applicata sia ai casi di rilevamento diretto, tecniche che permettono di osservare direttamente al telescopio i satelliti, e sia indiretti, cioè l'individuazione tramite effetti fisici che satellite può indurre. Il vincolo fondamentale è che sia possibile effettuare misurazioni dell'angolo rispetto al suo pianeta.

L'identificazione attraverso l'analisi del moto orbitale può essere utile in tutte le situazioni dove non è possibile riconoscere o distinguere due o più satelliti o qualunque entità il quale movimento è osservabile e rotante.

# Capitolo 3

## Sviluppo del modello

### 3.1 Definizione del modello

Propongo un modello per la risoluzione del sottoproblema di interpolazione. La sinusoidale che si vuole ottenere dall'interpolazione dei punti deve avere determinate caratteristiche. Tra tutte le sinusoidi possibili che interpolano gli  $n$  valori, che in questo contesto chiamerò punti, si vuole quella di massimo periodo e sicché le misurazioni possono essere affette da errore si deve combinare la ricerca del massimo periodo con quella di minimo errore, definendo un problema a due obiettivi.

Il processo risolutivo per un problema a molti obiettivi in conflitto tra loro, cioè che il miglioramento di una comporti un peggioramento di un'altra, prevede:

1. Il calcolo della regione pareto-ottima, cioè l'insieme di soluzioni ammissibili non dominate, denominate anche soluzioni paretiane.
2. La scelta di una soluzione tra quelle individuate.

La determinazione della regione pareto-ottima può essere effettuata in diversi modi. I modelli che ho definito inseguito utilizzano rispettivamente metodo dei pesi e metodo dei vincoli, i due modelli si differenziano solo per vincoli e funzione obiettivo.

In questo processo di interpolazione è necessario fare attenzione alle sinusoidi con basso periodo rispetto alla sinusoidale da individuare: si può osservare dalle regioni paretiane rappresentate dalle figure da 2 a 4 del paragrafo 3.2.6 che aumentando la frequenza, quindi al diminuire del periodo, esiste sempre un modo di interpolare  $n$  punti con una sinusoidale minimizzando a piacimento l'errore di interpolazione.

**Parametri** I dati a disposizione del modello sono gli  $n$  punti assegnati da interpolare, il quale numero è determinato dal modulo di assegnamento. I punti sono individuati in un piano cartesiano dove l'asse delle ascisse rappresenta il tempo e

l'asse delle ordinate l'angolo. I parametri del modello sono quindi le  $n$  ascisse  $t_i$  e le  $n$  ordinate  $e_i$  degli  $n$  punti, con  $i$  da 1 a  $n$ .

Per una maggiore semplicità numerica considero i secondi per il tempo mentre rappresento gli angoli in radianti.

**Variabili** Le variabili sono individuate nelle componenti che determinano una sinusoide, data l'espressione analitica generale di una funzione sinusoidale:

$$e_i = A \sin(\omega t_i + \phi) \quad (1)$$

Dall'espressione ho definito le variabili che rappresentano fase  $\phi$ , ampiezza  $A$  e la pulsazione  $\omega$ .

Inoltre c'è la necessità di rappresentare l'errore che può essere generato dalle misurazioni o da un assegnamento di punti errato. L'espressione analitica diventa:

$$e_i = A \sin(\omega t_i + \phi) + \varepsilon_i \quad (2)$$

Perciò ho definito  $n$  variabili aggiuntive che rappresentano l'errore per ogni punto.

### 3.1.1 Calcolo della regione pareto-ottima

#### Metodo dei pesi

Il metodo dei pesi consiste nel dare un peso a ciascuna delle funzioni obiettivo, così da ridurre il problema a molti obiettivi in uno di programmazione matematica parametrica.

**Vincoli** Ho definito come unico vincolo l'equazione (2)

**Funzione obiettivo** Le due funzioni obiettivo da ottimizzare sono:

$$\min f_1 = \omega \quad (3)$$

La massimizzazione del periodo

$$\min f_2 = \frac{\sum_{i=1}^n \varepsilon_i^2}{n} \quad (4)$$

La minimizzazione dell'errore complessivo, calcolo l'errore quadratico complessivo per l'eliminazione dei possibili valori negativi che si possono generare.

I pesi definiti per le funzioni  $f_1$  e  $f_2$  sono rispettivamente 1 e -1, quindi ottenendo la funzione obiettivo:

$$\max f = f_1 + f_2 \quad (5)$$

Questa configurazione dei pesi di partenza permette di penalizzare tutte quelle soluzioni ammissibili che massimizzano il periodo ma hanno un errore elevato.

Il processo di individuazione delle soluzioni paretiane, con questo metodo e numero di funzioni, consiste nel eseguire un'analisi parametrica sui pesi, cioè analizzare e individuare soluzioni non dominate al variare dei pesi entro un certo range. In questo caso può essere sufficiente solo effettuare la variazione del peso dell'errore.

Un modello che da priorità al periodo ottenendo un alto errore è per certo non utilizzabile, perché l'alto errore denota una completa mancanza di relazione tra i punti e la sinusoide individuata, questo spiega l'analisi parametrica solo sul peso dell'errore. È anche possibile non effettuare alcuna variazione dei pesi, ed accontentarsi della prima soluzione individuata tramite la configurazione dei pesi di partenza.

### Metodo dei vincoli

Il metodo dei vincoli consiste nell'ottimizzare una delle funzioni obiettivo, trasformando le rimanenti in vincoli, utilizzando un termine noto parametrico.

**Vincoli** Si ha l'equazione (2) e la trasformazione in vincolo della funzione obiettivo legato al periodo (3)

$$\frac{2\pi}{\omega} \geq \beta \quad (6)$$

Il termine noto  $\beta$  sta ad indicare il minimo periodo ammissibile per una soluzione, questo permette di escludere le soluzioni a basso periodo rendendo inammissibili soluzioni con periodo più piccolo del termine noto.

Oppure se si applica il metodo dei vincoli sulla funzione obiettivo sull'errore (4) si ottiene:

$$\frac{\sum_{i=1}^n \varepsilon_i^2}{n} \leq \alpha \quad (7)$$

Il termine noto  $\alpha$  rappresenta il massimo errore ammissibile per una soluzione.

**Funzione obiettivo** La funzione obiettivo definita è la funzione (4).

L'individuazione delle soluzioni paretiane è effettuata variando il termine noto parametrico nel vincolo (6). Ho scelto di definire il vincolo (6) perché permette di avere un controllo migliore sull'esplorazione dei periodi possibili della sinusoide da individuare, data dalla possibilità di definire come soluzione di partenza del modello, un

assegnazione della variabile che rappresenta la pulsazione a partire dal termine noto  $\beta$ .

$$\omega = \frac{2\pi}{\beta} \quad (8)$$

Questa scelta risulta preferibile anche a seguito di sperimentazioni. Eseguendo i solutori con un modello definito con il metodo dei vincoli sull'errore e un modello con il metodo dei vincoli definito sul periodo, dai seguenti problemi e punti di inizializzazione si ottiene:

- Problema di 10 punti generati dalla sinusoide  $e_i = \sin(\omega t_i)$  con  $\omega$  pari a  $0.0013 \frac{rad}{s}$  che corrisponde ad un periodo di 4830s, con  $t_i$  che assume i valori nell'insieme intero  $[0, 9]$

Tabella 1: Prestazioni dei solutori con modello con vincolo sull'errore

Solutore	soluzione $p_{init} = 2500s$		soluzione $p_{init} = 3694s$		soluzione $p_{init} = 4186s$	
	periodo (s)	errore (n)	periodo(s)	errore (n)	periodo (s)	errore (n)
COBYLA	6e6	1.0e-3	6e6	3.0e-4	6e6	3.7e-4
BOBYQA	14e9	4.6e-5	14e9	4.6e-5	14e9	4.6e-5
NEWUOA	6e6	4.8e-5	6e6	4.8e-5	6e6	4.8e-5
PRAXIS	2e5	4.8e-5	2e4	4.1e-5	5e5	5.1e-5
SBPLX	3e5	3.7e-5	3e5	3.7e-5	3e5	3.7e-5

$p_{init}$  rappresenta il periodo di inizializzazione da cui si ricava la pulsazione da fissare alla variabile  $\omega$ , per ottenere una soluzione iniziale.

Tabella 2: Prestazioni dei solutori con modello con vincolo sul periodo

Solutore	soluzione $p_{init} = 2500s$		soluzione $p_{init} = 3694s$		soluzione $p_{init} = 4186s$	
	periodo (s)	errore (n)	periodo(s)	errore (n)	periodo (s)	errore (n)
COBYLA	4244	2.4e-5	4477	1.4e-5	4185	1.6e-5
BOBYQA	7324e5	3.4e-5	9645e3	3.1e-5	2708e6	3.4e-5
NEWUOA	3980e-7	0.0	2707e-7	0.0	2388e-7	0.0
PRAXIS	3982e-6	3e-1	2710e-6	6.3	2387e-6	6.2
SBPLX	inf		inf		4188	1.3e-15

$p_{init}$  rappresenta il periodo di inizializzazione da cui si ricava la pulsazione da fissare alla variabile  $\omega$ , per ottenere una soluzione iniziale.

- Problema di 10 punti generati dalla sinusoide  $e_i = \sin(\omega t_i)$  con  $\omega$  pari a  $0.8 \frac{rad}{s}$  che corrisponde ad un periodo di 7.8s con  $t_i$  intero da 0 a 9 che assume i valori nell'insieme intero  $[0, 9]$

Tabella 3: Prestazioni dei solutori con modello con vincolo sull'errore

Solutore	soluzione $p_{init} = 2500s$		soluzione $p_{init} = 1250s$		soluzione $p_{init} = 500s$	
	periodo (s)	errore (n)	periodo(s)	errore (n)	periodo (s)	errore (n)
COBYLA	514.7	0.2	230.0	0.2	184.5	0.2
BOBYQA	787.6	0.4	870.6	0.3	870.6	0.3
NEWUOA	999.9	0.0	999.9	0.0	999.9	0.0
PRAXIS	999.9	0.0	999.9	0.0	999.9	0.0
SBPLX	inf		inf		inf	

$p_{init}$  rappresenta il periodo di inizializzazione da cui si ricava la pulsazione da fissare alla variabile  $\omega$ , per ottenere una soluzione iniziale.

Tabella 4: Prestazioni dei solutori con modello con vincolo sul periodo

Solutore	soluzione $p_{init} = 2500s$		soluzione $p_{init} = 1250s$		soluzione $p_{init} = 500s$	
	periodo (s)	errore (n)	periodo(s)	errore (n)	periodo (s)	errore (n)
COBYLA	0.8	0.2	1.1	0.3	0.1	0.1
BOBYQA	0.003	0.09	0.007	0.1	0.02	0.1
NEWUOA	0.0004	4.8e−23	0.0008	3e−21	0.002	7.4e−19
PRAXIS	0.0004	0.2	0.0008	0.02	0.002	15 894
SBPLX	0.0008	1.7e−12	0.001	1e−12	0.004	3.7e−11

$p_{init}$  rappresenta il periodo di inizializzazione da cui si ricava la pulsazione da fissare alla variabile  $\omega$ , per ottenere una soluzione iniziale, inoltre si fissano periodi negativi a causa del vincolo 6 che vincola la ricerca a periodi maggiori del termine noto  $\beta$

Dai risultati delle sperimentazioni si può notare che nel primo problema, il solutore COBYLA utilizzato con il vincolo sul periodo individua soluzioni che si scostano mediamente di 529s, a differenza se utilizzato con il vincolo sull'errore con gli stessi punti di inizializzazione si ottengono soluzioni dal periodo molto più distanti. Le stesse osservazioni possono essere effettuate nel secondo problema. Invece il restante dei solutori non ottiene soluzione accettabile con nessuno dei due modelli.

Per le sperimentazioni effettuate ho fissato il termine noto  $\alpha$  nel vincolo dell'errore (7) a  $1e-3$ , anche provando a diminuire il valore del termine noto, quindi vincolare il

solutore ad individuare soluzioni dall'errore quadratico medio entro il valore fissato, i risultati risultano peggiori rispetto all'utilizzo di un modello con metodo dei vincoli applicato alla funzione obiettivo del periodo, da come si può osservare dalle tabelle ...

Dati i risultati ed il miglior controllo sull'inizializzazione e sul vincolo 6, ho scelto di utilizzare il modello con il metodo dei vincoli applicato al periodo.

## 3.2 Algoritmo di controllo

Il modello che ho scelto di implementare è il modello che utilizza il metodo dei vincoli per i vantaggi citati nella sua specificazione. Una volta definito l'implementazione del modello è necessario specificare un algoritmo di controllo e di esecuzione di esso. I compiti sono la definizione del valore del minimo periodo  $\beta$ , nel vincolo (6) ad ogni iterazione e la gestione di eventuali errori.

---

### Algoritmo 1: Algoritmo di controllo del modello

---

**input** : *pt* Collezione di punti da interpolare  $[(t_1, e_1), \dots, (t_k, e_k)]$   
**output**: soluzioni Insieme di soluzioni che interpolano i punti *pt*

```

1 passo  $\leftarrow$  1
2  $\beta \leftarrow$  periodoIniziale
3 soluzioni  $\leftarrow$   $\emptyset$ 
4 while  $\beta \leq$  periodoLimite do
5   solPrecedente  $\leftarrow$  sol
6   sol  $\leftarrow$  Interpola(pt,  $\beta$ , solutore)
7   Aggiungi sol a soluzioni
8   if sol[periodo] = solPrecedente[periodo] then
9     | passo = f(passo)
10  else
11    | passo  $\leftarrow$  1
12  end
13   $\beta \leftarrow \beta +$  passo
14 end

```

---

### 3.2.1 Definizione del passo per il minimo periodo $\beta$

Il passo definisce la quantità aggiunta al minimo periodo ad ogni iterazione. Per facilitare l'individuazione di soluzioni differenti, è possibile definire un passo variabile,

così da spostare il punto di inizializzazione del problema dalla regione di un minimo locale, in base alla soluzione ottenuta.

L'algoritmo va ad aumentare il passo ogniqualvolta il periodo della soluzione individuata in una iterazione, appartiene ad un intorno del periodo della soluzione dell'iterazione precedente (riga 8), altrimenti si ha un ripristino del passo ad un valore di default (riga 11). In base alle necessità è possibile definire velocità differenti di variazione del passo, tramite la definizione della funzione  $f$  (riga 9). Per aumentare il numero di soluzioni individuate è necessario diminuire la velocità di variazione di passo in modo da ottenere inizialmente piccoli spostamenti del minimo periodo, al contrario per ottenere meno soluzioni ed una esplorazione più veloce, è necessario aumentare la velocità di variazione del passo. esempi possono essere:

$$f(x) = x * k \quad (9)$$

Si moltiplica il passo  $p$  per una costante  $k$ , è possibile variare la velocità della variazione del passo aumentando o diminuendo il parametro  $k$ . Si misura la velocità della funzione di variazione del passo tramite la derivata della funzione  $f(x)$ .

Un altro fattore che determina l'esplorazione è la dimensione dell'intorno. È necessario definire un intorno poiché è possibile che con approssimazioni e specifiche interne del solutore o anche in base alle caratteristiche della regione ammissibile, non si individui l'esatto valore di un minimo locale, ma un valore nell'intorno di esso in iterazioni successive. Perciò è necessario definire un livello di tolleranza con il quale definire se due soluzioni sono da considerare uguali (riga 8).

### 3.2.2 Punto di inizializzazione

Il processo di inizializzazione di un modello, consiste nel fissare le variabili del problema a dei valori di partenza. Generalmente definire una soluzione iniziale per il solutore nei problemi di programmazione non lineare è importante, poiché ne determina l'abilità di individuare un minimo locale diverso da un altro, e poter individuare un minimo locale migliore sotto una determinata caratteristica.

In questo problema la qualità della soluzione è individuata prevalentemente nel periodo e nell'errore, perciò la variabile che assume più importanza nella determinazione di una soluzione iniziale è la pulsazione  $\omega$ , gli errori per ogni punto sono variabili risultanti perciò non vanno fissate.

Con questa definizione del modello, mi è risultato difficile discernere un algoritmo di definizione della soluzione di partenza, a partire dai test effettuati. La difficoltà nasce dal interpretare il comportamento del solutore dato un problema e una serie di soluzioni di partenza all'aumentare del minimo periodo  $\beta$ , quindi dall'individuazione di un pattern chiaro e veloce che definisca la posizione ideale della soluzione di partenza, nelle successive iterazioni.



I test sono stati effettuati eseguendo l'algoritmo di controllo, 1 dando come punto di iniziale il seguente:

- $A = 0$
- $\phi = 0$
- $w = \frac{4\pi}{5000-\beta}$

Con i seguenti parametri dell'algoritmo 1:

- $f(x)$  in riga 9, algoritmo 1 pari alla funzione (9) con  $k = 5$
- $periodoIniziale = 0s$
- $periodoFinale = 5000s$

Fissando  $\omega$  con il periodo a metà tra il periodo limite e il minimo periodo  $\beta$ , ad esempio nel problema con 30 punti generati dalla sinusoide  $e(t) = \sin(\omega t)$

Tabella 5: Iterazioni effettuate per il problema con Sinusoide con  $\omega = 0.8 \text{ rad/s}$

Periodo limite $\beta$ (s)	Periodo di partenza (s)	Periodo soluzione (s)	Errore quadratico medio
1	1201	7.5	1.3e-11
2	2095	10.3	7.5e-5
7	4175	10.9	3.2e-4
32	4175	10.9	3.2e-4
157	4175	10.9	3.2e-4
782	4175	10.9	3.2e-4
3907	4175	10.9	3.2e-4
4532	4175	10.9	3.2e-4
4657	4175	10.9	3.2e-4

### 3.2.3 Criterio arresto

È necessario specificare un criterio di arresto per l'algoritmo di controllo. L'idea applicata è quella di determinare l'arresto fissando un periodo limite per il minimo periodo  $\beta$  (riga 4). Effettuando una ricerca sui satelliti del sistema solare, ho individuato che il massimo periodo di un satellite nel sistema solare è pari a 9374.0 giorni [3].

Quindi è possibile fissare un periodo limite, effettuando un mappaggio da giorni in secondi, pari a 5000s. Non è necessario fissare esattamente il valore estratto dallo studio dei periodi del sistema solare, poiché il periodo  $\beta$ , utilizzato nel vincolo (6) come limite inferiore, non impedisce al solutore di individuare periodi dai valori molto più grandi rispetto ai periodi del sistema solare, tutto questo facendo partire l'individuazione delle soluzioni da un periodo pari a 0, escludendo i periodi negativi. L'esclusione dei periodi negativi non influisce sull'individuazione delle soluzioni poiché i punti sono rilevati all'avanzare del tempo.

### 3.2.4 Scelta del solutore

A causa dei vincoli di utilizzo dei solutori globali in nlopt [1], la scelta del solutore è ricaduta ai solutori locali (riga 6). È possibile condurre la selezione dell'algoritmo solutore valutandoli su problemi di interpolazione definiti da punti appartenenti ad una stessa sinusoide  $e(t) = \sin(\omega t)$  per un numero di punti definito. Le ascisse dei  $n$  punti assumono i valori interi dell'insieme  $[0, n]$ . Analizzo il numero totale di soluzioni individuate, il numero di soluzioni individuate in un intorno del periodo della sinusoide del problema, infine il tempo di calcolo. L'intorno scelto è di dimensioni pari a 100s, è possibile scegliere un diverso intorno in base alle proprie necessità di precisione rispetto alla sinusoide da individuare.

I risultati delle sperimentazioni sono illustrati nelle tabelle da 6 a 10. I parametri dell'algoritmo 1 per le seguenti sperimentazioni sono:

- $f(x)$  in riga 9, algoritmo 1 pari alla funzione (9) con  $k = 2$
- $periodoIniziale = 0s$
- $periodoFinale = 5000s$

Tabella 6: Prestazioni dei solutori: Sinusoide con  $\omega = 0.8 \text{ rad/s}$

Solutore	n. soluzioni individuate	n. soluzioni nell'intorno	Tempo di calcolo (s)
COBYLA	120	21	6.1
BOBYQA	19	6	0.9
NEWUOA	113	101	5.9
PRAXIS	139	139	6.9
SBPLX	26	9	1.3

Tabella 8: Prestazioni dei solutori: Sinusoide con  $\omega = 0.005 \text{ rad/s}$ 

Solutore	n. soluzioni individuate	n. soluzioni nell'intorno	Tempo di calcolo (s)
COBYLA	218	3	11.0
BOBYQA	20	0	1.0
NEWUOA	113	0	5.6
PRAXIS	149	0	7.4
SBPLX	25	0	1.2

Tabella 7: Prestazioni dei solutori: Sinusoide con  $\omega = 0.0125 \text{ rad/s}$ 

Solutore	n. soluzioni individuate	n. soluzioni nell'intorno	Tempo di calcolo (s)
COBYLA	141	19	7.1
BOBYQA	20	0	1.0
NEWUOA	114	0	5.7
PRAXIS	139	0	6.9
SBPLX	26	1	1.3

Tabella 9: Prestazioni dei solutori: Sinusoide con  $\omega = 0.0013 \text{ rad/s}$ 

Solutore	n. soluzioni individuate	n. soluzioni nell'intorno	Tempo di calcolo (s)
COBYLA	193	4	9.8
BOBYQA	19	0	0.9
NEWUOA	113	0	5.6
PRAXIS	146	0	7.3
SBPLX	24	0	1.2

Tabella 10: Prestazioni dei solutori: Sinusoide con  $\omega = 0.0010 \text{ rad/s}$ 

Solutore	n. soluzioni individuate	n. soluzioni nell'intorno	Tempo di calcolo (s)
COBYLA	54	0	2.7
BOBYQA	19	0	0.9
NEWUOA	113	0	5.6
PRAXIS	144	0	7.2
SBPLX	27	0	1.3

È interessante analizzare il comportamento dei solutori se il modello utilizzasse trasformasse in vincolo la funzione dell'errore (funzione 4) invece del periodo: l'algoritmo scelto dall'insieme disponibile è COBYLA [2], da come si può notare, è l'unico solutore che individua soluzioni nell'intorno specificato. È da evidenziare la difficoltà per il solutore COBYLA nel individuare soluzioni al di fuori del range di esplorazione fissato per il criterio d'arresto, come da esempio in tabella 10 dove i punti del problema sono generati da una sinusoide con periodo pari a  $T = \frac{2\pi}{\omega} = \frac{2\pi}{0.0010} = 6280s$ .

### 3.2.5 Gestione degli errori

L'unico errore che può verificarsi da tenere in considerazione è `nlopt.RoundoffLimited`. È un evento che specifica una situazione di errore inerente progressivo, cioè l'errore che si commette rappresentando un numero reale con un numero finito di cifre, intrinseco nei calcolatori. Si può verificare quando si esegue l'implementazione del modello per l'interpolazione dei punti in riga 6 dell'algoritmo 1.

Quando si verifica un'eccezione di errore inerente, è possibile rieseguire il solutore aumentando i valori di tolleranza sulla precisione della variazione delle variabili e della funzione obiettivo (riga 18, algoritmo 2).

La tolleranza determina il criterio di arresto nel processo di individuazione di un ottimo locale [1], quando la variazione del valore della funzione obiettivo o delle variabili in ogni direzione è minore della tolleranza impostata, il solutore restituirà l'ultima soluzione individuata, perciò aumentando i valori di tolleranza si impedisce la generazione di un errore inerente progressivo, non considerando a priori, valori dalla precisione elevata.

---

**Algoritmo 2:** Algoritmo di controllo del modello con gestione degli errori

---

**input** :  $pt$  Collezione di punti da interpolare  $[(t_1, e_1), \dots, (t_k, e_k)]$   
**output**: soluzioni Insieme di soluzioni che interpolano i punti  $pt$

```

1 passo  $\leftarrow 1$ 
2  $\beta \leftarrow valoreIniziale$ 
3  $tolFun \leftarrow tolleranzaDefault$ 
4  $tolVar \leftarrow tolleranzaDefault$ 
5 soluzioni  $\leftarrow \emptyset$ 
6 while  $\beta \leq periodoLimite$  do
7   try:
8     solPrecedente  $\leftarrow sol$ 
9     sol  $\leftarrow Interpola(pt, \beta, solutore, tolFun, tolVar)$ 
10    Aggiungi sol a soluzioni
11    if sol[periodo] = solPrecedente[periodo] then
12      | passo = f(passo)
13    else
14      | passo  $\leftarrow 1$ 
15    end
16     $\beta \leftarrow \beta + passo$ 
17  catch RoundoffLimited:
18    | Aumenta tolFun e tolVar
19  end
20 end

```

---

### 3.2.6 Selezione della soluzione

La selezione di una soluzione tra le tante individuate determina un passo fondamentale per tutto il modulo del sottoproblema di interpolazione, poiché si potrebbe incorrere nel rischio di scartare la sinusoide che rappresenta esattamente uno dei  $k$  satelliti. I due metodi identificati per questo compito si basano su diverse ipotesi sulla sinusoide ideale e sulle caratteristiche di tutte le sinusoidi individuate, questi sono il criterio del punto di utopia, e criterio degli standard solo sull'errore.

#### Criterio del punto di utopia

Il punto di utopia è la soluzione che nello spazio degli obiettivi ha come coordinate i valori ottimi di ciascuno. Il problema principale che sorge nello applicare questo

criterio al problema di interpolazione, è che non si conosce il valore ottimo ideale dell'obiettivo rispetto al periodo, mentre per l'errore, il valore è pari a 0.

Questo succede poiché non esiste un limite superiore al periodo che specifichi l'insieme di sinusoidi che violino uno dei vincoli del modello, essi avranno unicamente come conseguenza, se non relazionati ai punti, un alto errore.

L'idea che quindi ho elaborato è quella di determinare il punto di utopia in base ai valori dei periodi delle sinusoidi individuate. La coordinata rispetto all'errore viene fissata a 0, mentre per la coordinata rispettiva al periodo, calcolo la media dei periodi delle sinusoidi. Questo si basa sull'ipotesi che se la maggior parte dei periodi, individuati dal solutore, cade in un intorno di un determinato valore, allora tale valore è il periodo della sinusoide ricercata. Determino la soluzione da selezionare scegliendo la soluzione più vicina, valutando la distanza euclidea di ogni soluzione rispetto al punto di utopia. Illustro esempi di generazione e scelta di una soluzione:

I problemi scelti per la valutazione e analisi del comportamento del criterio del punto di utopia, sono problemi definiti per 10, 20, 30 punti generati da una sinusoide con un determinato periodo. I parametri dell'algoritmo 2 per le seguenti sperimentazioni sono:

- $f(x)$  in riga 9, algoritmo 1 pari alla funzione (9) con  $k = 2$
- $periodoIniziale = 0s$
- $periodoFinale = 5000s$
- $tolleranzaDefault = 1e-6$
- $e(t) = \sin(0.0010t)$  sinusoide con periodo pari a:  $T = \frac{2\pi}{\omega} = \frac{2\pi}{0.0010} = 6280s$

Tabella 11: periodo da individuare uguale a 6280s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	662	4.5	2.7e-8
20	499	5.4	3.6e-5
30	1342	3.09	1.7e-4

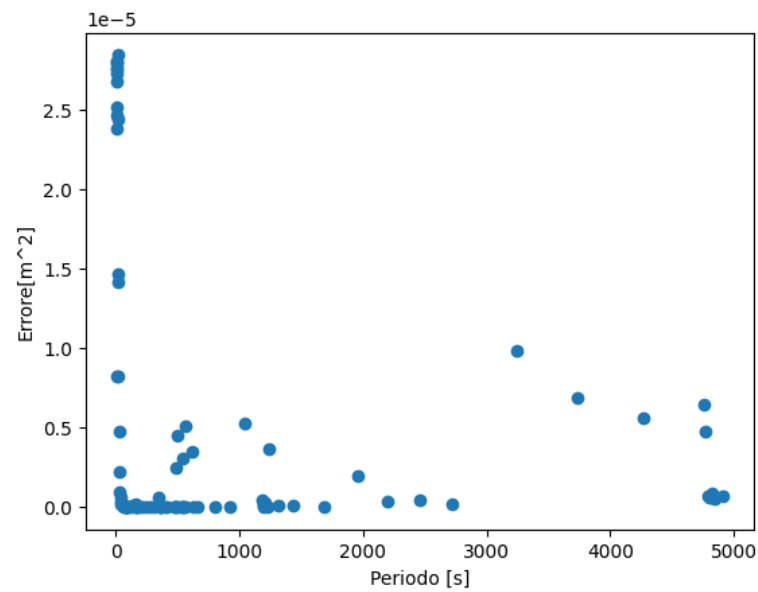


Figura 2: Regione paretiana per problema da 10 punti di tabella 11

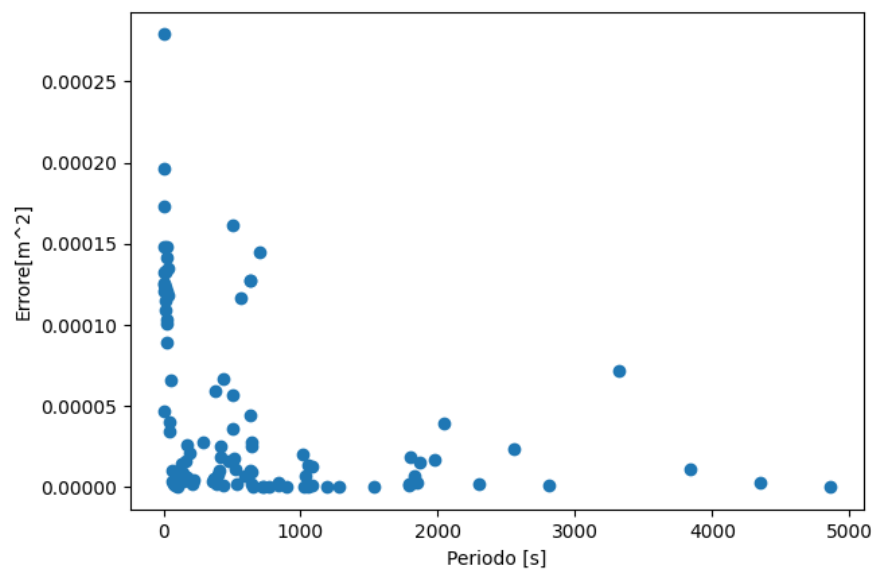


Figura 3: Regione paretiana per problema da 20 punti di tabella 11

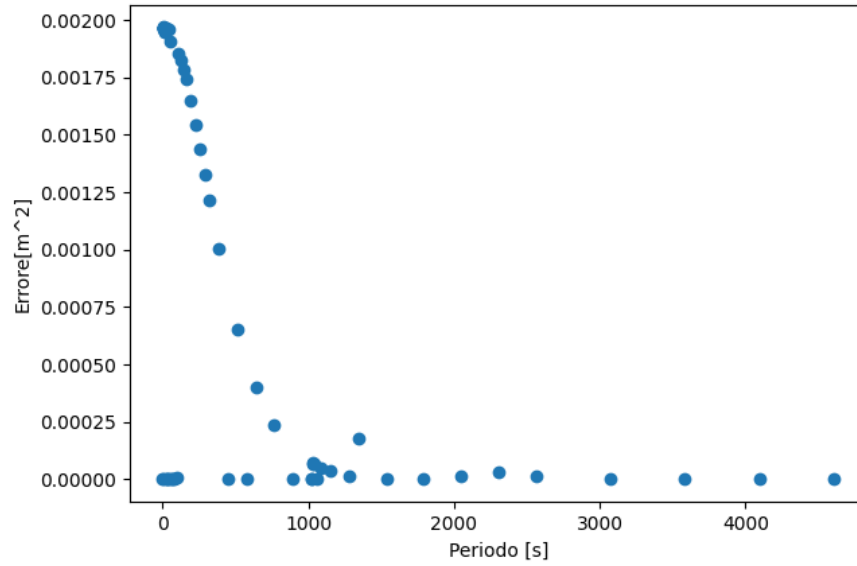


Figura 4: Regione paretiana per problema da 30 punti di tabella 11

- $e(t) = \sin(0.0013t)$  sinusoide con periodo pari a:  $T = 4830.7s$

Tabella 12: periodo da individuare uguale a 4830.7s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	1201	7.5	1.3e-11
20	2095	10.3	7.5e-5
30	4175	10.9	3.2e-4



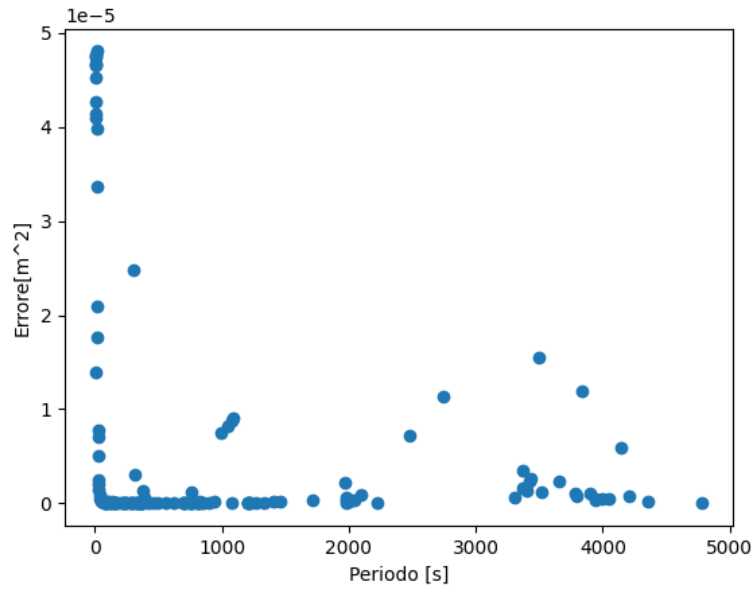


Figura 5: Regione paretiana per problema da 10 punti di tabella 12

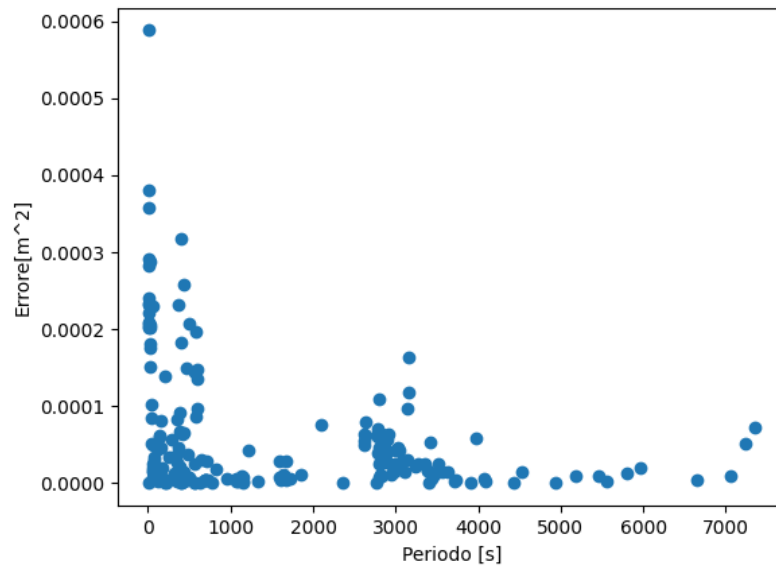


Figura 6: Regione paretiana per problema da 20 punti di tabella 12

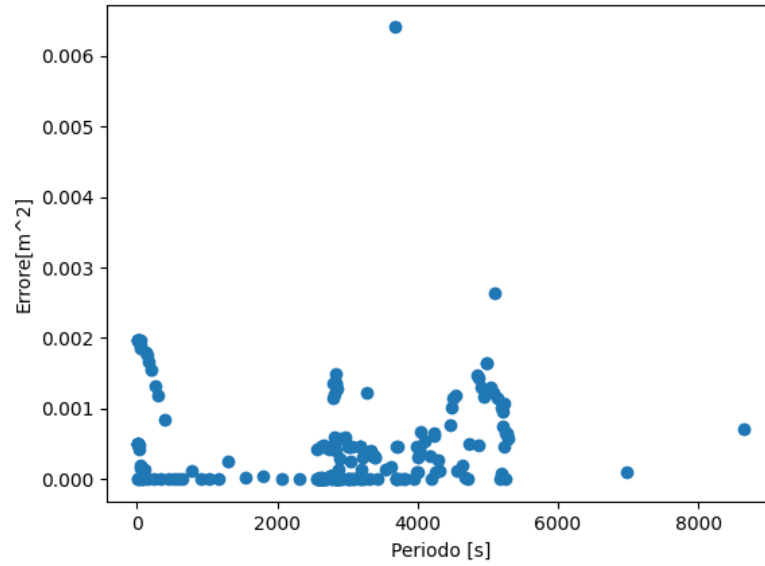


Figura 7: Regione paretiana per problema da 30 punti di tabella 12

- $e(t) = \sin(0.005t)$  sinusoide con periodo pari a:  $T = 1256s$

Tabella 13: periodo da individuare uguale a 1256s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	753	6.8	$1.3e-6$
20	1321	16.2	$1.7e-4$
30	1681	8.2	$9.6e-4$

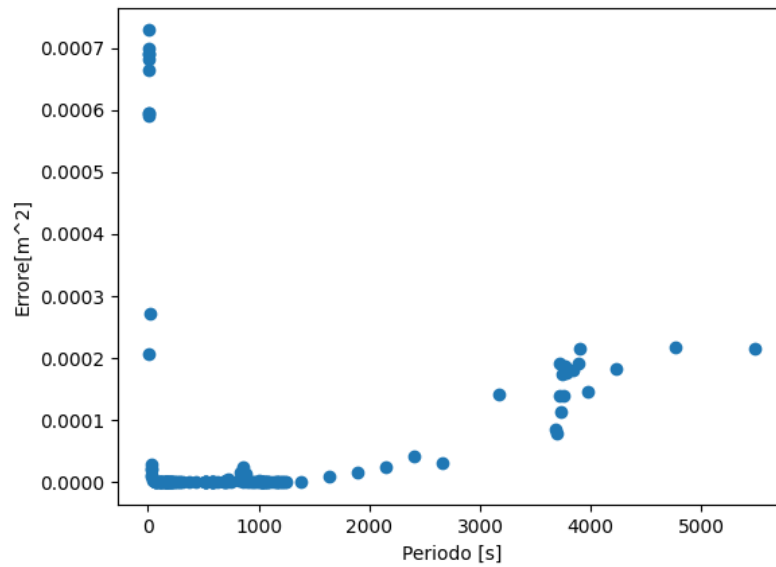


Figura 8: Regione paretiana per problema da 10 punti di tabella 13

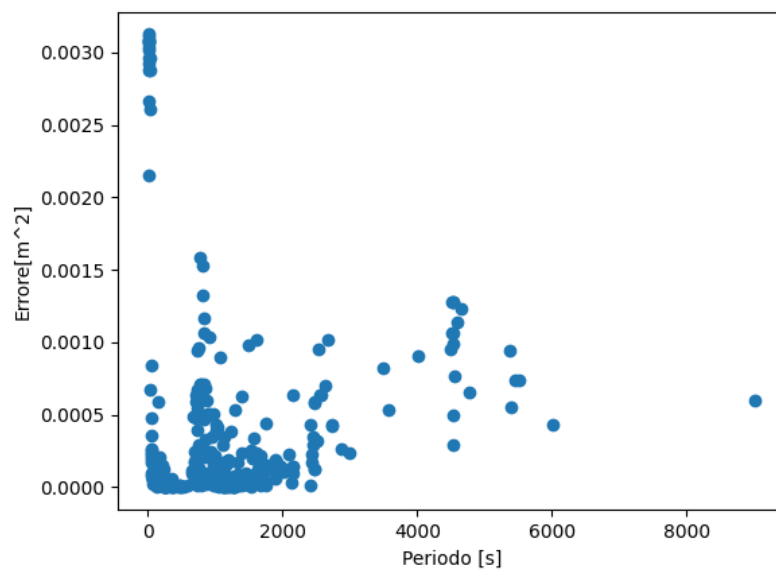


Figura 9: Regione paretiana per problema da 20 punti di tabella 13

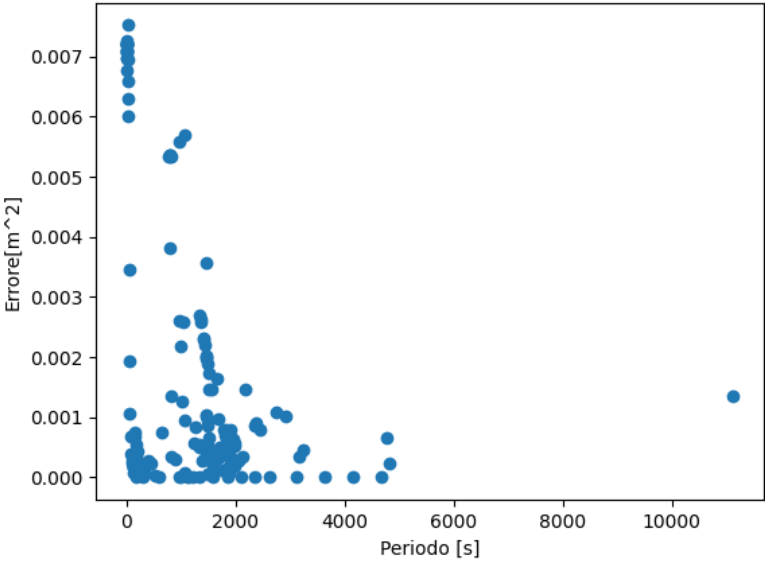


Figura 10: Regione paretiana per problema da 30 punti di tabella 13

- $e(t) = \sin(0.0125t)$  sinusoide con periodo pari a:  $T = 502.4s$

Tabella 14: periodo da individuare uguale a 502.4s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	425	6.4	$2.5e-10$
20	701	18.1	$2.7e-4$
30	589	9.3	$6.0e-4$

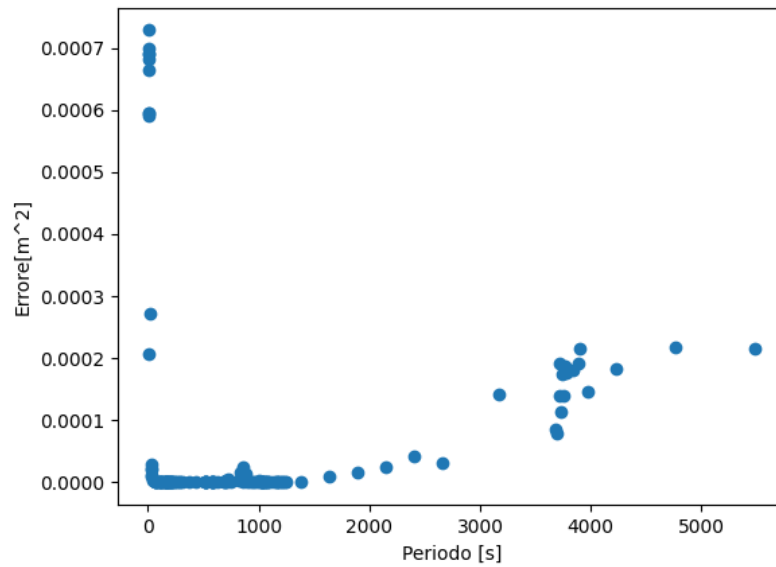


Figura 11: Regione paretiana per problema da 10 punti di tabella 14

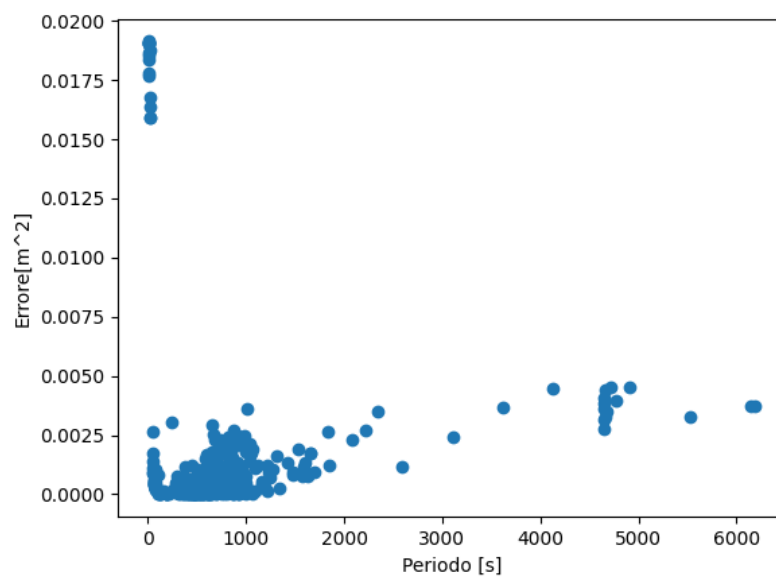


Figura 12: Regione paretiana per problema da 20 punti di tabella 14

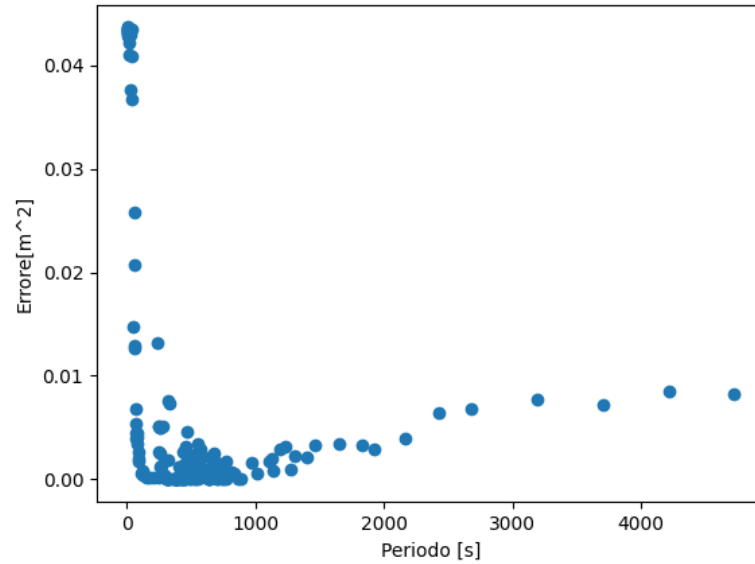


Figura 13: Regione paretiana per problema da 30 punti di tabella 14

- $e(t) = \sin(0.8t)$  sinusoide con periodo pari a:  $T = 7.85s$

Tabella 15: periodo da individuare uguale a 7.85s






Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	40	2.2	$1.3e-6$
20	8313	97.0	$1.7e-4$
30	4519	12.1	$9.6e-4$

L'esecuzione dell'algoritmo è caratterizzato da uno scostamento medio dal periodo da individuare di  $2377s$  e una mediana di  $1278s$  ed infine un tempo medio di esecuzione di  $13.62s$  e una mediana di  $6.8s$ .

La mediana e la media dello scostamento denotano un alta differenza tra la soluzione scelta dall'algoritmo e la sinusoide del problema, la causa di ciò è da individuarsi nel posizionamento del punto di utopia ed una mancata normalizzazione dei domini dell'errore e periodo.

Prendendo ad esempio la tabella 13, in particolare la soluzione individuata per il problema da 20 punti ed il suo grafico delle soluzioni generate, figura 14 e figura 15, si può notare che una soluzione con scostamento minore è stata individuata, ma non scelta dall'algoritmo di selezione poiché la distanza euclidea dal punto di utopia risulta maggiore rispetto alla soluzione scelta.

Legenda per i grafici da figura 14 a figura 20:

-  : soluzione individuata.
-  : soluzione individuata scelta dal criterio.
-  : punto di utopia.
-  : soluzione individuata ideale.
-  : punto della sinusoide del problema.

Da come si può notare dal primo grafico di figura 14 il punto di utopia risulta essere traslato al di fuori della zona dove la maggior parte delle soluzioni sono situate, a causa di soluzioni outlier.

È necessario quindi implementare un metodo di determinazione del periodo del punto di utopia migliore o passare ad un altro criterio di selezione.

Un miglioramento può essere l'implementazione di una media pesata, dove si determina il peso in base all'errore della soluzione, questo permette di dare più rilevanza alle soluzioni con errore minore, comprendendo però in questo modo anche soluzioni con basso periodo rispetto al periodo della sinusoide da individuare.

Una soluzione alternativa può essere la modifica dell'algoritmo di controllo ed esplorazione dei periodi, modificando la gestione del passo. Diminuendo il passo quando si individua una soluzione con errore minore di un determinato valore di soglia, si approfondisce e si aumenta il numero di soluzioni individuate in un intorno di esso.

L'altra probabile causa è la mancata normalizzazione dei domini dell'errore e del periodo. Si possono notare gli effetti di ciò nel grafico in figura 17 corrispondente alla tabella 15 per il problema da 20 punti.

Si può notare, dai grafici in figura 18 e 19 che la distanza tra il punto di utopia e la soluzione con periodo che si scosta meno dalla soluzione da individuare, è approssimativamente pari ad un valore di 8000s, molto maggiore rispetto ad una soluzione che ha lo stesso periodo del punto di utopia ma con uno scostamento maggiore, pari a 9.5

Un altro fattore che ha determinato la mancata scelta della soluzione ideale, mostrata in figura 18, sono il numero di soluzioni individuate in un suo intorno limitato,

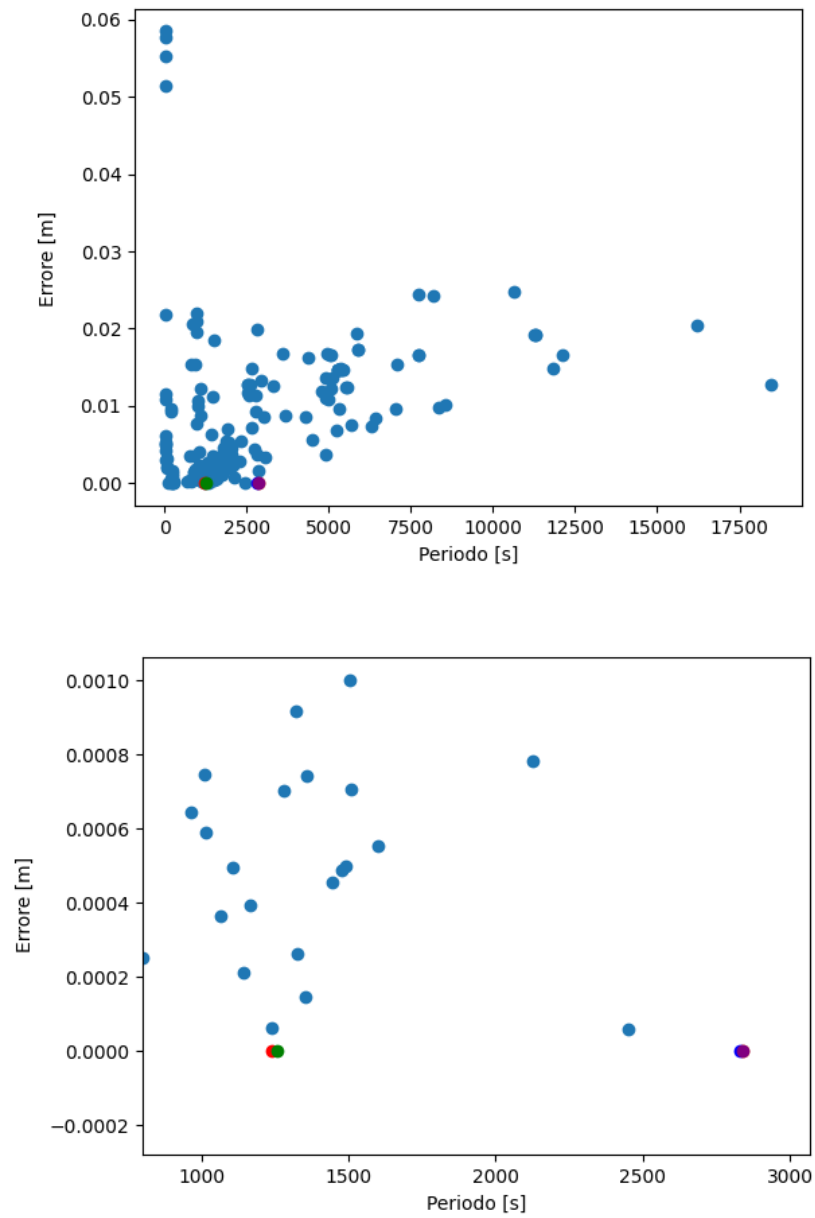


Figura 14: Soluzioni generate per il problema di 13 con 20 punti



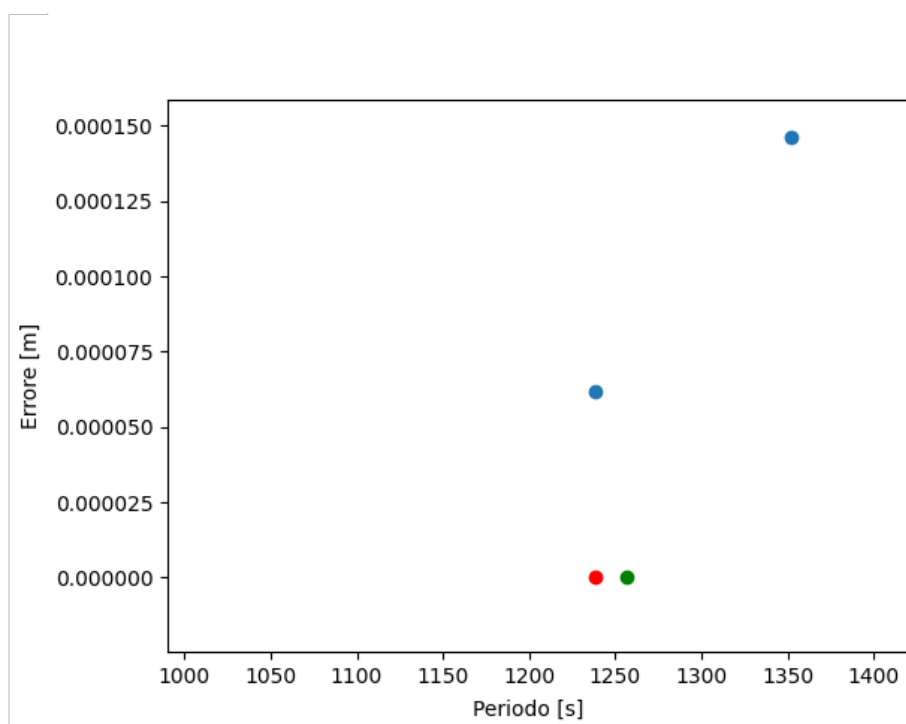


Figura 15: soluzione ideale per il problema di tabella 13 con 20 punti

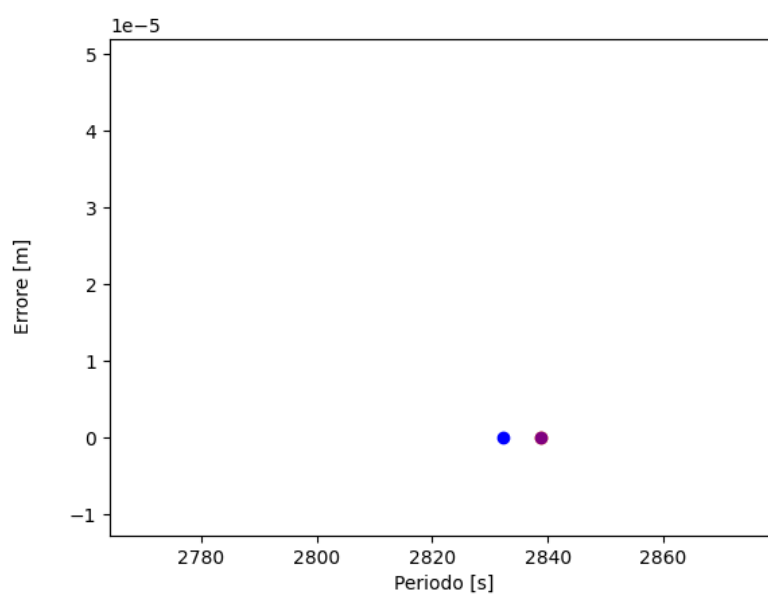


Figura 16: soluzione scelta per il problema di tabella 13 con 20 punti

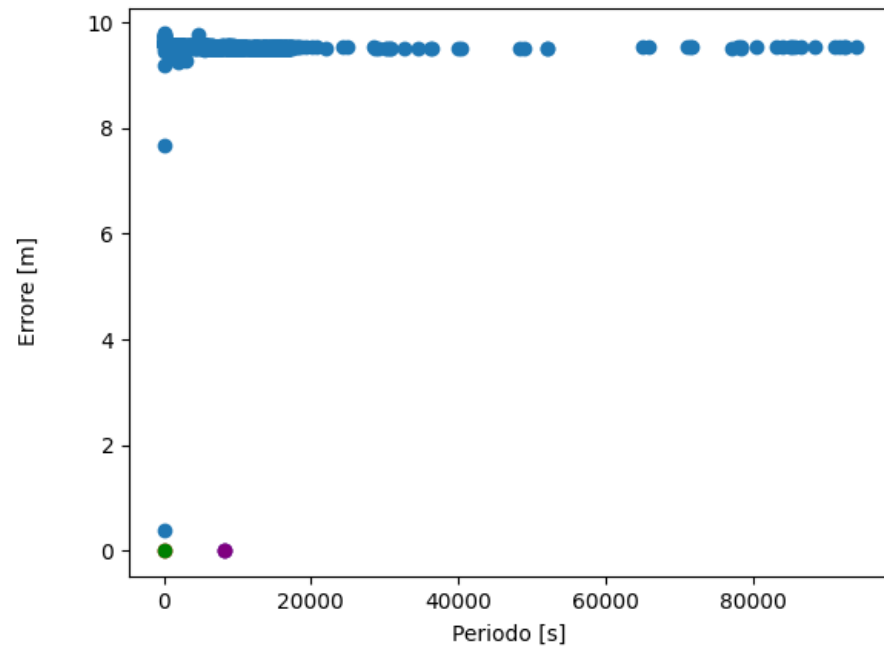


Figura 17: Soluzioni generate per il problema di tabella 15 con 20 punti

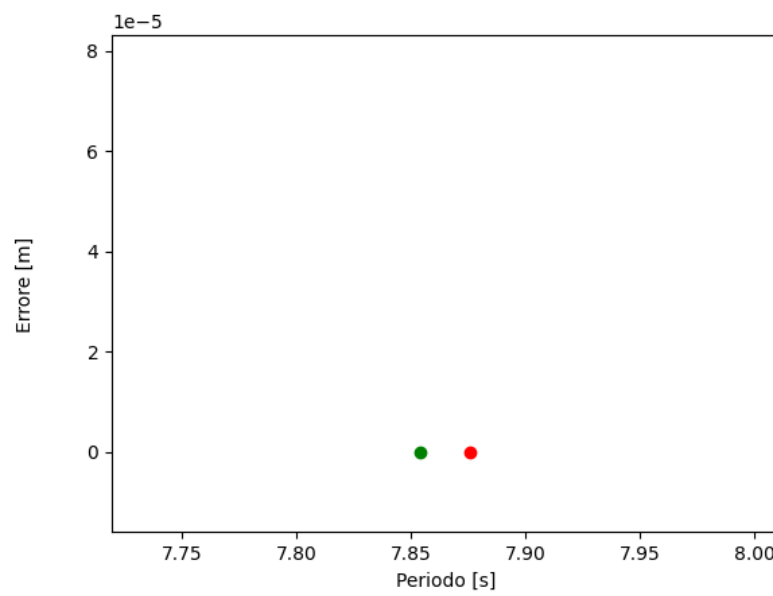


Figura 18: Soluzione ideale per il problema di 15 con 20 punti

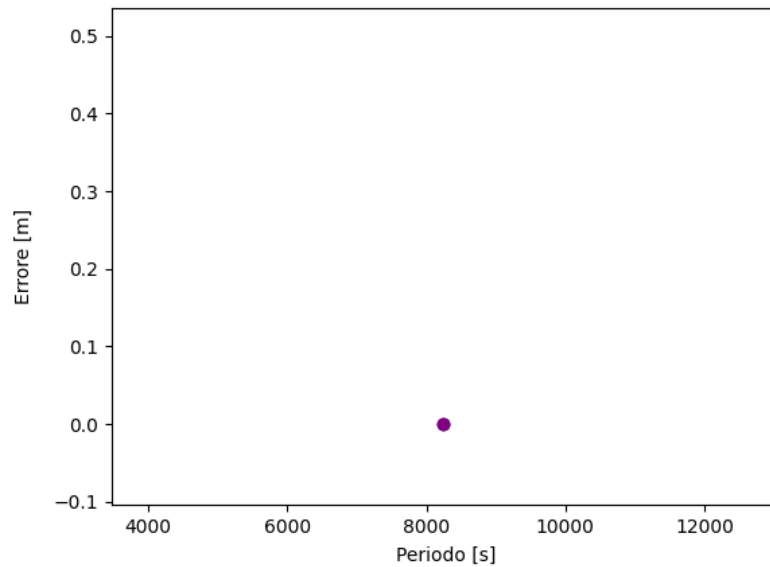


Figura 19: Posizione del punto di utopia per il problema di 15 con 20 punti

come per esempio nell'insieme dei periodi  $[0, 20]$  illustrato in figura 20, dove sono individuate solo 6 soluzioni rispetto al numero totale di soluzioni individuate pari a 1527 in questo esempio.

Il problema di questa metodologia di specifica del periodo per il punto di utopia è che è basata in maniera marcata sulle prestazioni, il numero e la tipologia di soluzioni individuate dall'algoritmo di esplorazione dei periodi.

Riassumendo, per il miglioramento del criterio di selezione esplicito le seguenti possibili modifiche:

- La normalizzazione dei domini di errore e di periodo per tutte le soluzioni individuate
- L'implementazione di una media pesata e quindi il calcolo del peso per ogni soluzione individuata in base al suo errore.
- Aggiunta di condizione di diminuzione del passo nel algoritmo esplorazione dei periodi.

Dato le ampie modifiche richieste, ho optato per l'individuazione e definizione di un altro criterio di selezione.

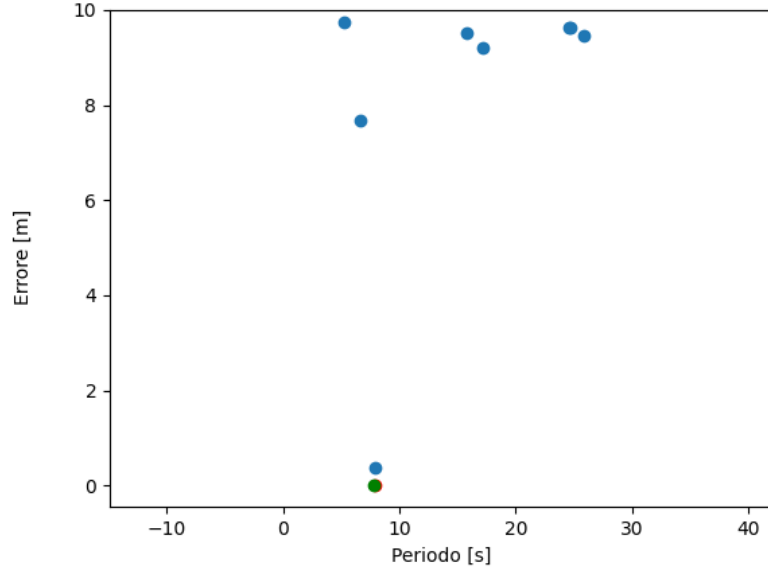


Figura 20: Soluzioni individuate nell'insieme di periodi  $[0, 20]$  per il problema di tabella 15 con 20 punti

### Criterio degli standard sull'errore

Gli standard sono valori-soglia al di sotto dei quali non si vuole che gli obiettivi possano peggiorare. Data la non predicibilità del periodo ottimale in questo problema, è possibile definire gli standard solo per l'errore. La modalità con il quale ho definito gli standard per la scelta della soluzione prevede la suddivisione in range del dominio dell'errore, ipotizzando che ogni soluzione in uno stesso range sia equivalente, se si considera solo l'errore. Ogni range è identificato con un numero ordinale specificandoli come livelli: primo livello, secondo livello, ..., n-esimo livello, dove per ordine crescente dei livelli si ha l'aumento del range d'errore. I range devono essere definiti a priori, in maniera attenta rispetto all'errore intrinseco dello strumento di rilevazione dei punti. È necessario inoltre effettuare ipotesi sul errore di una soluzione: errore di una soluzione ottimale, errore di una soluzione non correlata ai punti, errore di una soluzione legata ad un sottoproblema con un assegnamento di punti errato.

Il seguente pseudocodice illustra una procedura di una struttura dati che ho definito, per contenere la soluzione con livello più basso possibile e con più alto periodo tra tutte le soluzioni individuate all'interno del corrispettivo livello, perciò gli standard dell'errore sono fissati dalla soluzione con il livello minore.

Tutte le soluzioni con un livello maggiore della soluzione ottimale vengono scartate (riga 1, algoritmo 3), invece se la soluzione corrente ha un livello inferiore, essa viene

---

**Algoritmo 3:** Criterio degli standard

---

```

input : solCorrente tupla contenente (errore, periodo, ampiezza, fase)
1 if LivelloDi(solCorrente) < LivelloDi(solOttimale) then
2   | solOttimale  $\leftarrow$  solCorrente
3 else
4   | if LivelloDi(solCorrente) = LivelloDi(solOttimale) then
5     | if LivelloDi(solCorrente) = ultimoLv then
6       | if solCorrente[errore] < solOttimale[errore] then
7         | | solOttimale  $\leftarrow$  solCorrente
8       | end
9     | else
10      | if solCorrente[periodo] > solOttimale[periodo] then
11        | | solOttimale  $\leftarrow$  solCorrente
12      | end
13    | end
14  end
15 end

```

---

memorizzata come soluzione ottimale. Se i livelli della soluzione ottimale e della soluzione corrente si equivalgono, si memorizza la soluzione corrente solo se essa ha un periodo maggiore della soluzione ottimale (riga 10, algoritmo 3).

Solo l'ultimo livello ha un criterio di selezione della soluzione ottimale differente, poiché si seleziona la soluzione con il minimo errore (riga 6, algoritmo 3). L'ultimo livello, rappresenta l'insieme dei valori d'errore più alti, che vanno da un valore fissato a  $+\infty$ , perciò all'interno di questo insieme illimitato, cerco di selezionare la soluzione che più si correla all'assegnamento dei punti.

Illustro i risultati dell'implementazione ed esecuzione del criterio degli standard, generando le soluzioni utilizzando il modello basato sul metodo dei vincoli, variando il minimo periodo  $\beta$  nel range fissato (sezione 3.2.3).

I livelli fissati sono

- Livello 1: errore <  $1e-5$
- Livello 2: errore <  $1e-2$
- Livello 3: errore <  $1e-1$
- Livello 4: errore < 1
- Livello 5: errore < 3

- Livello 6: errore  $< 5$
- Livello 7: errore  $< 7$
- Livello 8: errore  $< 10$
- Livello 9: errore  $\geq 10$

Ho fissato i range in maniera da ottenere maggiore precisione e diversificazione delle soluzioni con errore tra 0 e 1, suddividendo tale spazio in 4 livelli. Nella definizione dei livelli, non ho tenuto in considerazione l'errore intrinseco dello strumento di rilevazione, per ottenere delle prime informazioni sulla prestazione dell'algoritmo in una situazione ideale. A seguito si elencano problemi con punti generati da una sinusoide  $e(t)$  e nelle tabelle le soluzioni individuate dall'algoritmo.

I parametri dell'algoritmo 2 per le seguenti sperimentazioni sono:

- $f(x)$  in riga 9, algoritmo 1 pari alla funzione (9) con  $k = 5$
- $periodoIniziale = 0s$
- $periodoFinale = 5000s$
- $tolleranzaDefault = 1e-6$
- $e(t) = \sin(0.0010t)$  sinusoide con periodo pari a:  $T = \frac{2\pi}{\omega} = \frac{2\pi}{0.0010} = 6280s$

Tabella 16: periodo da individuare uguale a 6280s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	4631.8	2.2	9.0e-7
20	5585.4	1.9	1.0e-34
30	4906.0	2.0	6.0e-4

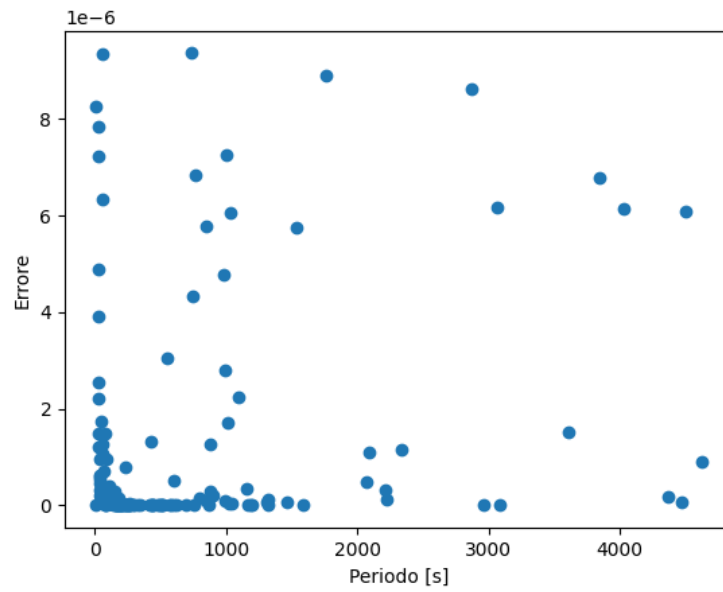


Figura 21: Regione paretiana per problema da 10 punti di tabella 16

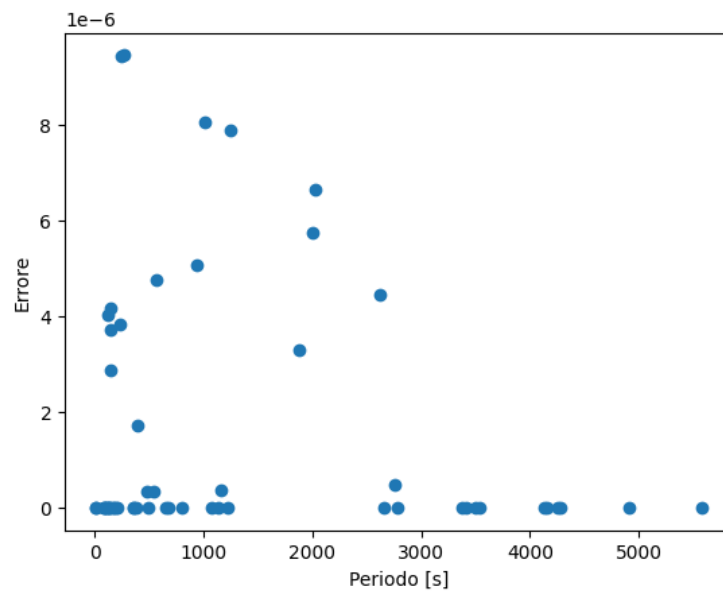


Figura 22: Regione paretiana per problema da 20 punti di tabella 16

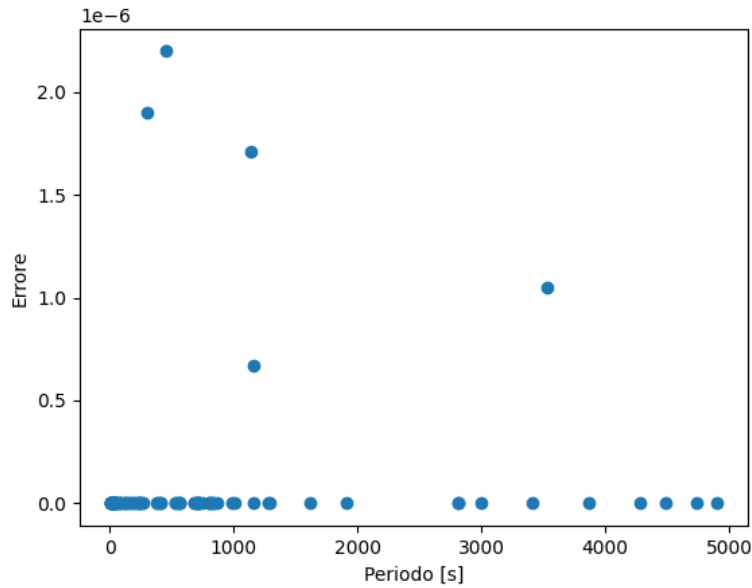


Figura 23: Regione paretiana per problema da 30 punti di tabella 16

- $e(t) = \sin(0.0013t)$  sinusoide con periodo pari a:  $T = 4830.7s$

Tabella 17: periodo da individuare uguale a 4830.7s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	4471.5	2.2	$6.4e-8$
20	4907.0	2.0	0.0
30	4492.0	2.3	0.0



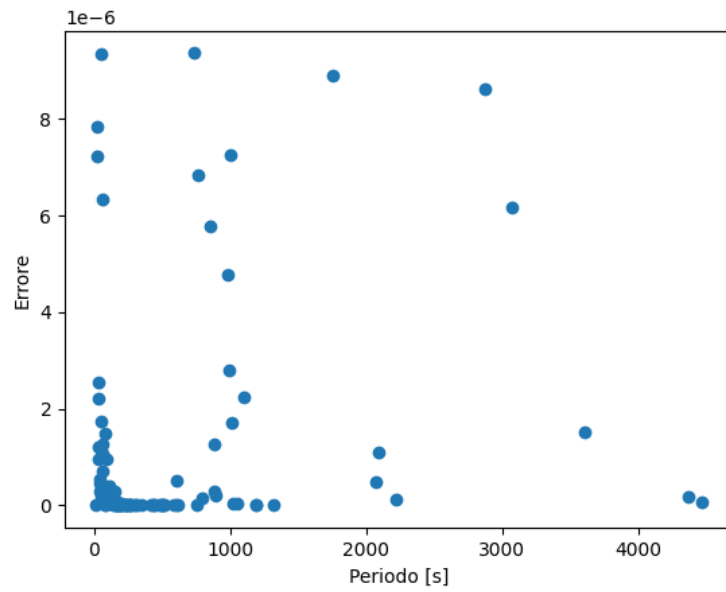


Figura 24: Regione paretiana per problema da 10 punti di tabella 17

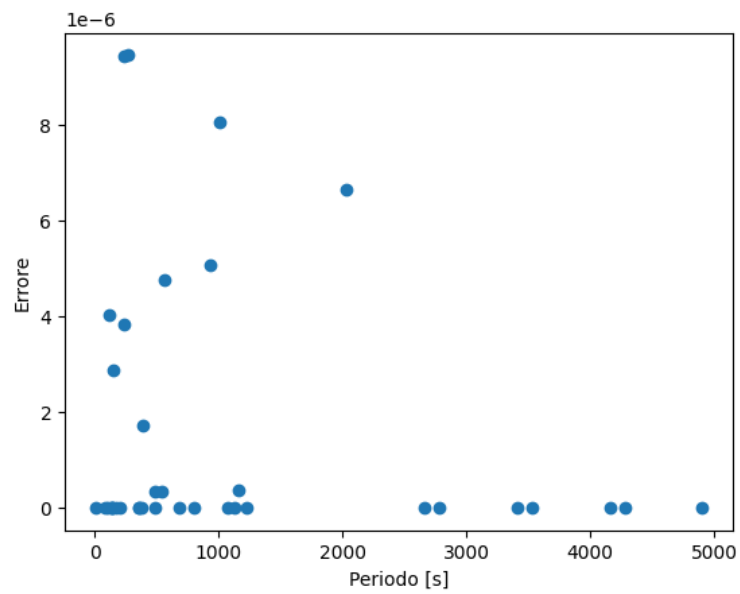


Figura 25: Regione paretiana per problema da 20 punti di tabella 17

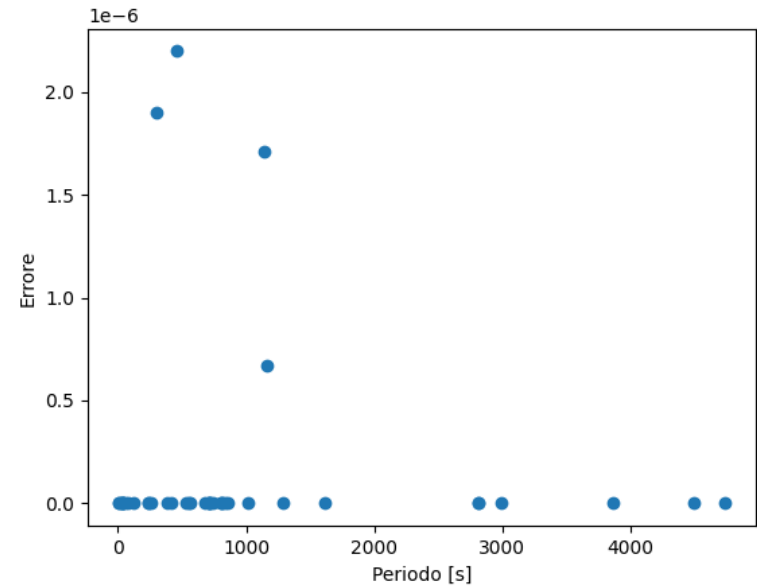


Figura 26: Regione paretiana per problema da 30 punti di tabella 17

- $e(t) = \sin(0.005t)$  sinusoide con periodo pari a:  $T = 1256s$

Tabella 18: periodo da individuare uguale a 1256s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	1759.0	2.1	8.9e−6
20	1224.0	2.6	0.0
30	4740	2.0	0.0

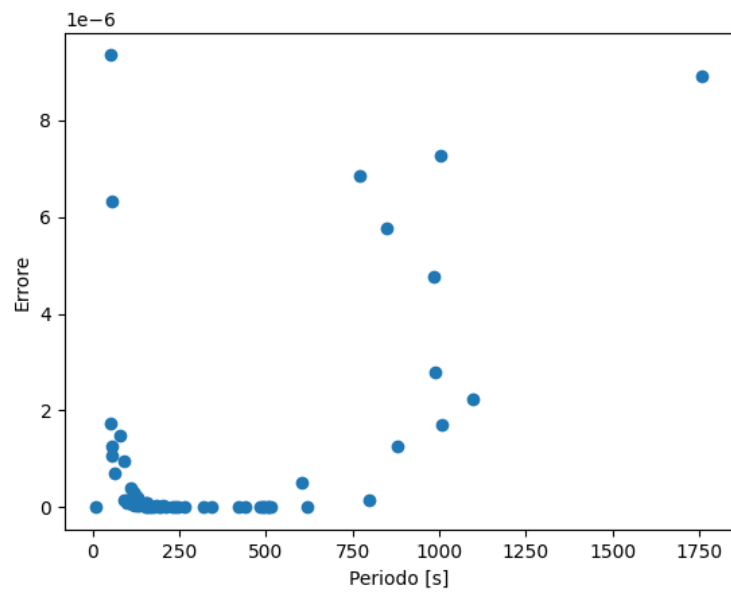


Figura 27: Regione paretiana per problema da 10 punti di tabella 18

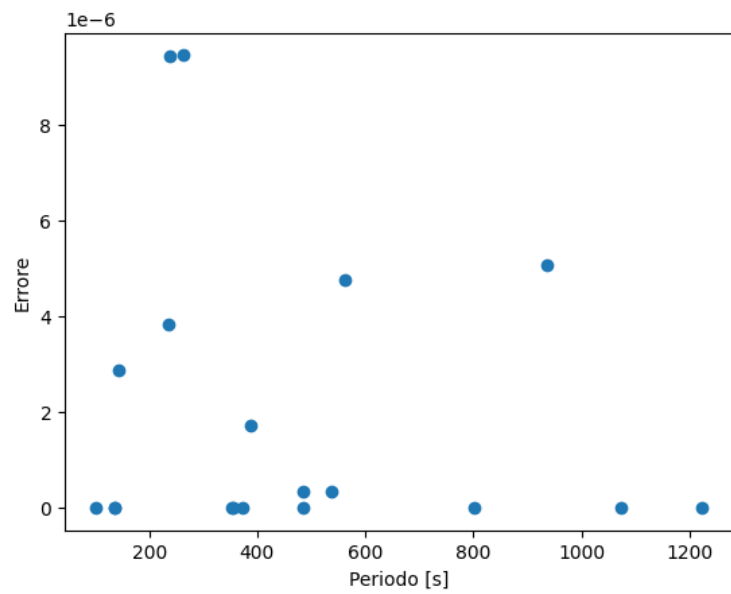


Figura 28: Regione paretiana per problema da 20 punti di tabella 18

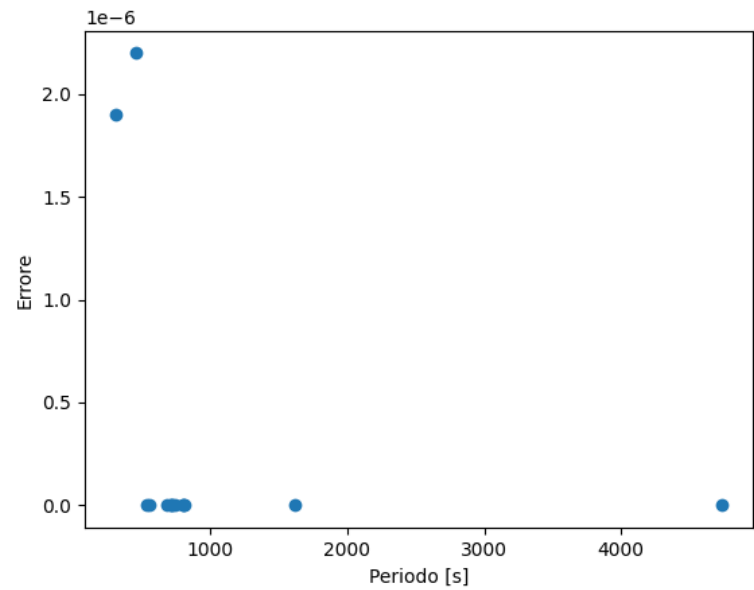


Figura 29: Regione paretiana per problema da 10 punti di tabella 18

- $e(t) = \sin(0.0125t)$  sinusoide con periodo pari a:  $T = 502.4s$

Tabella 19: periodo da individuare uguale a 502.4s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	602.0	2.7	5.1e−7
20	801.9	6.0	0.0
30	743.0	2.2	0.0

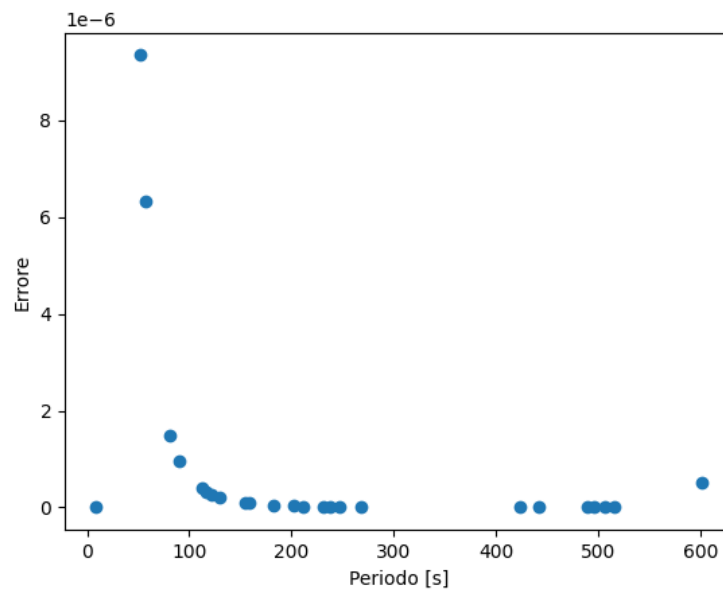


Figura 30: Regione paretiana per problema da 10 punti di tabella 19

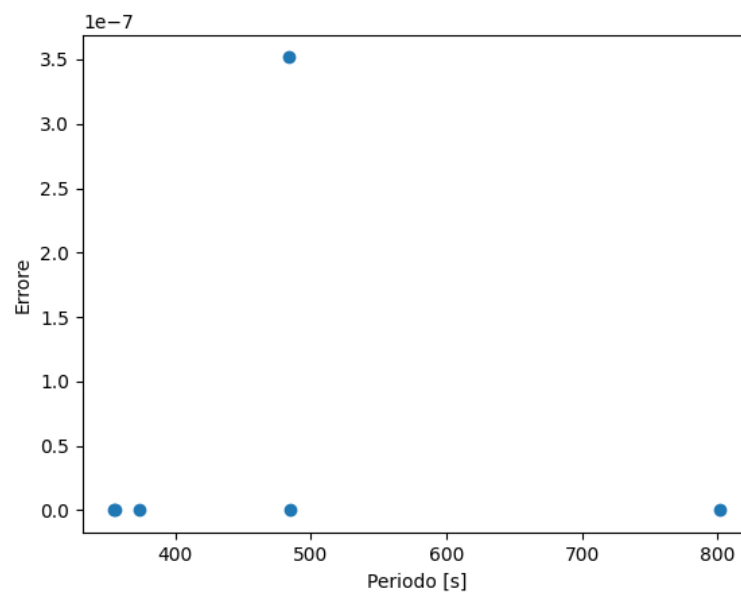


Figura 31: Regione paretiana per problema da 20 punti di tabella 19

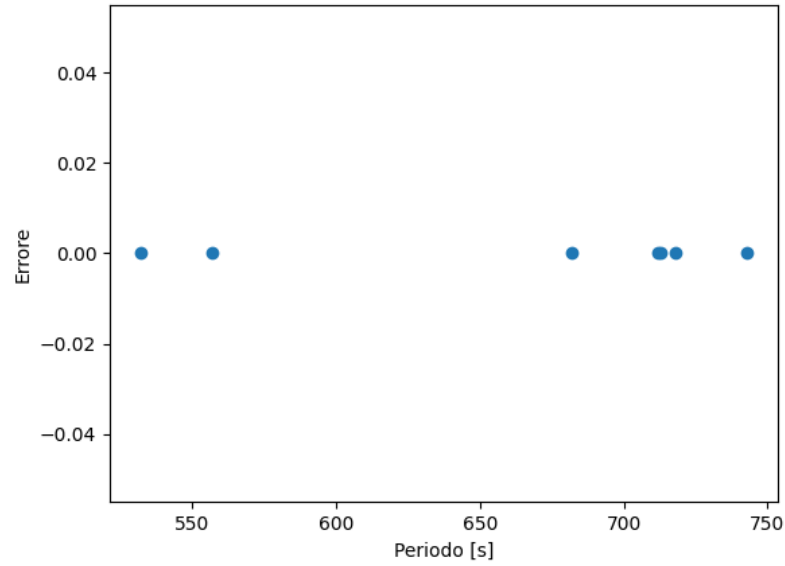


Figura 32: Regione paretiana per problema da 30 punti di tabella 19

- $e(t) = \sin(0.8t)$  sinusoide con periodo pari a:  $T = 7.85s$

Tabella 20: periodo da individuare uguale a 7.85s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	7.8	1.8	5.0e−9
20	8.0	4.1	1.1e−2
30	8.0	2.5	2.5e−2

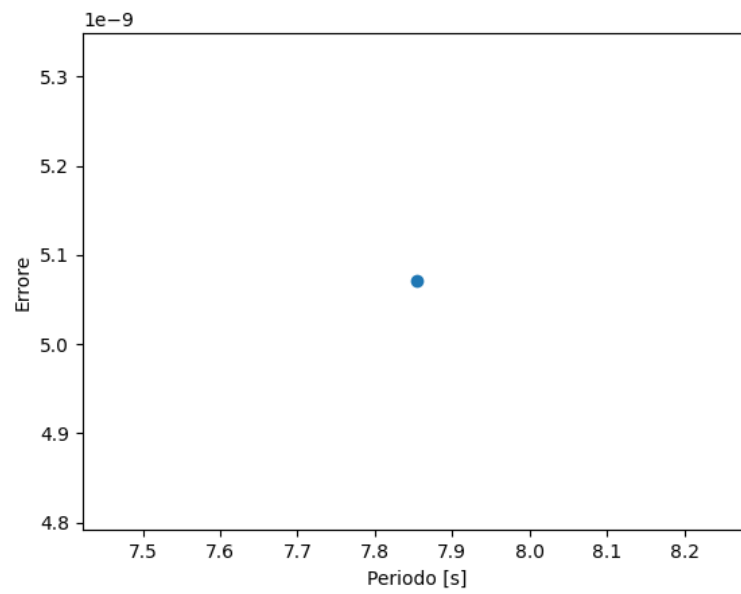


Figura 33: Regione paretiana per problema da 10 punti di tabella 20

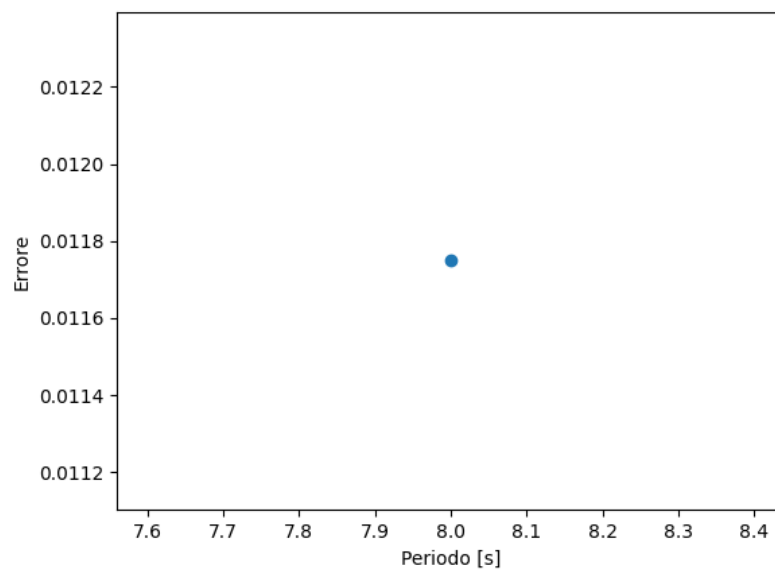


Figura 34: Regione paretiana per problema da 10 punti di tabella 20

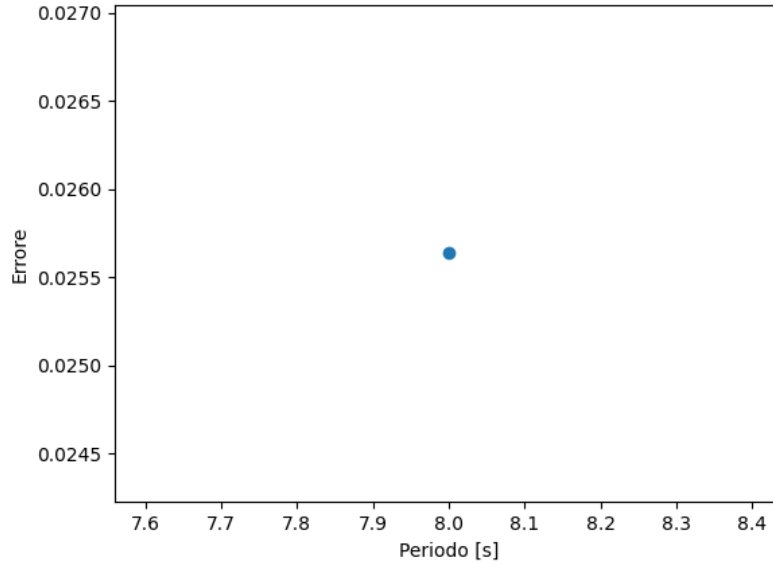


Figura 35: Regione paretiana per problema da 30 punti di tabella 20

L'esecuzione dell'algoritmo è caratterizzato da uno scostamento medio dal periodo da individuare di  $610.0s$  e una mediana di  $299.3s$  ed infine un tempo medio di esecuzione di  $2.5s$  e una mediana di  $2.2s$ .

È possibile provare a diminuire il tempo di esecuzione modificando l'algoritmo di controllo ed esplorazione, aggiungendo come condizione di aumento del passo, l'individuazione di una soluzione dal livello maggiore di un livello di soglia. Questo permette una velocizzazione dell'esplorazione saltando tutti quei periodi di partenza lontani dal periodo della sinusoide da individuare.

Inoltre si può notare che l'errore delle soluzioni individuate si distribuisce in un range pari a  $[0.0, 2.5e-2]$ , questo potrebbe suggerire la definizione di livelli dai range più stringenti come ad esempio:

- Livello 1: errore  $< 1e-9$
- Livello 2: errore  $< 1e-7$
- Livello 3: errore  $< 1e-5$
- Livello 4: errore  $< 1e-3$
- Livello 5: errore  $< 1e-1$



- Livello 6: errore  $< 1$
- Livello 7: errore  $< 3$
- Livello 8: errore  $< 5$
- Livello 9: errore  $\geq 5$

Ma eseguendo i problemi da periodo elevato come i problemi di tabella 16 e 17, si può notare nelle tabelle successive 21 e 22, che le sinusoidi individuate per i problemi da 10 punti, si scostano mediamente di 4409.1s dalla sinusoidi da individuare, evidenziando il problema delle sinusoidi da basso periodo rispetto alla sinusoidi da individuare.

- $e(t) = \sin(0.0010t)$  sinusoidi con periodo pari a:  $T = \frac{2\pi}{\omega} = \frac{2\pi}{0.0010} = 6280s$

Tabella 21: periodo da individuare uguale a 6280s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	1587.0	1.29	5.0e-10
20	4388	0.65	0.0
30	4907.0	0.9	0.0

- $e(t) = \sin(0.0013t)$  sinusoidi con periodo pari a:  $T = 4830.7s$

Tabella 22: periodo da individuare uguale a 4830.7s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	704	2.2	6.4e-8
20	4478.0	0.77	0.0
30	4935.0	0.8	0.0

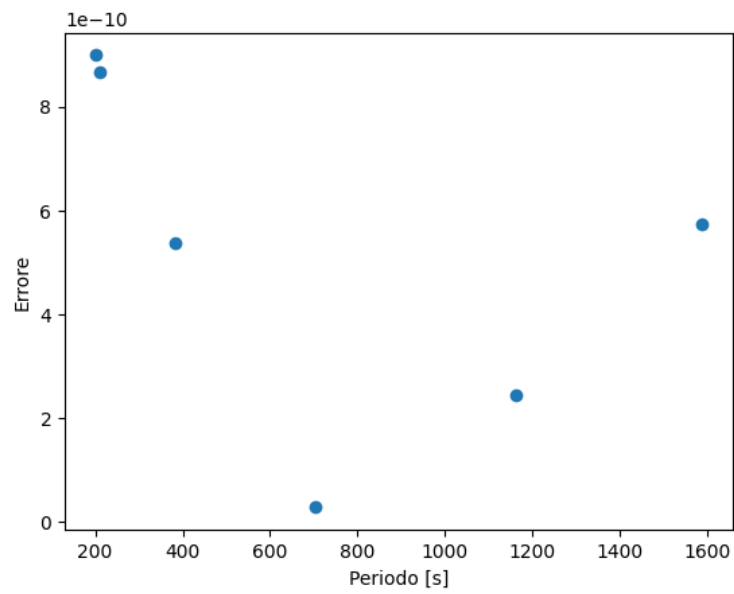


Figura 36: Regione paretiana per problema da 10 punti di tabella 21

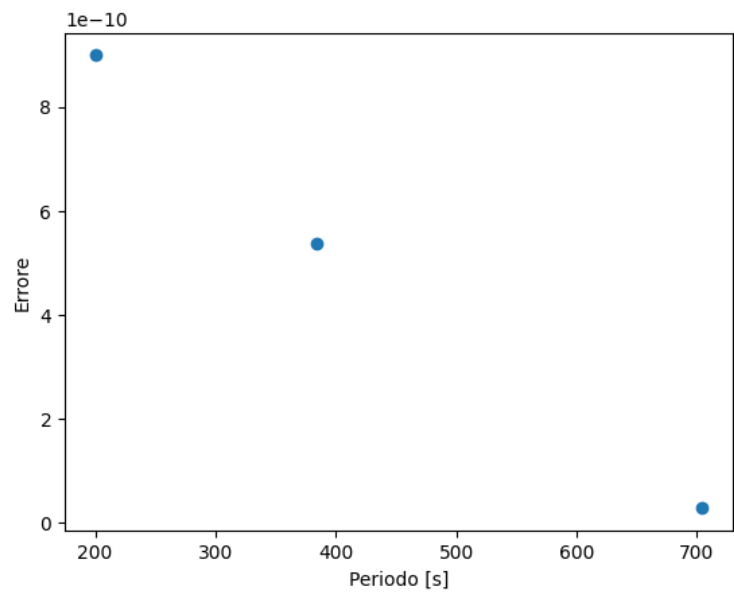


Figura 37: Regione paretiana per problema da 10 punti di tabella 22

Perciò ho deciso di mantenere i livelli precedentemente definiti.

Infine un'ulteriore modifica per l'accelerazione dell'algoritmo è agire sul parametro  $k$  della  $f(x)$  in riga 9, algoritmo 1, fissandolo pari a 10 il doppio del valore fissato precedentemente.

Effettuando questa modifica e scegliendo come livello di soglia il quinto, si ottengono i seguenti risultati:

- $e(t) = \sin(0.0010t)$  sinusoide con periodo pari a:  $T = \frac{2\pi}{\omega} = \frac{2\pi}{0.0010} = 6280s$

Tabella 23: periodo da individuare uguale a 6280s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	4610.8	1.2	9.8e-7
20	4461.0	1.2	0.0
30	4906.0	1.2	0.0

- $e(t) = \sin(0.0013t)$  sinusoide con periodo pari a:  $T = 4830.7s$

Tabella 24: periodo da individuare uguale a 4830.7s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	4582.0	1.0	4.3e-8
20	4942.0	1.2	0.0
30	4461.0	1.2	0.0

- $e(t) = \sin(0.005t)$  sinusoide con periodo pari a:  $T = 1256s$

Tabella 25: periodo da individuare uguale a 1256s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	1683.0	1.3	8.2e-6
20	1224.0	1.8	0.0
30	4320.4	2.3	0.0

- $e(t) = \sin(0.0125t)$  senoide con periodo pari a:  $T = 502.4s$

Tabella 26: periodo da individuare uguale a 502.4s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	498.1	1.8	3.1e-7
20	706.0	2.2	0.0
30	756.0	1.4	0.0

- $e(t) = \sin(0.8t)$  senoide con periodo pari a:  $T = 7.85s$

Tabella 27: periodo da individuare uguale a 7.85s

Numero di punti	Periodo della soluzione (s)	Tempo di calcolo (s)	Errore quadratico medio ( $m^2$ )
10	7.8	1.1	5.7e-5
20	8.0	1.2	1.1e-2
30	8.0	1.2	2.5e-2

Così ottenendo uno scostamento medio di 648.1s e con mediana pari a 248.7s e un tempo medio di esecuzione di 1.4s con mediana pari a 1.2s. Si ottiene così una diminuzione dei tempi di esecuzione.

## Capitolo 4

# Sviluppo dell'algoritmo

Date  $n$  osservazioni composti da  $k$  valori, dove  $k$  è il numero di satelliti, l'algoritmo di assegnamento ha il compito di definire e valutare gli assegnamenti di  $n$  valori, presi da  $n$  osservazioni differenti. Lo scopo è di individuare gli assegnamenti che identificano le sinusoidi corrispondenti al moto dei satelliti.

Tabella 28: Esempio di struttura dati per le osservazioni

	1° satellite	2° satellite	...	k-esimo satellite
1° osservazione	val	val	...	val
2° osservazione	val	val	...	val
...	val	val	...	val
$k -esima$ osservazione	val	val	...	val

### 4.1 Struttura e approccio

Il concetto alla base è quello di generare tutte le permutazioni possibili delle  $n$  osservazioni, e di valutarle tramite i parametri restituiti dal modulo di interpolazione. Il numero delle permutazioni è pari a  $(k!)^n$ .

La permutazione di una singola osservazione produce  $k!$  elementi, ognuno di questi elementi deve essere assegnato a tutti i  $k!$  elementi delle successive osservazioni fino alla  $n$ -esima osservazione:

$$\prod_{i=1}^n k! \quad (10)$$

Il numero delle permutazioni evidenzia il costo estensivo del processo e comporta l'applicazione di metodologie atti a ridurre il numero di permutazioni valutate per l'individuazione degli assegnamenti corretti in tempo utile.

Per queste motivazioni, ho implementato un algoritmo divide et impera, che divide il problema di assegnamento in sottoproblemi dal numero di osservazioni minore e unisce i risultati ottenuti nei vari nodi fino ad ottenere la soluzione del problema originario.

#### 4.1.1 Creazione dei sottoproblemi

Per la generazione dei sottoproblemi, sfrutto la tecnica della ricorsione, generando un albero di chiamate dove ogni nodo corrisponde ad un sottoproblema. I nodi figli sono definiti in un nodo suddividendo a metà il numero di osservazioni dati in input, si effettua questo processo finché il numero di osservazioni in un nodo è minore o uguale ad un numero di osservazioni fissato, determinando i nodi foglia.

Una volta che i nodi foglia individuano una assegnazione dei punti ritenuta corretta, questa viene restituita al nodo padre che unirà la soluzione con la soluzione individuata dall'altro nodo figlio.

---

##### Algoritmo 4: Generatore Albero

---

```

input : valori Matrice contenente n osservazioni da k valori
output: Struttura dati contenente n osservazioni da k valori
1 nOss  $\leftarrow$  NumeroOsservazioni(valori)
2 if nOss  $\leq$  dimensioneFoglia then
3   | return OttimizzaFoglia(valori)
4 end
5 return OttimizzaNodo(GeneraAlbero(valori[0 : nOss/2]),
   GeneraAlbero(valori[nOss/2 : nOss]))

```

---

#### 4.1.2 Gestione dei nodi non foglia

Il compito di un nodo non foglia è quello di unire gli assegnamenti individuati nei nodi figli. Questo viene effettuato tramite la permutazione delle colonne di uno delle due matrici, unendo per le colonne, la matrice non permutata con la matrice permutata.

Le colonne della matrice risultante sono valutate ad una ad una tramite il modulo di interpolazione, non appena si individua una colonna dall'errore con livello inferiore ad un livello soglia (riferimento ai livelli dell'algoritmo di selezione di una soluzione in 3.2.6), la si memorizza nella matrice da restituire, togliendo le colonne

che compongono la colonna memorizzata dalla matrice da permutare e dalla matrice non permutata.

Nel caso non si trovi una colonna che rispetti il criterio di selezione, si seleziona la colonna con errore dal livello minore individuata nel corso di tutte le permutazioni (da riga 25 a 32 dell'algoritmo 5).

Si effettuano queste operazioni finché non rimane una sola colonna nella matrice da permutare e nella matrice non permutata.

Il caso migliore è identificato nella individuazione della colonna ottimale nella prima iterazione delle permutazioni, dall'inizio fino ad ogni rimozione di colonna, dove l'algoritmo analizza  $k - 1$  permutazioni,  $k$  rappresenta il numero di colonne. Il caso peggiore è identificato nell'individuazione della colonna ottimale nell'ultima iterazione delle permutazioni dall'inizio fino ad ogni rimozione di colonna, dove l'algoritmo analizza  $\sum_{i=0}^k (k - i)!$  permutazioni. Nel caso si valutasse le colonne di una permutazione complessivamente, quindi senza rimozioni di colonne, sarebbe necessario valutare tutte le permutazioni, l'algoritmo analizzerebbe  $k!$  permutazioni.

Ho scelto di applicare l'operazione di rimozione per i vantaggi portati dal suo caso migliore e dal fatto che il caso peggiore non abbia un aumento così drastico delle permutazioni da analizzare.

### 4.1.3 Gestione dei nodi foglia

I nodi foglia sono caratterizzati da un numero di osservazioni minore o uguale ad un determinato valore (riga 2, algoritmo 4). Questo valore determina il numero di nodi foglia che verranno generati, per esempio con foglie di dimensioni pari a 10 osservazioni su un totale di 20, si hanno  $\frac{20}{10} = 2$  foglie, oppure con foglie di dimensione pari a 5, si hanno  $\frac{20}{5} = 4$  foglie. Per ridurre il numero di permutazioni valutate, l'obiettivo è quello di generare quante più foglie possibili. La riduzione del numero di permutazioni valutate è dimostrata dalla seguente espressione:

$$\text{lavori in corso} \tag{11}$$

Nella determinazione del numero di osservazioni massimo per una foglia è necessario tenere a mente anche il modulo di interpolazione: non è fattibile o è di difficile esecuzione l'interpolazione di una sinusoide a partire da pochi punti, come un solo punto o due.

---

**Algoritmo 5:** Gestione nodo

---

```

input : val1 Matrice contenente  $j$  osservazioni da  $k$  valori
        val2 Matrice contenente  $j$  osservazioni da  $k$  valori
output: Struttura dati contenente  $2j$  osservazioni da  $k$  valori
1 permutazioni  $\leftarrow$  GeneraPermutazioni(val2)
2 colonnaMigliore  $\leftarrow$  [ ]
3 while True do
4   for perm in permutazioni do
5     unione  $\leftarrow$  Unisci(val1, perm)
6     for col in unione do
7       senoide  $\leftarrow$  IndividuaSenoide(col)
8       if LivelloDi(senoide)  $\leq$  livelloSoglia then
9         rimuovi le colonne di val1 e val2 che compongono col
10        rimuovi col da unione
11        permutazioni  $\leftarrow$  GeneraPermutazioni(val2)
12        aggiungi col ad assegnamentoOttimale
13        if NumeroColonne(unione) = 1 then
14          rimuovi le colonne di val1 e val2 che compongono la colonna
15          rimasta in unione
16          aggiungi la colonna rimasta in unione ad
17          assegnamentoOttimale
18          return assegnamentoOttimale
19        end
20        break
21      end
22      if LivelloDi(sin)  $\leq$  LivelloDi(colonnaMigliore) then
23        | colonnaMigliore  $\leftarrow$  col
24      end
25    end
26    rimuovi le colonne di val1 e val2 che compongono colonnaMigliore
27    permutazioni  $\leftarrow$  GeneraPermutazioni(val2)
28    aggiungi col ad assegnamentoOttimale
29    if NumeroColonne(val2) = 1 then
30      | rimuovi le colonne di val1 e val2 rimaste
31      | aggiungi colonnaMigliore ad assegnamentoOttimale
32      | return assegnamentoOttimale
33    end
34  end

```

---



# Bibliografia

- [1] Steven G. Johnson, The NLOpt nonlinear-optimization package
- [2] M. J. D. Powell, "A direct search optimization method that models the objective and constraint functions by linear interpolation," in *Advances in Optimization and Numerical Analysis*, eds. S. Gomez and J.-P. Hennart (Kluwer Academic: Dordrecht, 1994), da pagina 51 a 67.
- [3] Dr. David R. Williams, Planetary Fact Sheets, NASA Goddard Space Flight Center 2016
- [4] Sartoretti P. and Schneide, J. "On the detection of satellites of extrasolar planets with the method of transits" 1999, da pagina 553 a 560, SAO/NASA Astrophysics Data System.