Università Campus Bio-Medico di Roma

Esercitazione III - Meccanica Applicata alle Macchine e Macchine

"Analisi di un Quadrilatero Articolato"

Francesco Fuggitti

1 dicembre 2015

Indice

1 Introduzione

Lo scopo di questa esercitazione è quella di eseguire l'analisi di un quadrilatero articolato di tipo crank-rocker, trovando la coppia motrice C_m che lo mantiene in equilibrio e i moduli delle forze interne, tutto in funzione di ϑ_1, angolo di manovella. Tale analisi del meccanismo è stata condotta attraverso le seguenti fasi:

- Analisi Cinematica
- Analisi Statica
- Implementazione del metodo risolutivo al calcolatore
- Verifica del metodo proposto attraverso il PLV

2 Analisi Cinematica

L'analisi cinematica del quadrilatero articolato è stata sviluppata attraverso due metodi equivalenti che hanno, però, approccio matematico diverso: il metodo analitico delle *equazioni di chiusura* e quello di *Newton-Raphson*. Tali metodi consentono di calcolare facilmente le posizioni assunte dal quadrilatero e, di conseguenza, le sue velocità ed accelerazioni. La nostra analisi li presenterà entrambi, ma si limiterà, per esigenza di esposizione, alla sola determinazione delle posizioni.

2.1 Equazioni di Chiusura

Il primo passo è scrivere l'equazione *vettoriale* di chiusura per il nostro quadrilatero in Figura 2.1.

$$\vec{r}_1 + \vec{r}_2 + \vec{r}_3 + \vec{r}_4 = 0 \tag{2.1}$$

Ma un vettore può anche essere rappresentato in notazione complessa attraverso il suo modulo 'r' e la sua anomalia 'ϑ': $re^{i\vartheta}$. Introducendo un

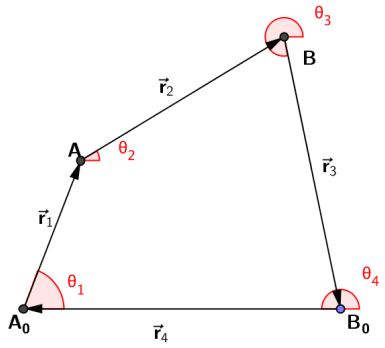

Figura 2.1: Quadrilatero articolato manovella-bilanciere

sistema di coordinate cartesiane e, definiti gli angoli ϑ_1, ϑ_2 e ϑ_3, possiamo scrivere:

$$r_1 e^{i\vartheta_1} + r_2 e^{i\vartheta_2} + r_3 e^{i\vartheta_3} + r_4 e^{i\vartheta_4} = 0 \tag{2.2}$$

Successivamente, applicando alla (2.2) la relazione di Eulero: $re^{i\vartheta} = r(\cos\vartheta + i\sin\vartheta)$, assumendo come ipotesi semplificativa $\vartheta_4 = \pi$ e, separando la parte reale da quella immaginaria, si ottiene il seguente sistema:

$$\begin{cases} r_1\cos\vartheta_1 + r_2\cos\vartheta_2 + r_3\cos\vartheta_3 - r_4 = 0 \\ r_1\sin\vartheta_1 + r_2\sin\vartheta_2 + r_3\sin\vartheta_3 = 0 \end{cases} \tag{2.3}$$

Note le lunghezze delle aste del quadrilatero ed assegnata la posizione ϑ_1, le uniche incognite del nostro problema rimangono ϑ_2 e ϑ_3. Esse, però, figurano come argomenti di funzioni trigonometriche e, quindi, il sistema è di tipo *non lineare*. La procedura di risoluzione, dunque, discende dall'analisi numerica. Esplicitiamo le due relazioni del sistema (2.3) come segue:

$$r_2\cos\vartheta_2 = r_4 - r_3\cos\vartheta_3 - r_1\cos\vartheta_1 \tag{2.4a}$$

$$r_2\sin\vartheta_2 = -(r_1\sin\vartheta_1 + r_3\sin\vartheta_3) \tag{2.4b}$$

Quadrando e sommando la (2.4a) con la (2.4b) si elimina ϑ_2 e si ottiene una sola equazione in ϑ_3:

$$A \sin \vartheta_3 + B \cos \vartheta_3 + C = 0 \tag{2.5}$$

dove:

$$A = 2r_1 r_3 \sin \vartheta_1 \tag{2.6a}$$
$$B = 2r_1 r_3 \cos \vartheta_1 - 2r_3 r_4 \tag{2.6b}$$
$$C = r_1^2 + r_3^2 + r_4^2 - r_2^2 - 2r_1 r_4 \cos \vartheta_1. \tag{2.6c}$$

Per risolvere la (2.5) effettuiamo la sostituzione $t = \tan \frac{\vartheta_3}{2}$ dalla quale abbiamo $\sin \vartheta_3 = \frac{2t}{1+t^2}$ e $\cos \vartheta_3 = \frac{1-t^2}{1+t^2}$. Con tale sostituzione, si arriva ad un' equazione *lineare* di secondo grado in t facilmente risolvibile:

$$t^2(C - B) + 2At + B + C = 0 \tag{2.7}$$

La soluzione dell'equazione (2.7) potrà generare radici t_1, t_2:

- reali $\Rightarrow$ il meccanismo occupa una posizione definita.
- coincidenti $\Rightarrow$ il meccanismo è in posizione di arresto.
- complesse e coniugate $\Rightarrow$ il meccanismo supera la posizione di arresto.

In particolare, per $t_1, t_2 \in \mathbb{R}$ avremo rispettivamente $\vartheta_3^{(1)} = 2 \arctan(t_1)$ e $\vartheta_3^{(2)} = 2 \arctan(t_2)$ a ciascuno dei quali corrispondono 2 angoli uguali a meno di π. Tra questi ultimi 2 valori la scelta è completamente arbitraria. Altrettanto non si può dire, invece, per la scelta tra $\vartheta_3^{(1)}$ e $\vartheta_3^{(2)}$. Tali valori corrispondono, difatti, a 2 configurazioni equivalenti del quadrilatero. Perciò, se ne sceglie una all'inizio e si rimane congruenti con essa per tutta l'analisi. Una volta ricavato il valore di ϑ_3, possiamo risalire al valore di ϑ_2 calcolando i valori di (2.4a), (2.4b) e poi l'arcotangente del loro rapporto calcolata tra $-\pi$ e π.

2.2 Newton-Raphson

Il metodo di Newton-Raphson qui discusso è una valida alternativa al metodo delle equazioni di chiusura. Esso si basa su concetti molto semplici e consente

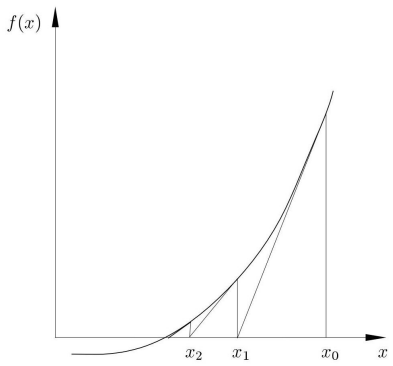

Figura 2.2: Interpretazione grafica del metodo Newton-Raphson

di pervenire rapidamente, a partire da una soluzione di primo tentativo - *initial guess* - ad una soluzione approssimata. Vediamo di seguito la sua applicazione.

Partendo dal caso scalare, supponiamo di avere una funzione $y = f(x)$ non lineare, continua e di classe C^∞ come in Figura 2.2.

Lo scopo è quello di determinare lo zero di questa funzione o, analogamente, risolvere $f(x) = 0$. Come soluzione si assume un valore di primo tentativo x_0 e si valuta la funzione in quel punto. In generale, risulterà $f(x_0) \neq 0$ e a questa quantità si dà il nome di resto r_0, quindi avremo che $(f(x_0) \equiv r_0)$. Sulla base di questo primo tentativo, si sceglie un nuovo valore approssimante x_1. Per sceglierlo opportunamente si approssima la funzione con il suo sviluppo in serie di Taylor nell'intorno del punto x_0, arrestato al primo ordine:

$$f(x) \simeq f(x_0) + f'(x_0)(x - x_0) = r_0 + f'(x_0)(x - x_0) \tag{2.8}$$

Di conseguenza, la nostra x_1 si troverà dall'intersezione tra l'asse della ascisse e la retta di equazione (2.8).

$$r_0 + f'(x_0)(x_1 - x_0) = 0 \qquad \Rightarrow \qquad x_1 = x_0 - \frac{r_0}{f'(x_0)} \tag{2.9}$$

Procedendo iterativamente, si ottiene la formula ricorsiva:

$$x_{i+1} = x_i - \frac{r_i}{f'(x_i)} \qquad \text{dove:} \qquad r_i = f(x_i) \tag{2.10}$$

Il processo di calcolo viene arrestato dopo k iterazioni, quando sia il modulo del resto che il modulo della differenza tra x_{k+1} e x_k diventano minori di una certa tolleranza desiderata ε, che rappresenta la precisione con la quale si vuole calcolare la soluzione. L'iterazione termina anche quando le k iterazioni superano un valore di soglia massimo K_{max}. Queste condizioni possono essere sintetizzate come segue:

$$\begin{cases} |r_k| & \leq \varepsilon \\ |x_k - x_{k+1}| & \leq \varepsilon \\ k & \geq K_{max} \end{cases}$$

Il procedimento appena visto si può facilmente estendere al caso generale (con grandezze vettoriali), ossia applicandolo ad un sistema di equazioni non lineari in più variabili (come nel caso del quadrilatero articolato che trattiamo). Dato, quindi, un sistema di n equazioni $\underline{\mathbf{f}}(\underline{\mathbf{x}}) = \underline{\mathbf{0}}$ e, fatte le stesse considerazioni per il caso scalare, si ottiene la seguente relazione in forma vettoriale:

$$\{x_{i+1}\} = \{x_i\} - [J_i]^{-1}\{R_i\} \tag{2.11}$$

dove

- $[J_i]$ è la matrice Jacobiana di $\underline{\mathbf{f}}$ valutata in $\underline{\mathbf{x_i}}$
- $\{R_i\}$ è il vettore dei resti

In modo del tutto analogo al caso scalare, l'iterazione termina quando sia i moduli di tutte le componenti di $\{R_k\}$, sia i moduli delle componenti del vettore $\{x_k - x_{k+1}\}$, sono minori di una certa tolleranza ε.

L'applicazione di questo metodo per il quadrilatero in esame è facilmente intuibile. Preso il sistema di equazioni (2.3) e, fissato un valore di ϑ_1, si dovrà scegliere il vettore $\{x_0\}$ di *initial guess* contenente le soluzioni di primo tentativo per ϑ_2 e ϑ_3. Tale scelta è fondamentale poichè determina la convergenza o meno alla soluzione. Una volta terminata l'iterazione sarà conclusa l'analisi delle posizioni del meccanismo.

3 Analisi Statica

L'analisi statica del quadrilatero in Figura 3.1 si articola nelle seguenti fasi:

- Diagramma del *free-body*
- Equazioni di equilibrio
- Calcolo delle reazioni vincolari e della coppia C_m che garantisce l'equilibrio della struttura

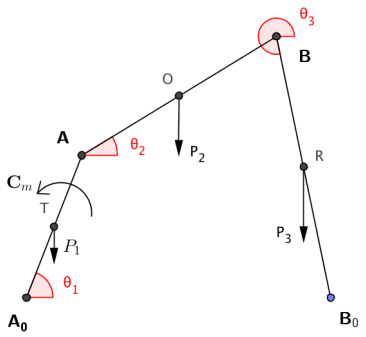

Figura 3.1: Quadrilatero articolato preso in esame

Attraverso tali fasi di studio, si possono facilmente calcolare le azioni delle forze interne e la coppia motrice C_m che mantiene in equilibrio il meccanismo.

Come già detto, il primo passo per eseguire l'analisi è quello di imporre le equazioni cardinali della statica per ogni membro del corpo. Infatti, il diagramma del *free-body* consiste nell'isolare ogni membro, eliminare le coppie cinematiche che lo collegano ad altri membri e sostituire a queste ultime le azioni dinamiche di reazione che i membri si scambiano tramite le coppie medesime. Nella Figura 3.2 viene rappresentato il free-body per il quadrilatero in esame.

Indicheremo con

- F_{ij_x} e F_{ij_y} le componenti cartesiane della reazione esercitata dal membro i-esimo sul j-esimo;
- P_i la forza peso applicata nella mezzeria del corpo i-esimo;

- C_m la coppia motrice.

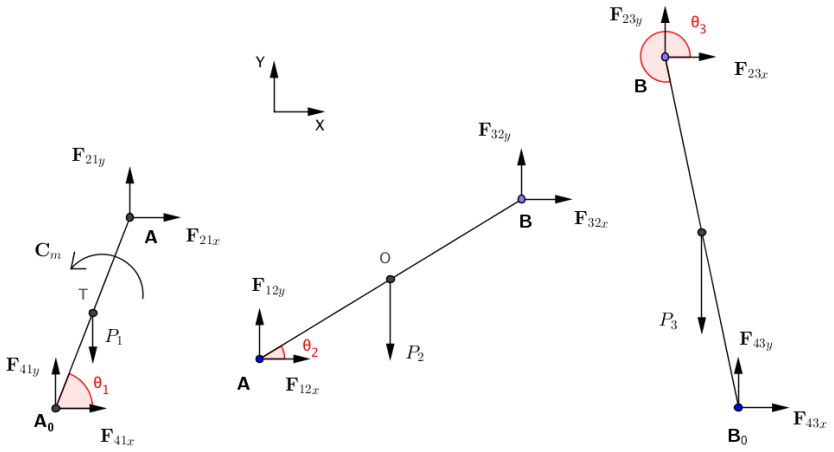

Figura 3.2: Driagramma del *free-body* per il quadrilatero articolato

Nel nostro caso possiamo distinguere 3 corpi, ciascuno dei quali viene equilibrato come segue:

1. Equilibrio manovella (corpo I)

$$\begin{aligned}
&\Sigma_x : F_{41_x} - F_{12_x} = 0 \\
&\Sigma_y : F_{41_y} - F_{12_y} - P_1 = 0 \\
&\Sigma_{M_{A_0}} : C_m - P_1 \frac{r_1}{2} \cos\vartheta_1 + F_{12_x} r_1 \sin\vartheta_1 - F_{12_y} r_1 \cos\vartheta_1 = 0
\end{aligned}$$

2. Equilibrio biella (corpo II)

$$\begin{aligned}
&\Sigma_x : F_{12_x} + F_{32_x} = 0 \\
&\Sigma_y : F_{12_y} + F_{32_y} - P_2 = 0 \\
&\Sigma_{M_A} : -P_2 \frac{r_2}{2} \cos\vartheta_2 + F_{32_y} r_2 \cos\vartheta_2 - F_{32_x} r_2 \sin\vartheta_2 = 0
\end{aligned}$$

3. Equilibrio bilanciere (corpo III)

$$\begin{aligned}
&\Sigma_x : F_{43_x} - F_{32_x} = 0 \\
&\Sigma_y : F_{43_y} - F_{32_y} - P_3 = 0 \\
&\Sigma_{M_{B_0}} : +P_3 \frac{r_3}{2} \cos\vartheta_3 + F_{32_y} r_3 \cos\vartheta_3 - F_{32_x} r_3 \sin\vartheta_3 = 0
\end{aligned}$$

Unendo queste equazioni, si ottiene un sistema che può essere riscritto in forma matriciale come segue:

$$\begin{bmatrix} 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & r_1 \sin\vartheta_1 & -r_1 \cos\vartheta_1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -r_2 \sin\vartheta_2 & r_2 \cos\vartheta_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -r_3 \sin\vartheta_3 & r_3 \cos\vartheta_3 & 0 & 0 \end{bmatrix} \begin{bmatrix} F_{41_x} \\ F_{41_y} \\ F_{12_x} \\ F_{12_y} \\ C_m \\ F_{32_x} \\ F_{32_y} \\ F_{43_x} \\ F_{43_y} \end{bmatrix} = \begin{bmatrix} 0 \\ P_1 \\ P_1 \frac{r_1}{2} \cos\vartheta_1 \\ 0 \\ P_2 \\ P_2 \frac{r_2}{2} \cos\vartheta_2 \\ 0 \\ p_3 \\ -P_3 \frac{r_3}{2} \cos\vartheta_3 \end{bmatrix}$$

Ossia, in forma compatta:

$$AX = b \tag{3.1}$$

che ha soluzione:

$$X = A^{-1}b \tag{3.2}$$

Questa espressione ci dà, per ogni singolo valore di ϑ_1 e, quindi, di ϑ_2 e ϑ_3, i valori delle reazioni vincolari e il valore della coppia C_m che mantiene in equilibrio il quadrilatero.

4 Implementazione metodo risolutivo in ambiente MATLAB®

L'analisi cinematica (equazioni di chiusura e Newton-Raphson) e l'analisi statica sono state interamente implementate su MATLAB®. Per farlo, si è voluto cercare un approccio risolutivo di tipo modulare e parametrico. La suddivisione in moduli ha permesso di scrivere un codice semplice, chiaro e facilmente leggibile. Inoltre, per una più facile comprensione, viene riportato nella tabella un confronto tra i valori usati nella trattazione e quelli in MATLAB®.

Trattazione	Matlab	Unità di Misura
r_1, r_2, r_3, r_4	r_1, r_2, r_3, r_4	m
P_1, P_2, P_3	$P1, P2, P3$	N
$\vartheta_1, \vartheta_2, \vartheta_3$	$T(:,1), T(:,2), T(:,3)$	rad

Sono stati implementati 2 programmi: il primo prevede l'analisi completa con l'approccio delle equazioni di chiusura, il secondo, invece, soltanto l'analisi cinematica col metodo di Newton-Raphson.

4.1 Programma 1

Il Programma 1 è suddiviso in tre moduli. Il primo è il main, che, di fatto, è il corpo principale del programma. Gli altri due sono *function* (kinematics1 e static1) che, "chiamati" dal main, eseguono rispettivamente l'analisi cinematica e quella statica.

```
1  %% KINEMATICS & STATIC ANALYSIS PROGRAM - MAIN
2
3  clear all; clc;
4
5  r_1=input ('inserisci r1 (m): '); % input lung. manovella
6  r_2=input ('inserisci r2 (m): '); % input lung. biella
7  r_3=input ('inserisci r3 (m): '); % input lung. bilanciere
8  r_4=input ('inserisci r4 (m): '); % input lung. telaio
9
10 if r_1+r_4≤r_2+r_3 && r_1<r_4
11
12     % stampa a schermo se il controllo e' andato a buon fine
13     disp('il quadrilatero   di Grashof! ');
14
15     % chiede il passo da tenere nel discretizzare theta1
16     gait = input('inserisci il passo: ');
17
18     % calcola il numero di passi totali eseguiti
19     max_len = numel(0:gait:2*pi);
20
21     % esegue l'analisi cinematica
22     [T] = kinematics1 (r_1,r_2,r_3,r_4,max_len,gait);
```

```
23
24      % chiede i moduli delle forze peso di ogni asta
25      P1 = input ('inserisci P1 (N): ');
26      P2 = input ('inserisci P2 (N): ');
27      P3 = input ('inserisci P3 (N): ');
28
29      % esegue analisi statica
30      [Pl_1,Pl_2,Pl_3,Pl_4,Pl_5, Q] = static1 ...
            (r_1,r_2,r_3,P1,P2,P3,T,max_len);
31
32  else
33      % errore se il quadrilatero non rispetta Grashof
34       error ('il quadrilatero immesso non   di Grashof! ...
            Ritenta');
35  end
```

```
1  function [T] = kinematics1 (r_1,r_2,r_3,r_4,max_len,gait)
2
3  % KINEMATICS ANALYSIS FOURBAR LINKAGE (CRANK-ROCKER) - ...
       KINEMATICS1
4  % Questa function di  Matlab ha lo scopo di risolvere le ...
       equazioni
5  % di chiusura di un quadrilatero articolato di tipo ...
       manovella-bilanciere.
6
7  % INPUT:
8  % r_1: lunghezza manovella
9  % r_2: lunghezza biella
10 % r_3: lunghezza bilanciere
11 % r_4: lunghezza telaio
12 % max_len: numero passi totali di discretizzazione
13 % gait: passo di discretizzazione angolo
14
15 % OUTPUT :
16 % T: matrice gaitx3 delle soluzioni theta1, theta2, theta3
17
18 %definisce il primo vettore colonna della matrice T con ...
       tutti i valori di
19 %theta1 da 0 a 2*pi discretizzato di passo 'gait'
```

```
20 T(:,1) = 0:gait:2*pi;
21
22 %esegue un ciclo for per 'max_len' volte andando a calcolare,
23 %per ogni iterazione, gli angoli theta3 e theta2 ...
      corrispondenti
24 %all'i-esimo theta1
25 for i=1:max_len
26
27     %coefficienti dell'equazione di secodo grado ridotta a ...
          theta3
28     A = 2*r_1*r_3*sin(T(i,1));
29     B = 2*r_1*r_3*cos(T(i,1)) - 2*r_3*r_4;
30     C = (r_1)^2 + (r_3)^2 + (r_4)^2 - (r_2)^2 - ...
          2*r_1*r_4*cos(T(i,1));
31
32     %definisce un vettore con i coefficienti ...
          dell'equazione di secondo
33     %grado in t per poi calcolarne le radici con la ...
          funzione roots.
34     p=[(C-B) 2*A (B+C)];
35
36     %Tali radici vengono assegnate ad un vettore colonna Y
37     Y = roots(p);
38
39     %controlla se le radici sono state assegnate in modo ...
          ordinato
40     %nel vettore Y per poi scegliere la soluzione ...
          congruente con quella
41     %scelta precedentemente.
42     if issorted(Y)
43         T(i,3) = 2*atan(Y(1));
44     else
45         T(i,3) = 2*atan(Y(2));
46     end
47
48     %rende positivo l'angolo theta3 aggiungendo 2*pi
49     T(i,3)=T(i,3)+2*pi;
50
51     %calcola rispettivamente r_2*cos(theta2) e ...
          r_2*sin(theta2).
52     x = r_4 - r_3*cos(T(i,3)) - r_1*cos(T(i));
```

```
53      y = -(r_1*sin(T(i)) + r_3*sin(T(i,3)));
54
55      %calcola theta2 tramite la funzione di matlab atan2 ...
            che calcola
56      %l'arcotangente del rapporto tra y ed x tenendo conto ...
            del loro
57      %segno.
58      T(i,2) = atan2(y,x);
59
60  end
61
62  end
```

```
1   function [Pl_1,Pl_2,Pl_3,Pl_4,Pl_5, Q] = static1 ...
        (r_1,r_2,r_3,P1,P2,P3,T,max_len)
2
3   % STATIC ANALYSIS FOURBAR LINKAGE (CRANK-ROCKER)
4   % Questa function Matlab ha lo scopo di risolvere le ...
        equazioni di
5   % equilibrio di un quadrilatero articolato di tipo ...
        manovella-bilanciere
6   % in cui sono note le dimensioni delle aste e le relative ...
        forze peso
7   % applicate nella mezzeria delle aste.
8
9   %INPUT
10  % r_1: lunghezza manovella
11  % r_2: lunghezza biella
12  % r_3: lunghezza bilanciere
13  % P1: forza peso della manovella applicata nella mezzeria
14  % P2: forza peso della biella applicata nella mezzeria
15  % P3: forza peso del bilanciere applicata nella mezzeria
16  % T: matrice gaitx3 delle soluzioni theta1, theta2, theta3
17  % max_len: numero passi totali di discretizzazione
18  % gait: passo di discretizzazione angolo
19
20  %OUTPUT
21  % Pl_i: plot delle soluzioni in funzione di theta1
22  % Q: matrice gaitx2 che restituisce i valori di theta1 e ...
```

```
    di Cm
23
24 %esegue un ciclo for per 'max_len' volte andando a calcolare,
25 %per ogni iterazione, le reazioni vincolari e la coppia C_m
26 %corrispondenti all'i-esimo theta1, e ai corrispondenti ...
    theta2 e theta3.
27 for i=1:max_len
28
29     %definisce la matrice E costruita a partire dalle ...
        equazioni di
30     %Equilibrio.
31     E = [ 1 0 -1 0 0 0 0 0 0
32         0 1 0 -1 0 0 0 0 0
33         0 0 r_1*sin(T(i,1)) -r_1*cos(T(i,1)) 1 0 0 0 0
34         0 0 1 0 0 1 0 0 0
35         0 0 0 1 0 0 1 0 0
36         0 0 0 0 0 -r_2*sin(T(i,2)) r_2*cos(T(i,2)) 0 0
37         0 0 0 0 0 -1 0 1 0
38         0 0 0 0 0 0 -1 0 1
39         0 0 0 0 0 -r_3*sin(T(i,3)) r_3*cos(T(i,3)) 0 0];
40
41     %definisce il vettore di termini noti b costruito a ...
        partire
42     %dalle equazioni di Equilibrio.
43     b=[0; P1; P1*((r_1)/2)*cos(T(i,1)); 0; P2; ...
        P2*((r_2)/2)*cos(T(i,2)); 0; P3; ...
        -P3*((r_3)/2)*cos(T(i,3))];
44
45     %effettua un controllo sul det(E) assicurandosi che la ...
        matrice
46     %non sia singolare.
47     if det(E)≠0
48
49         %si calcola il vettore della incognite X
50         X = E\b;
51
52         %in base alla disposizione della incognite nel ...
            vettore delle
53         %incognite X=[f41x; f41y; f12x; f12y; Cm; f32x; ...
            f32y; f43x;
54         %f43y]; si calcolano i moduli delle azioni ...
```

```
            dinamiche interne e
55          %il modulo della coppia motrice Cm che mantiene in
56          %equilibrio il meccanismo.
57
58          F41(i,1) = sqrt( (X(1))^2 + (X(2))^2 );
59          F12(i,1) = sqrt( (X(3))^2 + (X(4))^2 );
60          Cm(i,1) = X(5);
61          F32(i,1) = sqrt( (X(6))^2 + (X(7))^2 );
62          F43(i,1) = sqrt( (X(8))^2 + (X(9))^2 );
63
64      end
65  end
66
67  %costruisce la matrice Q
68  Q(:,1)=T(:,1);
69  Q(:,2)=Cm;
70
71  % Plot dei Risultati
72
73  %crea la finestra per graficare Cm
74  Pl_1=figure('Position',[200 100 700 500]);
75  plot(T(:,1),Cm,'LineWidth',2);
76  xlabel('\theta_1 [rad]');
77  ylabel('Coppia C_m');
78
79  %crea le finestre per graficare i moduli delle forze
80  Pl_2=figure('Position',[200 100 700 500]);
81  grid on;
82  plot (T(:,1),F41,'LineWidth',2);
83  xlabel('\theta_1 [rad]');
84  ylabel('F_{41}');
85  Pl_3=figure('Position',[200 100 700 500]);
86  grid on;
87  plot (T(:,1),F12,'LineWidth',2);
88  xlabel('\theta_1 [rad]');
89  ylabel('F_{12}');
90  Pl_4=figure('Position',[200 100 700 500]);
91  grid on;
92  plot (T(:,1),F32,'LineWidth',2);
93  xlabel('\theta_1 [rad]');
94  ylabel('F_{32}');
```

```
95  Pl_5=figure('Position',[200 100 700 500]);
96  grid on;
97  plot (T(:,1),F43,'LineWidth',2);
98  xlabel('\theta_1 (rad)');
99  ylabel('F_{43}');
100
101 end
```

4.2 Programma 2

Il Programma 2 è suddiviso in 4 moduli e presenta solamente l'implementazione del metodo di Newton-Raphson. Come per il programma 1, il modulo principale è il main. Gli altri moduli sono le *function* newtonraphson, jacob e func. Essi svolgono rispettivamente i seguenti compiti: algoritmo N-R, valutazione della matrice Jacobiana e valutazione del vettore resto.

```
1  %% NEWTON-RAPHSON METHOD - MAIN
2  clc; clear; close all;
3
4  % Soluzione primo tentativo
5  x_0=[1.49 ; 5.24];
6
7  % Tolleranza consentita
8  toll=1e-6;
9
10 % Massimo numero di iterazioni
11 itmax=100;
12
13 %Vettore colonna valori theta1
14 TH(:,1)=0:0.01*pi:2*pi;
15
16 for i=1:numel(TH)
17 % Metodo di Newton-Raphson per trovare la soluzione del ...
       sistema di
18 % equazioni non lineari
19 [x]=newtonraphson(@func,@jacob,x_0,toll,itmax,TH(i,1));
20
21 TH(i,2)=x(1); %assegno theta2 alla seconda colonna
```

```
22 TH(i,3)=x(2); %assegno theta3 alla terza colonna
23 end
```

```
1  function [x]=newtonraphson(func,jacob,x_0,toll,itmax,TH)
2  %
3  % NEWTON-RAPHSON ALGORITHM
4  % INPUT
5  % func : Function in cui viene valutato il vettore dei resti
6  % jacob : Function in cui viene valutata la Jacobiana
7  % x_0: Vettore soluzione di primo tentativo
8  % toll: Tolleranza
9  % itmax: Numero massimo di iterazioni
10 % TH: Valore di theta1
11 %
12 % OUTPUT
13 % x: Soluzione
14
15 % Inizializza le variabili
16
17 resid=1000;
18 iterations=0;
19
20 while resid>toll || iterations>itmax
21
22     % Calcola matrice Jacobiana
23     J=feval(jacob,x_0);
24     % Calcola residui
25     f=feval(func,x_0,TH);
26     % Calcola norma vettore dei residui
27     resid=norm(f);
28
29     % Trova il nuovo valore di x con il metodo di ...
           Newton-Raphson
30     x = x_0-J\f;
31
32     % Assegna il nuovo valore come condizioni iniziali per ...
           la successiva
33     % iterazione
34     x_0=x;
```

```
35
36      iterations=iterations+1;
37  end
38
39  end
```

```
1   function J = jacob(x)
2
3   % OUTPUT:
4   % J: matrice Jacobiana valutata nel punto x
5   %
6   % INPUT:
7   % x: vettore delle soluzioni
8
9   r(2)=3;
10  r(3)=3.5;
11
12  J(1,1) = -r(2)*sin( x(1));
13  J(1,2) = -r(3)*sin( x(2));
14  J(2,1) =  r(2)*cos( x(1));
15  J(2,2) =  r(3)*cos( x(2));
16
17  end
```

```
1   function f=func(x,TH)
2
3   % OUTPUT:
4   % f: vettore dei resti
5   %
6   % INPUT:
7   % x: vettore delle soluzioni
8   % TH: valore di theta1
9
10  r(1)=2;
11  r(2)=3;
12  r(3)=3.5;
13  r(4)=4;
14
```

```
15 theta(1)=TH;
16 f(1)= r(1)*cos(theta(1)) + r(2)*cos( x(1)) + r(3)*cos( ...
       x(2)) - r(4);
17 f(2)= r(1)*sin(theta(1)) + r(2)*sin( x(1)) + r(3)*sin( x(2));
18
19 f= f';
20 end
```

4.3 Risultati grafici della Simulazione

I risultati sono ottenuti imponendo come dati di input $r_1 = 2$, $r_2 = 3$, $r_3 = 3,5$, $r_4 = 4$, $P_1 = 20$, $P_2 = 30$ e $P_3 = 35$ con passo $0,01\pi$.

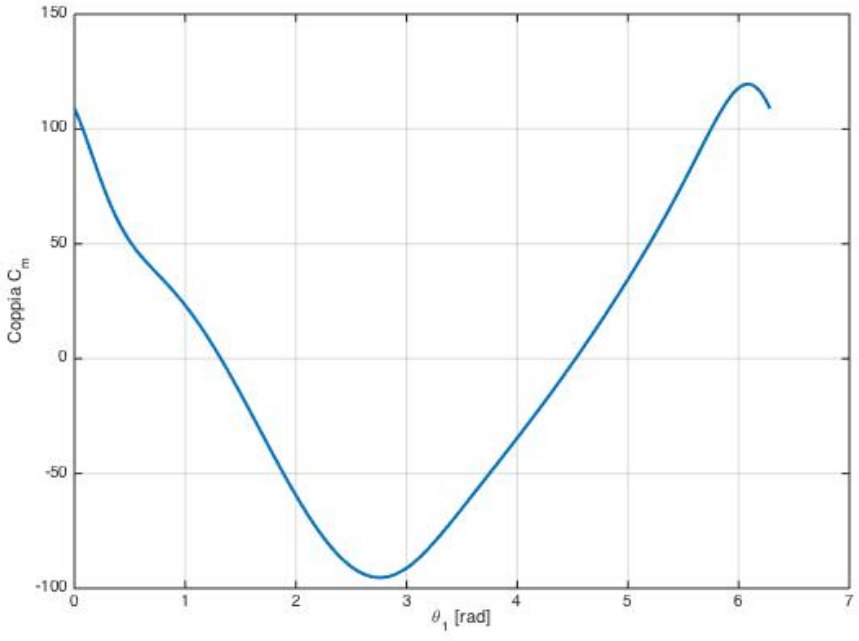

Figura 4.1: Andamento della coppia motrice C_m in funzione di ϑ_1

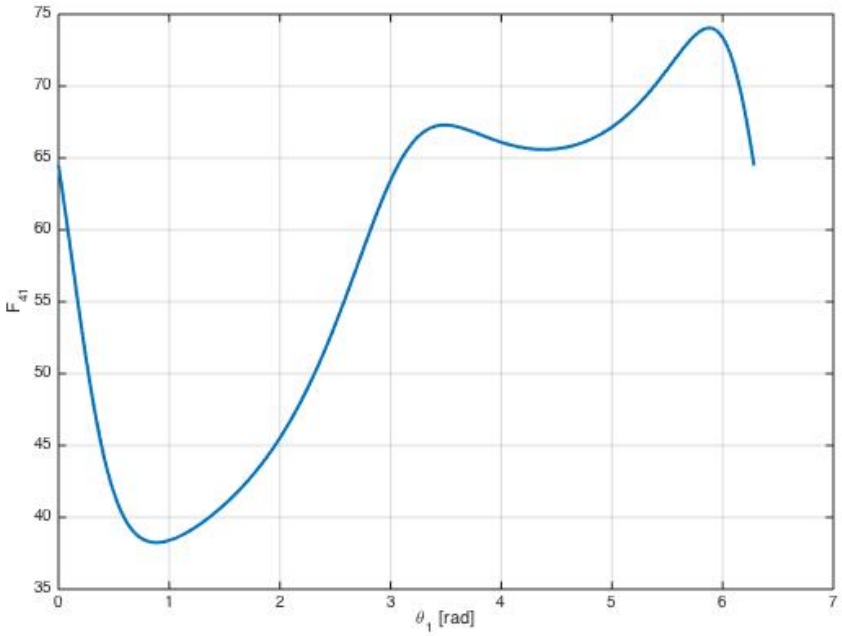

Figura 4.2: Andamento della forza F_{41} in funzione di ϑ_1

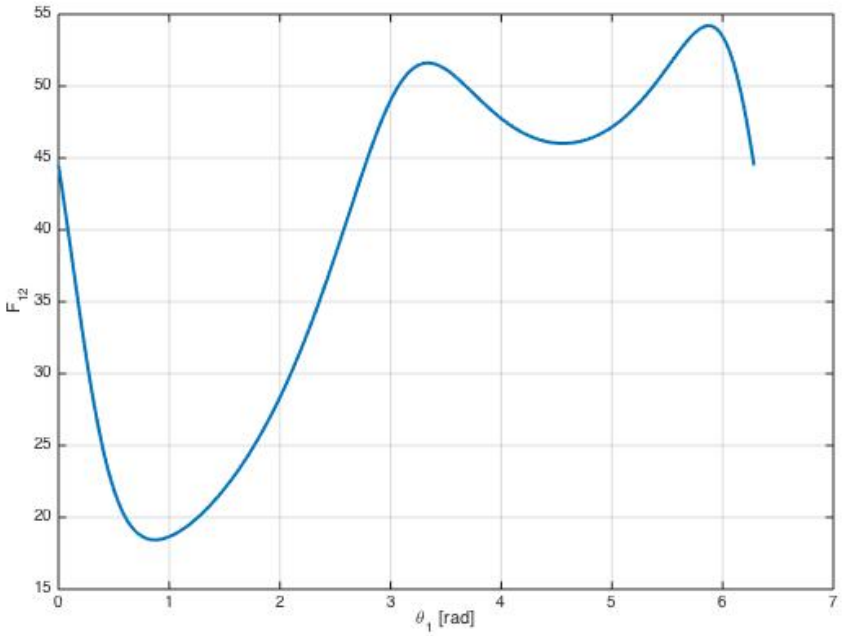

Figura 4.3: Andamento della forza F_{12} in funzione di ϑ_1

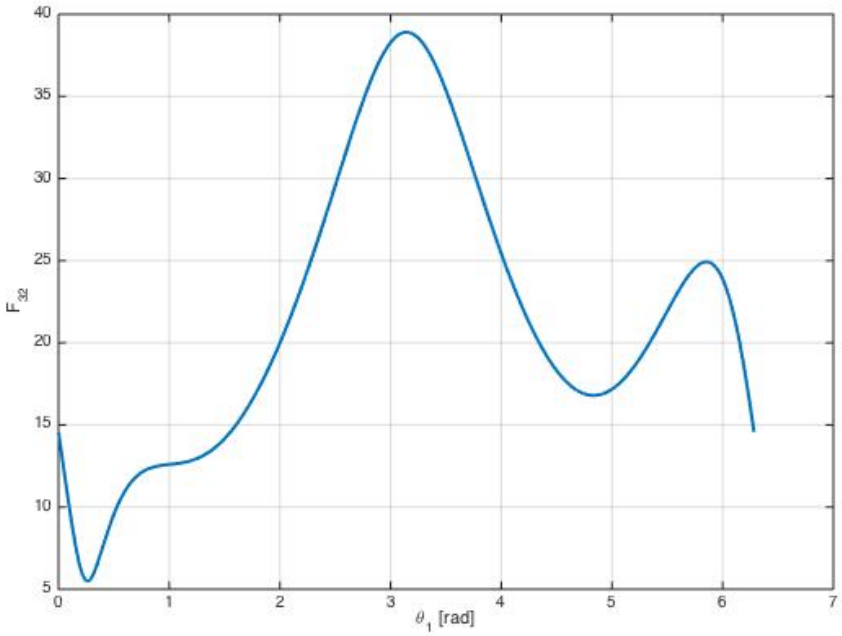

Figura 4.4: Andamento della forza F_{32} in funzione di ϑ_1

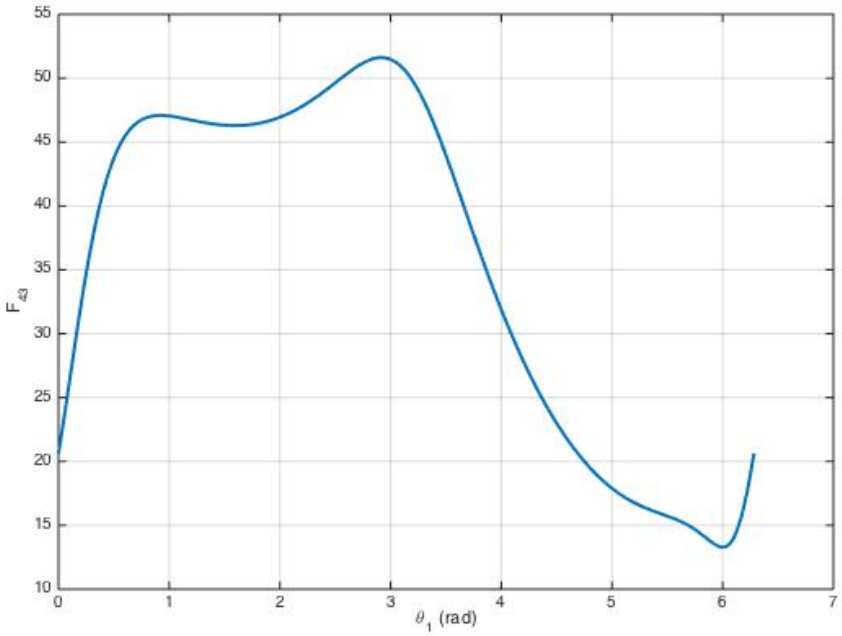

Figura 4.5: Andamento della forza F_{43} in funzione di ϑ_1

4.4 Confronto risultati numerici

Operando la simulazione con ciascuno dei due Programmi, si perviene ad un risultato intermedio uguale, come è possibile riscontrare dai rispettivi output riportati, esemplificativamente in parte, nelle seguenti tabelle.

Pgr. 1 - Eq. di Chiusura			Pgr. 2 - Newton-Raphson		
ϑ_1	ϑ_2	ϑ_3	ϑ_1	ϑ_2	ϑ_3
0	1,5082	5,2567	0	1,5082	5,2567
0,0314	1,4762	5,2254	0,0314	1,4762	5,2254
0,0628	1,4432	5,1943	0,0628	1,4432	5,1943
0,0942	1,4095	5,1638	0,0942	1,4095	5,1638
0,1256	1,3751	5,1340	0,1256	1,3751	5,1340
0,1570	1,3403	5,1050	0,1570	1,3403	5,1050
...	...	...	...	...	...

5 Verifica dei risultati col PLV

Infine, è stata eseguita la verifica, per la posizione $\vartheta_1 = 1\ rad$ (Figura 5.1) della coppia motrice C_m attraverso il Principio dei Lavori Virtuali. Ipotizzando una velocità angolare ω_{12} virtuale di verso orario impressa alla manovella, si può scrivere l'equazione delle potenze virtuali di tutte le forze esterne:

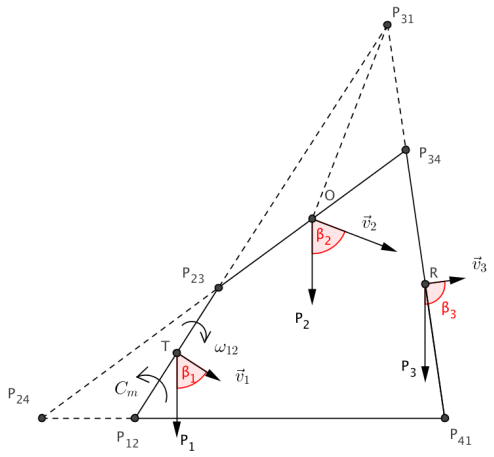

Figura 5.1: Centri d'istantanea rotazione per l'applicazione del PLV

$$\vec{P}_1 \cdot \vec{v}_1 + \vec{P}_2 \cdot \vec{v}_2 + \vec{P}_3 \cdot \vec{v}_3 + \vec{C}_m \cdot \vec{\omega}_{12} = 0 \tag{5.1}$$

dove le uniche incognite sono

- $\vec{v}_1$ è la velocità della manovella applicata nella sua mezzeria
- $\vec{v}_2$ è la velocità della biella applicata nella sua mezzeria
- $\vec{v}_3$ è la velocità del bilanciere applicata nella sua mezzeria

Per esplicitare queste velocità, usiamo il teorema di Aronhold-Kennedy trovando i centri d'istantanea rotazione del quadrilatero, indicati in Figura 5.1 come P_{ij}.

Possiamo, quindi, porre in forma scalare la relazione (5.1):

$$P_1 v_1 \cos\beta_1 + P_2 v_2 \cos\beta_2 + P_3 v_3 \cos\beta_3 - C_m \omega_{12} = 0$$

ovvero

$$P_1\omega_{12}P_{12}T\cos\beta_1 + P_2\frac{P_{12}P_{23}}{P_{31}P_{23}}P_{31}O\omega_{12}\cos\beta_2 + P_3\frac{P_{12}P_{24}}{P_{41}P_{24}}P_{41}R\omega_{12}\cos\beta_3 - C_m\omega_{12} = 0$$

che, semplificando per ω_{12} e arrangiando i termini, diventa:

$$C_m = P_1P_{12}T\cos\beta_1 + P_2\frac{P_{12}P_{23}}{P_{31}P_{23}}P_{31}O\cos\beta_2 + P_3\frac{P_{12}P_{24}}{P_{41}P_{24}}P_{41}R\cos\beta_3 \tag{5.2}$$

Per quantificare la (5.2) ci si è avvalsi di GeoGebra calcolando sia la lunghezza dei segmenti sconosciuti, sia l'ampiezza degli angoli β_1, β_2 e β_3 formati dai vettori velocità con i vettori forza peso.

Grandezze	Valori
$P_{12}T$	1 m
$P_{31}P_{23}$	$4,03$ m
$P_{31}O$	$2,69$ m
$P_{12}P_{24}$	$1,2$ m
$P_{41}P_{24}$	$5,2$ m
$P_{41}R$	$1,75$ m
β_1	1 rad
β_2	$1,2$ rad
β_3	$1,72$ rad

Sostituendo tali valori, si ottiene $C_m = 23,2182\ Nm$. Allo stesso modo, si può vedere come nella simulazione del Programma 1 (cfr. 4.1), inserendo per semplicità passo uguale a $0,01$, la matrice Q di output riporta $C_m = 23,2246\ Nm$, che è approssimativamente uguale a quello ricavato in precedenza attraverso il PLV.