

UNIVERSIDAD NACIONAL DE COLOMBIA

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA MECÁNICA Y MECATRÓNICA

MÉTODOS NUMÉRICOS

GRUPO 2

2021-1

Aplicación del método de Newton Raphson para describir la cinemática de un mecanismo

Autores:

Andrés Holguín R.

Julián A. Caipa P.

Santiago Marín B.

Docente:

Ing. Oswaldo
Rojas Camacho

Índice

1. Objetivos	1
2. Introducción y planteamiento del problema	1
3. Marco teórico	2
3.1. Método de Newton Raphson	2
3.1.1. Método de Newton Raphson en una variable real	2
3.1.2. Extrapolación del método a varias variables	2
4. Perfil cinemático del motor	3
5. Resolución del problema	4
5.1. Cadena cinemática	4
5.1.1. Posición	4
5.1.2. Velocidad	4
5.1.3. Aceleración	5
5.2. Implementación del método de Newton Raphson	5
5.2.1. NR Consecutivo	5
5.2.2. NR Simultaneo	7
5.3. Solución analítica	8
6. Resultados	8
6.1. NR Simultaneo	9
6.2. NR Consecutivo	9
6.3. Simulación	9
7. Análisis de resultados	10
7.1. Comparación de aplicación de NR	10
7.2. Error absoluto del método de Newton Raphson	11
7.3. Comparación de resultados NR-Simulación <i>Inventor</i>	12
8. Conclusiones	13
9. Referencias	13
A. Gráficas de resultados para el método de Newton Raphson consecutivo	14
B. Resultados de simulación de inventor	15
C. Jacobianos Inversos	16
C.1. Posición:	16
C.2. Velocidad:	16
C.3. Aceleración:	17
D. Jacobiano del sistema simultaneo:	18
E. Códigos de Matlab del desarrollo	20
E.1. Perfil cinemático	21

E.2. Solución por Newton Raphson Consecutivo	23
E.3. Solución por Newton Raphson simultaneo	28
E.4. Solución por método de Newton Raphson	31
E.5. Solución analítica	40
E.6. Resultados NR simultaneo vs solución analítica	43
E.7. Resultados NR simultaneo VS simulación	45

1. Objetivos

- Resolver un problema aplicado a la ingeniería implementando los conceptos adquiridos en el curso de métodos numéricos.
- Realizar el proceso de resolución de mecanismos mediante el uso recurrente de métodos numéricos como procedimiento de solución del problema con el fin de optimizar la precisión de la solución y de los recursos computacionales asociados al mismo.
- Implementar el método de Newton Raphson para lograr determinar los perfiles cinemáticos del mecanismo a partir de un movimiento ejercido por el eje de un motor.

2. Introducción y planteamiento del problema

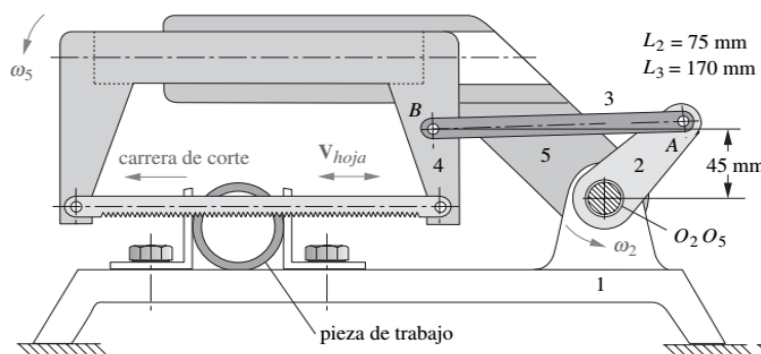


Figura 1: Mecanismo de trabajo tipo sierra eléctrica.

Se tiene un mecanismo de tipo sierra eléctrica, utilizado para cortar barras de metal. Este trabaja mediante transmisión de movimiento radial a lineal a través de la conexión de sus diferentes eslabones. Visto en la figura 1, el movimiento radial empieza con el eje en el punto 1, y termina en movimiento lineal en 4, permitiendo el corte de la pieza. Se va a determinar el análisis cinemático completo del mecanismo a partir del análisis de su cadena cinemática, logrando describir la posición, velocidad y aceleración de sus componentes para cualquier condición (1). Esto es necesario para lograr realizar un análisis cinético, de resistencia y a fatiga, sin embargo, estos conceptos no se abarcan en este informe. De este modo, se describen los vectores y ángulos del sistema:

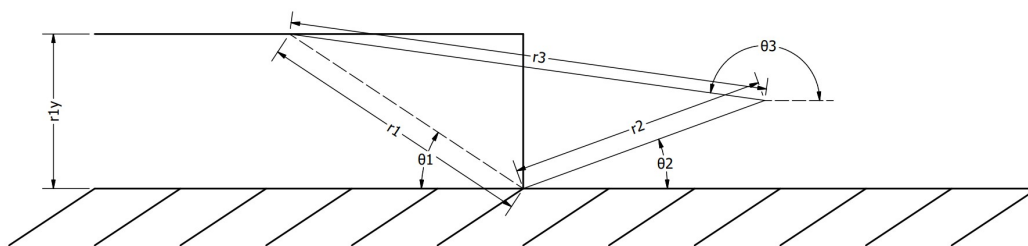


Figura 2: Diagrama del sistema.

Donde:

- | | | |
|---|--|--|
| ■ \vec{r}_1 : Eslabón formado por $\vec{r}_2 + \vec{r}_3$. | ■ θ_1 : Ángulo de \vec{r}_1 respecto a la horizontal. | ■ θ_3 : Ángulo de \vec{r}_3 respecto a la horizontal. |
| ■ V_r : Velocidad lineal de \vec{r}_1 . | ■ ω_1 : Velocidad angular de θ_1 . | ■ ω_3 : Velocidad angular de θ_3 . |
| ■ A_r : Aceleración lineal de \vec{r}_1 . | ■ α_1 : Aceleración angular de θ_1 . | ■ α_3 : Aceleración angular de θ_3 . |

3. Marco teórico

3.1. Método de Newton Raphson

3.1.1. Método de Newton Raphson en una variable real

El método de Newton Raphson es un método numérico abierto algorítmico que permite el cálculo de raíces, inicialmente planteado para una función real, pero extrapolado, para una función en el dominio de los complejos, o un sistema de ecuaciones lineales o no lineales. Este método tiene diversas aplicaciones en el campo de la computación y en general, en cualquier campo de aplicación matemática de problemas de solución de raíces, máximos y mínimos, optimización, ceros de funciones notables, entre otros (2).

Inicialmente el método tiene la definición más sencilla en una función real f y una aproximación inicial a la raíz x_0 , llamado valor semilla, y que está considerablemente cerca a la raíz que se espera encontrar, para posteriormente aproximar la función utilizando su recta tangente en dicho punto utilizando cálculo diferencial, encontrando el intercepto de dicha línea con el eje x . Dicho intercepto será una mejor aproximación de la raíz que el supuesto inicial, por lo cual es posible iterar este procedimiento hasta encontrar el valor que aproxime de mejor forma a la raíz con la tolerancia de error definida para la aproximación. Formalmente el método se puede definir de la siguiente forma (2):

Sea f de dominio y rango real una función diferenciable en el intervalo (a, b) , y una aproximación a la raíz de dicha función x_n , se puede encontrar una mejor aproximación x_{n+1} que se puede deducir desde la recta tangente a la curva dada como:

$$y = f'(x_n)(x - x_n) + f(x_n)$$

Evalutando para $(x, y) = (x_{n+1}, 0)$:

$$0 = f'(x_n)(x_{n+1} - x_n) + f(x_n)$$

Despejando para x_{n+1} :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

A partir de esta expresión se realiza el proceso iterativo de determinar x_{n+1} , sin embargo, se tienen ciertas condiciones de convergencia a tener en cuenta:

- El valor semilla x_0 debe estar relativamente cerca a la raíz.
- El valor semilla x_0 no puede ser estacionario, esto quiere decir que su derivada no puede ser cero, ya que al evaluar en el algoritmo, se genera una división por cero.
- La derivada debe ser continua en el intervalo donde se encuentra la raíz.

Tratándose de un método abierto, el error absoluto y relativo asociado al NR se define como:

$$\varepsilon_{abs} = |x_{n+1} - x_n| \qquad \varepsilon_{rel} = \frac{|x_{n+1} - x_n|}{|x_{n+1}|} \qquad (1)$$

3.1.2. Extrapolación del método a varias variables

En el caso de sistemas de ecuaciones, el método emplea el uso de derivadas parciales para hacer el mismo algoritmo iterativo de aproximación mediante tangentes. La aproximación se realiza variable

a variable de forma simultanea, por lo cual es necesario todas las derivadas parciales del sistema de ecuaciones (3).

Dicho esto, se tiene un sistema de ecuaciones lineal o no lineal compatible determinado, descrito por el vector columna de funciones $F(x)$, diferenciable en $F : \mathbb{R}^k \rightarrow \mathbb{R}^k$, donde x es el vector que contiene las variables del sistema, x_n es el vector columna con los valores actuales de la iteración de las variables, y x_{n+1} el vector columna que contiene los valores de la siguiente iteración. Para las derivadas, se aplica el concepto del Jacobiano del sistema, matriz que aplica un operador diferencial que permite derivar cada función respecto a cada variable y se obtienen los resultados en dicha matriz definida como:

$$J_F(x) = \begin{pmatrix} \frac{\partial f_1(x)}{\partial x_1} & \frac{\partial f_1(x)}{\partial x_2} & \dots & \frac{\partial f_1(x)}{\partial x_k} \\ \frac{\partial f_2(x)}{\partial x_1} & \frac{\partial f_2(x)}{\partial x_2} & \dots & \frac{\partial f_2(x)}{\partial x_k} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_k(x)}{\partial x_1} & \frac{\partial f_k(x)}{\partial x_2} & \dots & \frac{\partial f_k(x)}{\partial x_k} \end{pmatrix}$$

Ahora bien, aplicado al método, es necesario dividir por las derivadas en el algoritmo iterativo, por lo que se implementa la matriz inversa del Jacobiano, es decir, $J_F(x)^{-1}$. Finalmente, la ecuación iterativa para el método de Newton Raphson se da como:

$$x_{n+1} = x_n - J_F(x_n)^{-1} F(x_n) \quad (2)$$

Para el caso de varias variables, se puede determinar el error absoluto y relativo asociado a cada variable del vector x_n y determinar la norma del vector compuesto por los errores determinados.

4. Perfil cinemático del motor

Como condiciones iniciales del sistema, se va a generar el perfil cinemático del eje del motor. Dicho esto, se establece que el eje va a partir del reposo y, durante el periodo de tiempo en que el eje del motor complete su primera revolución, va a tener una aceleración máxima de 5 rad s^{-2} . Para minimizar el jerk del motor, este periodo de tiempo va a separarse en tres periodos de misma longitud:

1. Parte de aceleración en 0, en aumento lineal hasta 5 rad s^{-2} .
2. Se mantiene en 5 rad s^{-2} .
3. Disminuye de 5 rad s^{-2} hasta 0.

Después de haber ejercido la primera revolución, se va a mantener una velocidad angular constante $\omega_2 \approx 6,45 \text{ rad s}^{-1} \approx 62 \text{ rpm}$, la cual es la resultante de esta aceleración. De este modo, se puede definir las ecuaciones de aceleración angular, y mediante procesos de integración, su velocidad y posición angular. A continuación se muestran los resultados:

$$\alpha_2(t) = \begin{cases} \frac{15t}{T} & \text{si } 0 \leq t < T/3 \\ 5 & \text{si } T/3 \leq t < 2T/3 \\ 15 - \frac{15t}{T} & \text{si } 2T/3 \leq t < T \\ 0 & \text{si } T \leq t \end{cases} \quad (3)$$

$$\omega_2(t) = \begin{cases} \frac{15t^2}{2T} & \text{si } 0 \leq t < T/3 \\ 5t - \frac{5T}{6} & \text{si } T/3 \leq t < 2T/3 \\ 15t - \frac{25T}{6} - \frac{15t^2}{2T} & \text{si } 2T/3 \leq t < T \\ \frac{10T}{3} & \text{si } T \leq t \end{cases} \quad (4)$$

$$\theta_2(t) = \begin{cases} \frac{5t^3}{6} & \text{si } 0 \leq t < T/3 \\ \frac{5T^2}{6} - \frac{5t(T-3t)}{6} & \text{si } T/3 \leq t < 2T/3 \\ \frac{5T^2}{6} - \frac{25Tt}{6} + \frac{15t^2}{2T} - \frac{5t^3}{6T} & \text{si } 2T/3 \leq t < T \\ \frac{10Tt}{3} - \frac{5T^2}{3} & \text{si } T \leq t \end{cases} \quad (5)$$

Con base a los resultados obtenidos en (5), es posible determinar el tiempo T en que el eje realiza la primera revolución. Dado que $\theta_2(T) = 2\pi$, se soluciona para esta condición, determinando que $T = 1,942$ s. A partir de este resultado se grafican los valores de tiempo el perfil cinemático del eje del motor para un tiempo $2T$, con el fin de evidenciar el motor a velocidad constante.

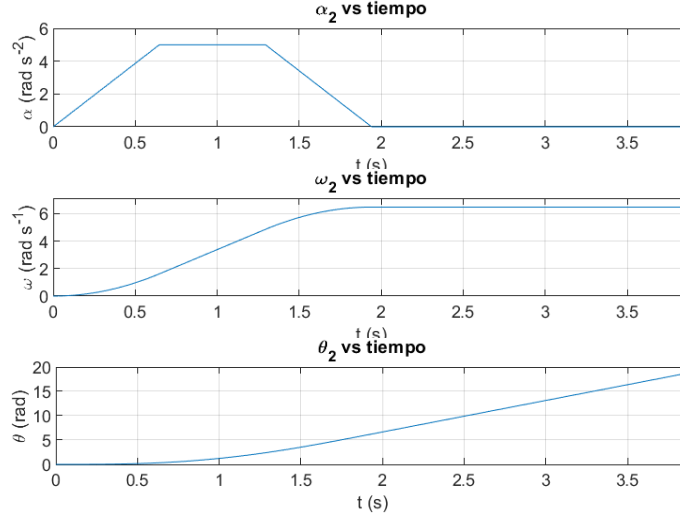


Figura 3: Perfil cinemático del eje del motor.

5. Resolución del problema

5.1. Cadena cinemática

Para poder solucionar el problema, lo primero es definir la cadena cinemática definida por los vectores del problema a partir de la figura 2. A continuación se evidencia la expresión correspondiente:

$$\vec{r}_1 = \vec{r}_2 + \vec{r}_3 \quad (6)$$

A partir de (6) se pueden definir las ecuaciones de posición respecto a las variables deseadas del sistema. Sumado a esto, igualando las ecuaciones a 0 y mediante procesos de derivación de las mismas, se pueden determinar los perfiles cinemáticos de las variables.

5.1.1. Posición

Con las condiciones dadas del problema, se pueden definir dos ecuaciones dimensionales respecto a los ejes coordenados x y y e igualarlas a 0. Adicionalmente, se debe plantear la ecuación de ligadura entre r_1 y θ_1 . Teniendo en cuenta que r_1 , θ_1 , θ_2 y θ_3 varían en el tiempo, se tiene que:

$$Fx(t) = 0 = -r_1(t)\cos(\theta_1(t)) + r_2\cos(\theta_2(t)) + r_3\cos(\theta_3(t)) \quad (7)$$

$$Fy(t) = 0 = r_1(t)\sin(\theta_1(t)) - r_2\sin(\theta_2(t)) - r_3\sin(\theta_3(t)) \quad (8)$$

$$Fc(t) = 0 = r_1(t)\sin(\theta_1(t)) - r_{1y} \quad (9)$$

5.1.2. Velocidad

Para las ecuaciones de velocidad basta con derivar las ecuaciones de posición respecto al tiempo.

$$\frac{\partial Fx(t)}{\partial t} = 0 = \omega_1 r_1 \sin(\theta_1) - V_r \cos(\theta_1) - \omega_2 r_2 \sin(\theta_2) - \omega_3 r_3 \sin(\theta_3) \quad (10)$$

$$\frac{\partial Fy(t)}{\partial t} = 0 = V_r \sin(\theta_1) + \omega_1 r_1 \cos(\theta_1) - \omega_2 r_2 \cos(\theta_2) - \omega_3 r_3 \cos(\theta_3) \quad (11)$$

$$\frac{\partial Fc(t)}{\partial t} = 0 = V_r \sin(\theta_1) + \omega_1 r_1 \cos(\theta_1) \quad (12)$$

5.1.3. Aceleración

Para determinar la aceleración, se derivan las funciones de velocidad.

$$\frac{\partial^2 Fx(t)}{\partial t^2} = 0 = r_1 \cos(\theta_1) \omega_1^2 + 2 V_r \sin(\theta_1) \omega_1 - r_2 \cos(\theta_2) \omega_2^2 - r_3 \cos(\theta_3) \omega_3^2 - a_r \cos(\theta_1) + \alpha_1 r_1 \sin(\theta_1) - \alpha_2 r_2 \sin(\theta_2) - \alpha_3 r_3 \sin(\theta_3) \quad (13)$$

$$\frac{\partial^2 Fy(t)}{\partial t^2} = 0 = -r_1 \sin(\theta_1) \omega_1^2 + 2 V_r \cos(\theta_1) \omega_1 + r_2 \sin(\theta_2) \omega_2^2 + r_3 \sin(\theta_3) \omega_3^2 + a_r \sin(\theta_1) + \alpha_1 r_1 \cos(\theta_1) - \alpha_2 r_2 \cos(\theta_2) - \alpha_3 r_3 \cos(\theta_3) \quad (14)$$

$$\frac{\partial^2 Fc(t)}{\partial t^2} = 0 = -r_1 \sin(\theta_1) \omega_1^2 + 2 V_r \cos(\theta_1) \omega_1 + a_r \sin(\theta_1) + \alpha_1 r_1 \cos(\theta_1) \quad (15)$$

5.2. Implementación del método de Newton Raphson

Con las ecuaciones del sistema definidas, se puede implementar el método de Newton Raphson para resolver las variables deseadas, con una cota de error $\varepsilon < 10^{-6}$. Inicialmente se trabajará el método de forma consecutiva, es decir, se resolverán las variables de posición, y se utilizarán sus valores como entradas para la resolución de la velocidad, y de la misma forma con los resultados de velocidad para la aceleración. Luego se realizará una segunda forma de aplicar el método, resolviendo las nueve variables de forma simultanea.

A partir de unas condiciones iniciales del motor en reposo: $\theta_2 = 0$, $\omega_2 = 0$ y $\alpha_2 = 0$, con valores de las constantes del sistema r_{1y} , r_2 y r_3 de 45 mm, 75 mm y 170 mm respectivamente, se va a tomar un valor semilla:

$$x_0 = \begin{pmatrix} r_1 \\ \theta_1 \\ \theta_3 \\ v_r \\ \omega_1 \\ \omega_3 \\ a_r \\ \alpha_1 \\ \alpha_3 \end{pmatrix} = \begin{pmatrix} -0,1 \\ -0,5 \\ 3 \\ 0,5 \\ 0 \\ -0,5 \\ 0,5 \\ 0 \\ -0,5 \end{pmatrix} \begin{bmatrix} m \\ rad \\ rad \\ m s^{-1} \\ rad s^{-1} \\ rad s^{-1} \\ m s^{-2} \\ rad s^{-2} \\ rad s^{-2} \end{bmatrix} \quad (16)$$

Se toman estos valores medianamente alejados de la solución para que se evidencie la convergencia mediante el método.

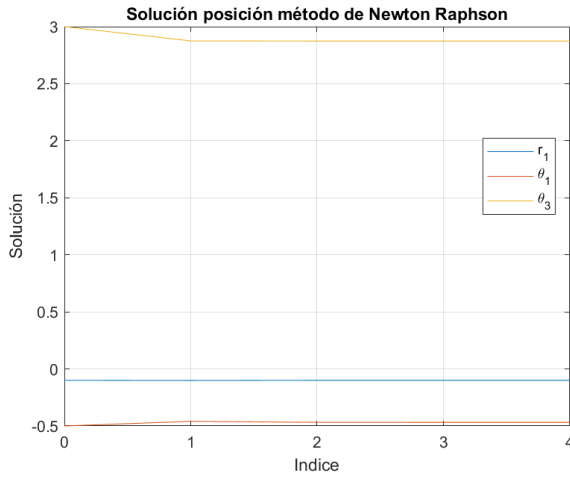
5.2.1. NR Consecutivo

Aplicando el método de Newton Raphson definido por (2), se implementan las expresiones de (7), (8) y (9) como el vector $F(x_n)$. Definiendo el jacobiano para $F(x_n)$ como:

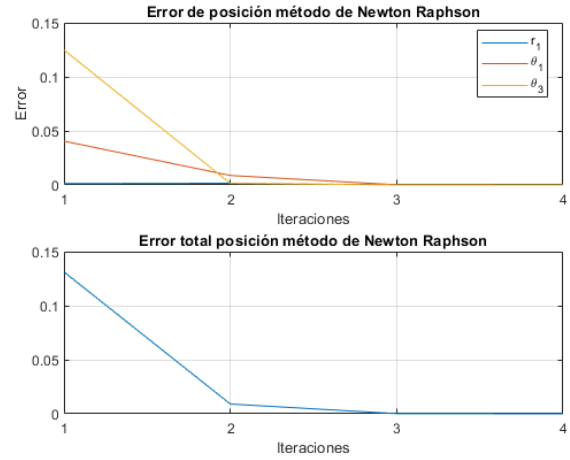
$$J_{pos} = \begin{pmatrix} \sin(\theta_1) & r_1 \cos(\theta_1) & 0 \\ 0 & r_{1y} (\cot(\theta_1)^2 + 1) & -r_3 \sin(\theta_3) \\ 0 & 0 & -r_3 \cos(\theta_3) \end{pmatrix} \quad (17)$$

Y su Jacobiano inverso que se encuentra en el Anexo C.1.

Con todo y lo anterior, se realiza el proceso iterativo para la solución del sistema, obteniendo las siguientes gráficas de la convergencia de las variables de posición r_1 , θ_1 y θ_3 , con sus respectivos errores por variable, y el error total:



(a) Convergencia de las variables de posición.



(b) Errores de las variables de posición.

Figura 4: Resultados del método de NR para la posición.

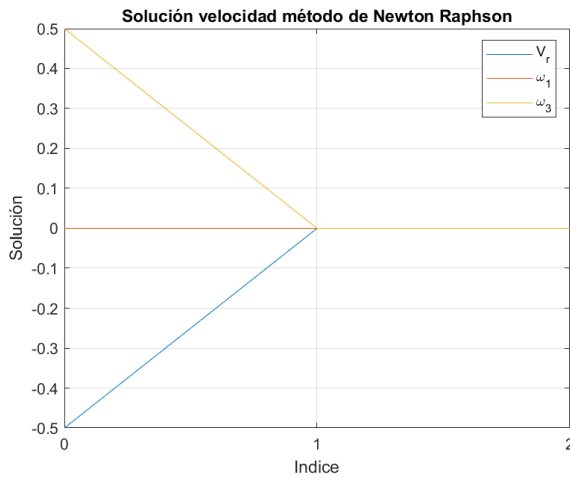
Después de 4 iteraciones, se tiene que:

$$r_1 = -0,099672 \text{ m} \quad \theta_1 = -0,468422 \text{ rad} \quad \theta_3 = 2,873694 \text{ rad}$$

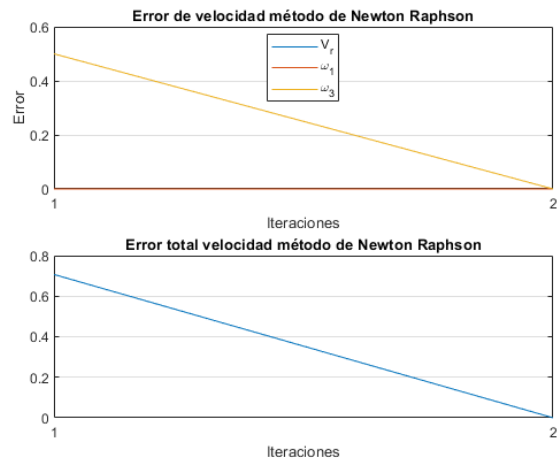
Para resolver las velocidades, se utilizan las ecuaciones (10), (11) y (12) para formar $F(x_n)$, y se define su matriz Jacobiana (su Jacobiano inverso se encuentra en el Anexo C.2):

$$J_{vel} = \begin{pmatrix} -\cos(\theta_1) & r_1 \sin(\theta_1) & -r_3 \sin(\theta_3) \\ \sin(\theta_1) & r_1 \cos(\theta_1) & -r_3 \cos(\theta_3) \\ \sin(\theta_1) & r_1 \cos(\theta_1) & 0 \end{pmatrix} \quad (18)$$

Al implementar el método descrito en (2) y los valores semilla dados, se obtienen los resultados para V_r , ω_1 y ω_3 :



(a) Convergencia de las variables de velocidad.



(b) Errores de las variables de velocidad.

Figura 5: Resultados del método de NR para la velocidad.

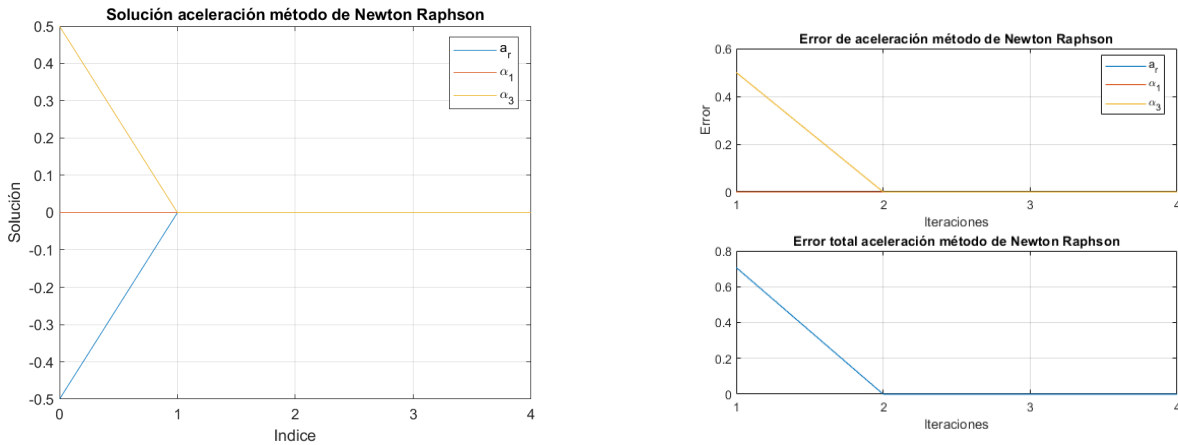
Después de 2 iteraciones, se tiene que:

$$V_1 = 0 \text{ m s}^{-1} \quad \omega_1 = 0 \text{ rad s}^{-1} \quad \omega_3 = 0 \text{ rad s}^{-1}$$

Finalmente, para los valores de aceleración, utilizando (13), (14) y (15) para formar $F(x_n)$, se define su Jacobiano (y su Jacobiano que se encuentra en el Anexo C.3) como:

$$J_{ac} = \begin{pmatrix} \sin(\theta_1) & r_1 \cos(\theta_1) & 0 \\ -\cos(\theta_1) & r_1 \sin(\theta_1) & -r_3 \sin(\theta_3) \\ \sin(\theta_1) & r_1 \cos(\theta_1) & -r_3 \cos(\theta_3) \end{pmatrix} \quad (19)$$

Al aplicar el método de NR con los valores semilla, se obtienen los siguientes resultados para A_r , α_1 y α_3 :



(a) Convergencia de las variables de aceleración.

(b) Errores de las variables de aceleración.

Figura 6: Resultados del método de NR para la aceleración.

Después de 4 iteraciones, se tiene que:

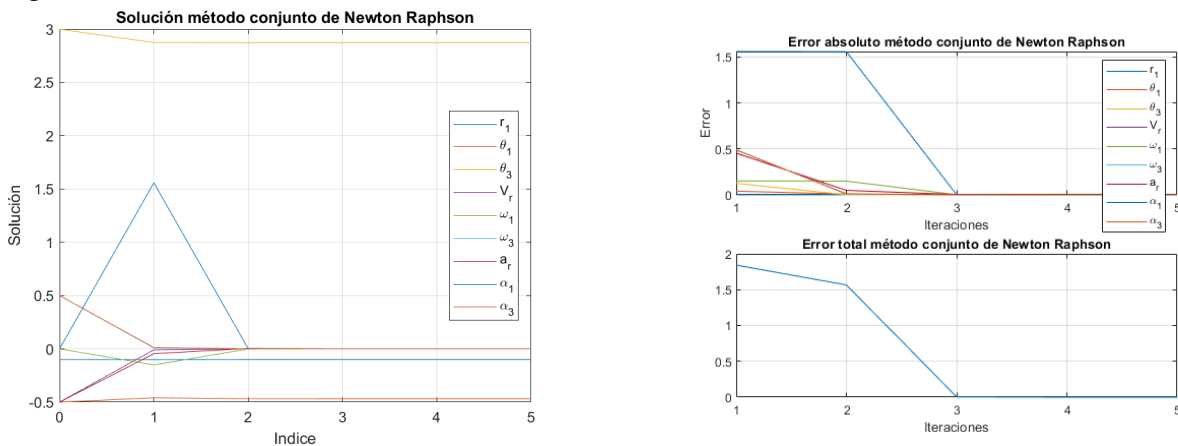
$$A_r = 0 \text{ m s}^{-2}$$

$$\alpha_1 = 0 \text{ rad s}^{-2}$$

$$\alpha_3 = 0 \text{ rad s}^{-2}$$

5.2.2. NR Simultaneo

Ahora bien, estudiando el método de NR, fue posible determinar que el sistema se puede resolver llevando a cabo una implementación simultanea de las nueve ecuaciones que describen la cinemática general del sistema. A pesar de que las soluciones de velocidad y aceleración dependan de valores de posición, y de posición y velocidad respectivamente, a partir del valor semilla y de la convergencia de la posición, se converge simultáneamente a la solución de velocidad, y del mismo modo para la aceleración. Debido a que el sistema se resuelve en simultaneo, $F(x_n)$ se define con las nueve ecuaciones de cinemática, su Jacobiano de tamaño 9×9 se encuentra en el anexo D, debido a la extensión del inverso del Jacobiano, no se anexa al documento. Con todo y lo anterior, se obtienen los siguientes resultados:



(a) Convergencia de las variables en simultaneo.

(b) Errores de las variables en simultaneo.

Figura 7: Resultados del método de NR en simultaneo.

Después de cinco iteraciones, se tiene que los valores finales son:

$$\begin{array}{lll} r_1 = -0,099672 \text{ m} & \theta_1 = -0,468422 \text{ rad} & \theta_3 = 2,873694 \text{ rad} \\ V_r = 0 \text{ m s}^{-1} & \omega_1 = 0 \text{ rad s}^{-1} & \omega_3 = 0 \text{ rad s}^{-1} \\ A_r = 0 \text{ m s}^{-2} & \alpha_1 = 0 \text{ rad s}^{-2} & \alpha_3 = 0 \text{ rad s}^{-2} \end{array}$$

Comparando estos resultados con los encontrados de manera consecutiva, se evidencia que ambos métodos convergen a la misma solución.

5.3. Solución analítica

A pesar de que el problema se va a solucionar implementando el método de Newton Raphson, es importante tener la solución analítica al problema. Esto se va a realizar porque es necesario comparar los resultados obtenidos por el método de NR con los valores exactos del sistema, para determinar el error real obtenido en la solución. Además, si se fueran a comparar los resultados del NR directamente con los resultados de simulación se pueden tener errores de medición, debido a las limitaciones de los software como *Inventor*.

Dicho esto, a partir de las ecuaciones encontradas en la sección 5.1, es posible determinar las soluciones analíticas para las nueve variables del sistema. Estas ecuaciones se determinan mediante procedimientos matemáticos que no se van a profundizar en el informe.

$$\theta_3 = \pi - \arcsin\left(\frac{r_{1y} - r_2 \sin(\theta_2)}{r_3}\right) \quad (20)$$

$$\theta_1 = \operatorname{acot}\left(\frac{r_2 \cos(\theta_2) + r_3 \cos(\theta_3)}{r_{1y}}\right) \quad (21)$$

$$r_1 = -\frac{\omega_2 r_2 \cos(\theta_2)}{r_3 \cos(\theta_3)} \quad (22)$$

$$\omega_3 = \operatorname{acot}\left(\frac{r_2 \cos(\theta_2) + r_3 \cos(\theta_3)}{r_{1y}}\right) \quad (23)$$

$$\omega_1 = -\frac{V_r \tan(\theta_1)}{r_1} \quad (24)$$

$$V_r = -\cos(\theta_1) (\omega_2 r_2 \sin(\theta_2) + \omega_3 r_3 \sin(\theta_3)) \quad (25)$$

$$\alpha_3 = \frac{r_2 \sin(\theta_2) \omega_2^2 + r_3 \sin(\theta_3) \omega_3^2 - \alpha_2 r_2 \cos(\theta_2)}{r_3 \cos(\theta_3)} \quad (26)$$

$$\alpha_1 = \omega_1^2 \tan(\theta_1) - \frac{2 V_r \omega_1 + a_r \tan(\theta_1)}{r_1} \quad (27)$$

$$\begin{aligned} A_r = r_1 \omega_1^2 - r_2 \cos(\theta_1) \cos(\theta_2) \omega_2^2 - r_3 \cos(\theta_1) \cos(\theta_3) \omega_3^2 \\ - \alpha_2 r_2 \cos(\theta_1) \sin(\theta_2) - \alpha_3 r_3 \cos(\theta_1) \sin(\theta_3) \end{aligned} \quad (28)$$

6. Resultados

Al demostrar la convergencia del método de Newton Raphson para el punto inicial del movimiento del mecanismo, es posible definir todo el movimiento rotacional del mecanismo bajo los perfiles cinemáticos del motor descritos en la sección 4. Esto se realizará mediante la implementación del NR de forma simultanea, de forma consecutiva, de manera analítica, y mediante el software de diseño asistido por computadora *Autodesk Inventor*. La solución se realiza para 1000 puntos distribuidos en un periodo de $2T \approx 3,88 \text{ s}$.

Acerca del método del NR, los valores semilla del punto P_i serán la solución determinada por el método para el punto P_{i-1} , lo cual garantiza que el valor semilla sea cercano a la solución. Para el

punto inicial en $t = 0$ cuando el motor está en reposo, el valor semilla será los resultados obtenidos en la sección 5.2.1 y 5.2.2, los cuales son equivalentes.

6.1. NR Simultaneo

Al aplicar el método de Newton Raphson de forma simultanea se obtienen los siguientes perfiles de posición, velocidad y aceleración para las variables a encontrar. También se muestra el error asociado al método.

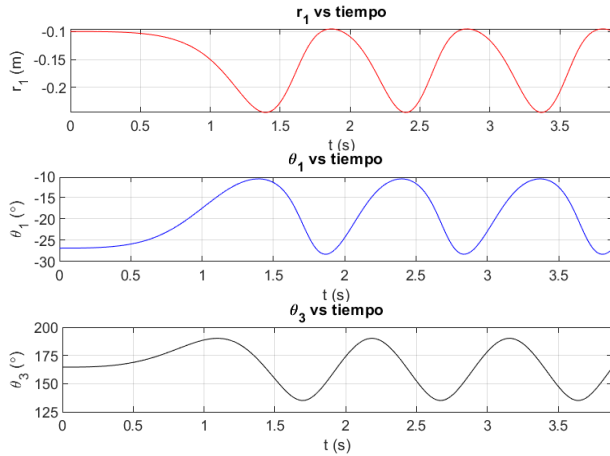


Figura 8: Posiciones para NR simultaneo.

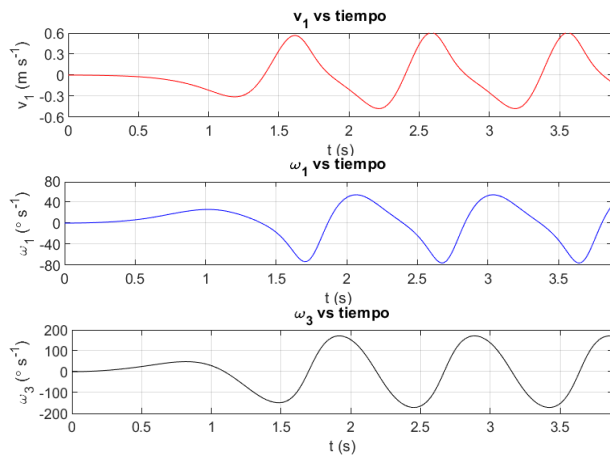


Figura 9: Velocidades para NR simultaneo.

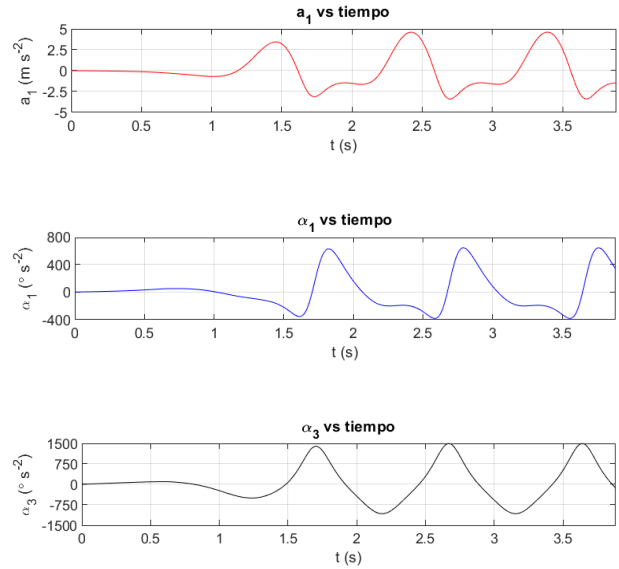


Figura 10: Aceleraciones para NR simultaneo.

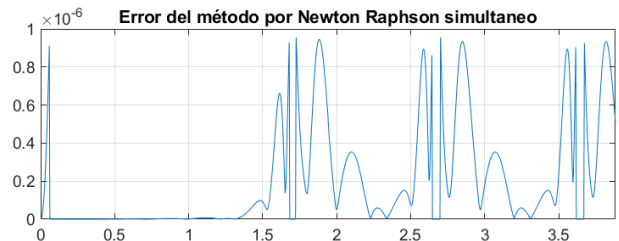


Figura 11: Error asociado al método de NR simultaneo.

6.2. NR Consecutivo

El mismo procedimiento realizado para el método de Newton Raphson de forma simultanea se realiza para el método de forma consecutiva. Debido a la extensión del documento, y a la similitud que se tienen con los resultados anteriores, las gráficas de resultados se encuentran en el anexo A.

6.3. Simulación

Para poder corroborar los resultados obtenidos mediante el uso del método de Newton Raphson, se decide hacer un modelado 3D del mecanismo en el software *Autodesk Inventor*. El mecanismo simulado resume el movimiento de la sierra eléctrica real utilizando eslabones y juntas que definidos a partir de las condiciones iniciales del problema.



Figura 12: Modelo en Inventor del Mecanismo 3D

Como se puede evidenciar, a la bancada se conecta la manivela, el cual es el eslabón donde se genera el movimiento por el perfil cinemático del motor. A la manivela se le conecta el acoplador, el cual es el eslabón de mayor extensión, también conectado a la corredera, la cual se une a la sierra y se pueden desplazar sobre la guía de la bancada.

Los resultados de simulación se generan a partir del entorno de simulación dinámica del software. En este entorno, se impone el perfil de velocidad del motor al eje de la manivela, donde es el mismo perfil utilizado para encontrar los resultados por NR. Con esto, se genera la simulación del modelado, donde se pueden extraer los valores cinemáticos de todas las variables implicadas en el sistema. Cabe resaltar que, al igual que con el método de NR, se resuelve para 1000 puntos en un periodo de 3,883 s. Las gráficas de resultados se encuentran en el anexo B.

7. Análisis de resultados

7.1. Comparación de aplicación de NR

Se realizaron dos variaciones del método de Newton Raphson, uno donde se resolvían las nueve variables en simultaneo, y otra donde se resolvían de manera consecutiva, por lo que es conveniente señalar las diferencias entre estos dos casos. Lo primero es comparar los errores asociados al método, comparando la figura 11 con la figura 24 del anexo A, El error del método simultáneo suele ser mayor al método consecutivo, sin embargo, todos los valores están acotados por la tolerancia establecida ε . Ahora, se muestra el error absoluto entre estos dos casos:

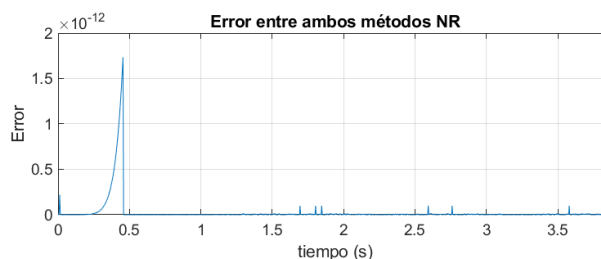
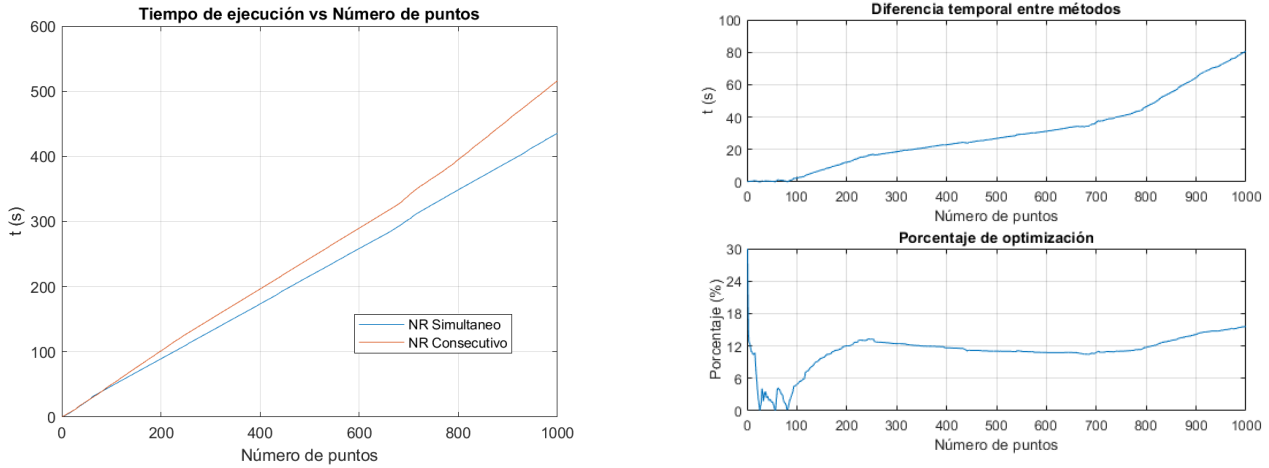


Figura 13: Error absoluto entre las versiones del método de Newton Raphson

Como se puede evidenciar, un error absoluto en el orden de 10^{-12} mientras se tiene una cota de error del método de $\varepsilon < 10^{-6}$ indica que los resultados obtenidos entre los métodos fueron bastante precisos a pesar de resolverse de maneras distintas.

Ahora bien, para definir cual de estos dos se toma como la solución final del sistema, se va a mostrar el tiempo de ejecución para cada caso. De este modo, se puede establecer cual caso es más eficiente en calcular los resultados.



(a) Tiempo de ejecución para los métodos.

(b) Diferencia en los tiempos de ejecución.

Figura 14: Resultados de tiempo de ejecución del sistema.

Como se puede evidenciar, resolver el sistema mediante el método de Newton Raphson de manera simultánea es más óptimo que realizarlo de manera consecutiva. A pesar de que los tiempos de ejecución tienen un comportamiento lineal, la pendiente del método de forma consecutiva es más elevada. Dicho esto, considerando que los resultados obtenidos por ambos métodos son casi iguales, y que se tiene una reducción del tiempo de ejecución de aproximadamente 12 % para el método en simultáneo, este se considera como la mejor solución, teniendo en cuenta que ese porcentaje es considerablemente alto cuando se trabaja con un número de datos elevados. Así pues, se van a considerar los resultados obtenidos por el método de NR simultáneo como los resultados finales del problema.

7.2. Error absoluto del método de Newton Raphson

A partir de las ecuaciones determinadas de la solución analítica en la sección 5.3 se pueden determinar las soluciones exactas del problema, donde se van a utilizar para determinar el error absoluto que se generó al solucionar el sistema. Al realizar este procedimiento, se determinan los siguientes errores absolutos a las soluciones encontradas.

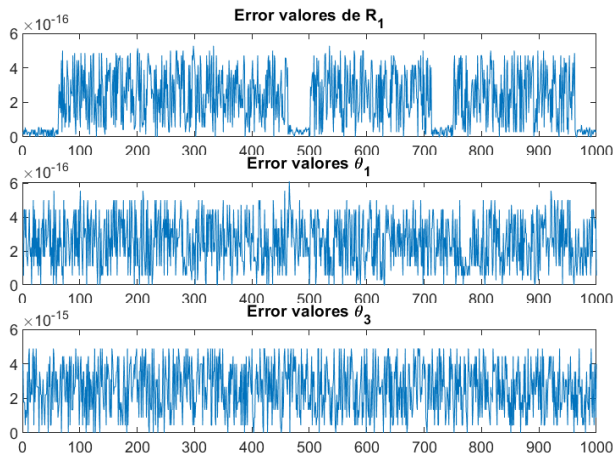


Figura 15: Error absoluto para resultados de posición.

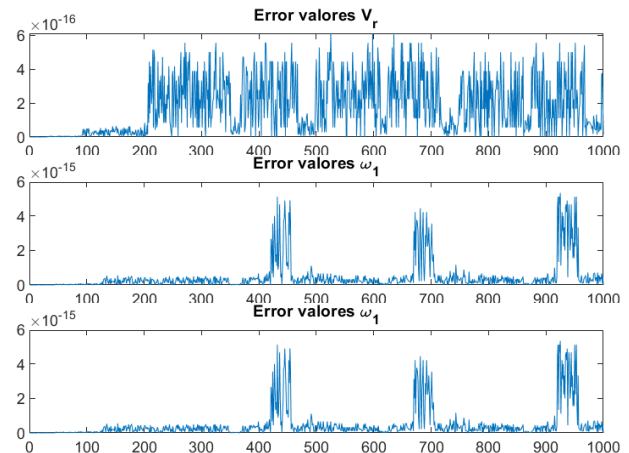


Figura 16: Error absoluto para resultados de velocidad.

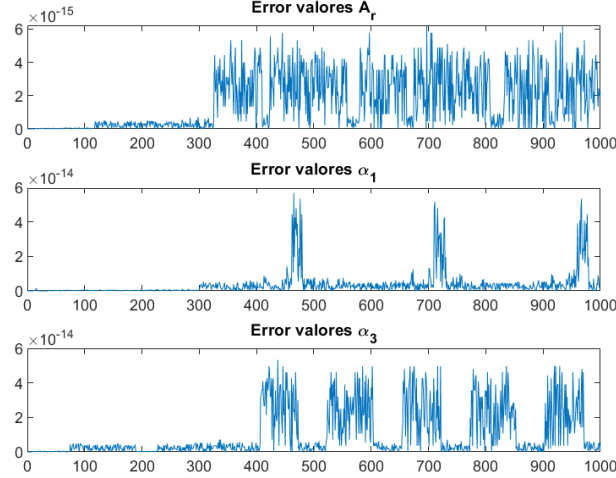


Figura 17: Error absoluto para resultados de aceleración.

Como se puede evidenciar en las gráficas, el error absoluto asociado a las posiciones y velocidades del sistema se encuentra en el orden de 10^{-15} , mientras que para las aceleraciones se encuentra en el orden de 10^{-14} . Ahora bien, teniendo en cuenta que el error de NR se acotó como $\varepsilon < 10^{-6}$, y que el error absoluto de todos los resultados siempre es menor a 10^{-13} , se establece que los valores obtenidos mediante el método de Newton Raphson simultaneo fueron altamente exactos y que resolver el sistema mediante esta versión del método numérico es una manera bastante sencilla y efectiva de solución.

7.3. Comparación de resultados NR-Simulación *Inventor*

Para finalizar, se van a comparar los resultados obtenidos por el método de Newton Raphson con los valores de simulación que se pueden evidenciar en el anexo B. De este modo, los resultados de *Inventor* se obtienen a partir de una interfaz de simulación dinámica altamente utilizada para resolver este tipo de sistemas en las aplicaciones de ingeniería, debido a su capacidad de mostrar de manera virtual el comportamiento de los mecanismos ensamblados en la vida real. Hecha esta observación, se superponen los resultados obtenidos para el sistema con los valores obtenidos en *Inventor*, obteniendo las siguientes gráficas:

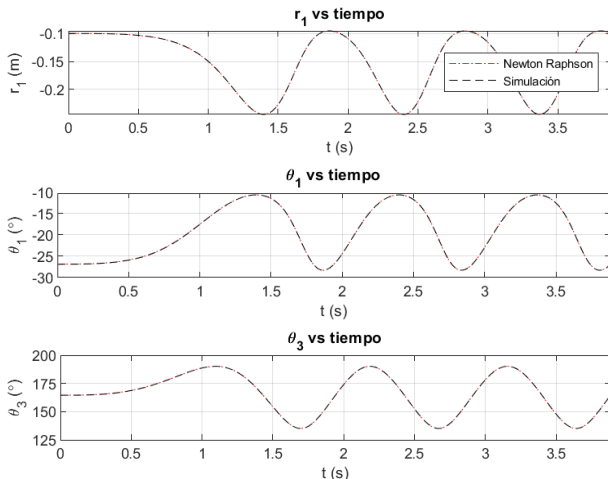


Figura 18: Comparación NR-Inventor para el perfil de posición.

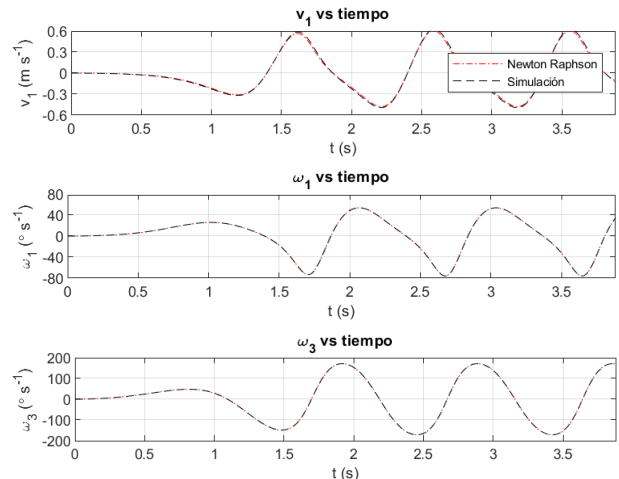


Figura 19: Comparación NR-Inventor para el perfil de velocidad.

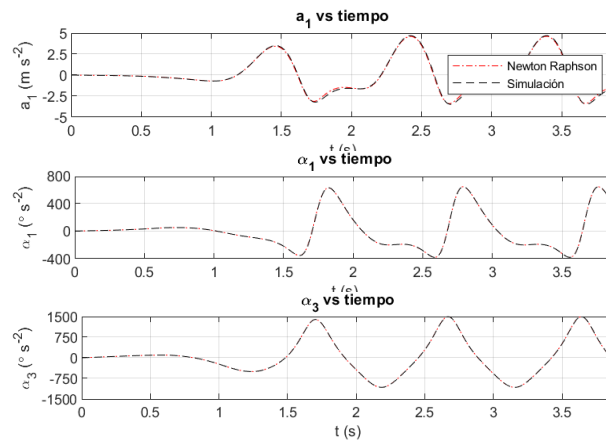


Figura 20: Comparación NR-Inventor para el perfil de aceleración.

Como es posible apreciar mediante el análisis de las gráficas, los resultados obtenidos en la simulación en *Inventor* concuerdan con los valores obtenidos mediante el método de Newton Raphson simultaneo. Esto quiere decir que mediante la implementación de sistemas de solución por simulaciones CAD se obtienen resultados semejantes a los que se determinan resolviendo el mismo sistema utilizando el método numérico de Newton Raphson para sistemas de ecuaciones no lineales, con lo que se puede establecer la veracidad de solución del mecanismo a partir del método.

8. Conclusiones

- Se puede implementar el método de Newton Raphson para resolver los sistemas de ecuaciones que describen problemas aplicados a la ingeniería, en este caso, los perfiles cinemáticos asociados a un mecanismo.
- El uso recurrente de métodos numéricos se puede implementar como procedimiento de solución a la cinemática de mecanismos para cualquier instante t o intervalo $[t_0 \ t_1]$, esto con el fin de optimizar la precisión de la solución y de los recursos computacionales asociados al mismo.
- Se obtienen los mismos resultados finales resolviendo mediante Newton Raphson el sistema de posición, velocidad y aceleración simultáneamente, a resolver cada uno de estos de manera individual. Además, resolver el mecanismo de manera simultánea reduce los recursos computacionales asociados a la solución, disminuyendo el tiempo total para solucionar el sistema.
- Se tiene una alta exactitud de solución al problema al implementar el método de Newton Raphson, obteniendo un error absoluto con la solución analítica menor a 10^{-13} para cualquier caso del sistema. Sumado a esto, al contrastar los resultados con las simulaciones del mecanismo en *Autodesk Inventor*, se logra establecer la veracidad de la solución para toda variable e instante del movimiento.

9. Referencias

- [1] Myzka, H. David, Máquinas y Mecanismos, Pearson, México, 2012
- [2] Chapra, S. C., & Canale, R. P. Numerical methods for engineers. McGraw-Hill Higher Education, Boston, 2006
- [3] Mathews, J.H. & Fink, K.D. Métodos numéricos con Matlab. Prentice Hall, Madrid, 2000

A. Gráficas de resultados para el método de Newton Raphson consecutivo

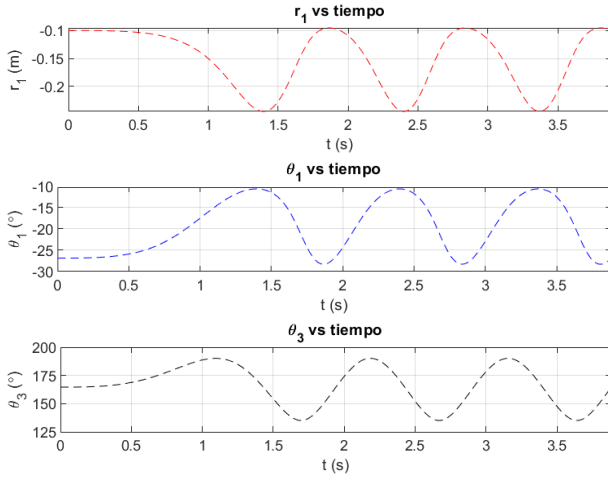


Figura 21: Posiciones para NR consecutivo.

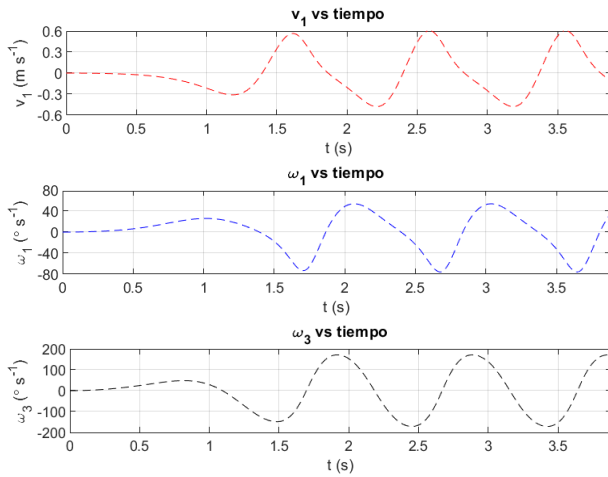


Figura 22: Velocidades para NR consecutivo.

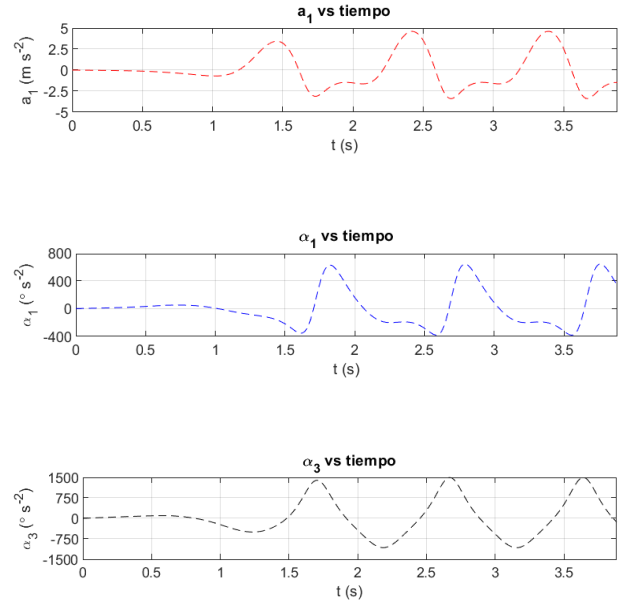


Figura 23: Aceleraciones para NR consecutivo.

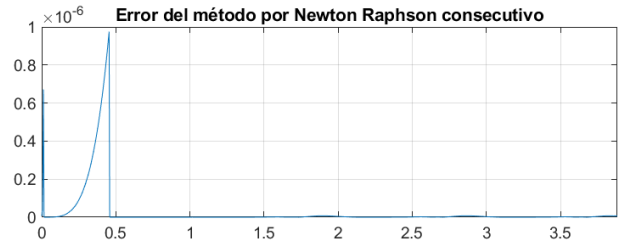


Figura 24: Error asociado al método de NR consecutivo.

B. Resultados de simulación de inventor

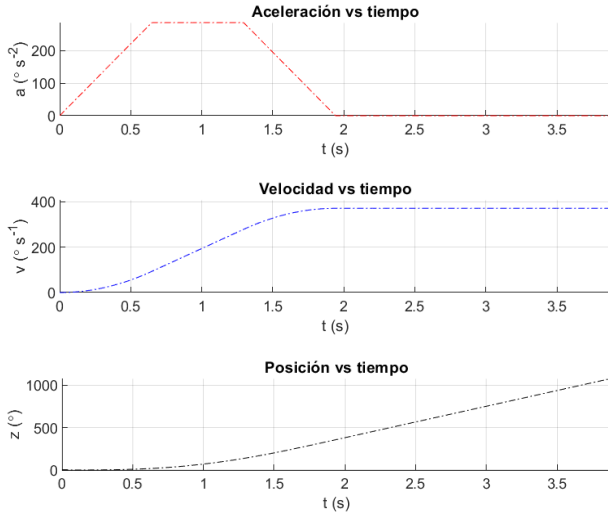


Figura 25: Perfil cinemático del eje del motor en Autodesk Inventor.

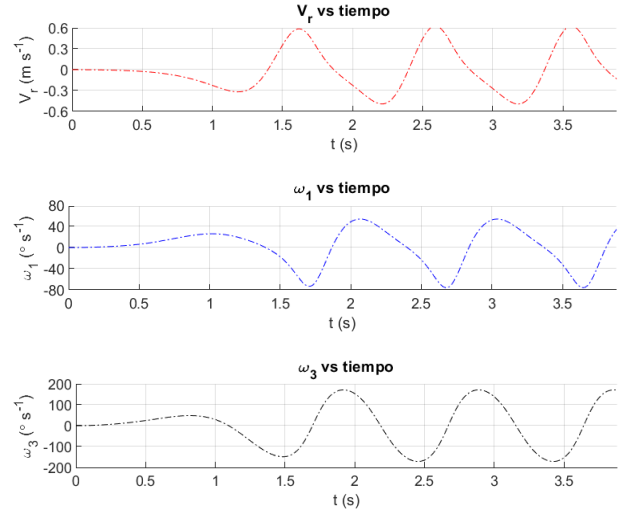


Figura 27: Velocidades obtenidas en Autodesk Inventor

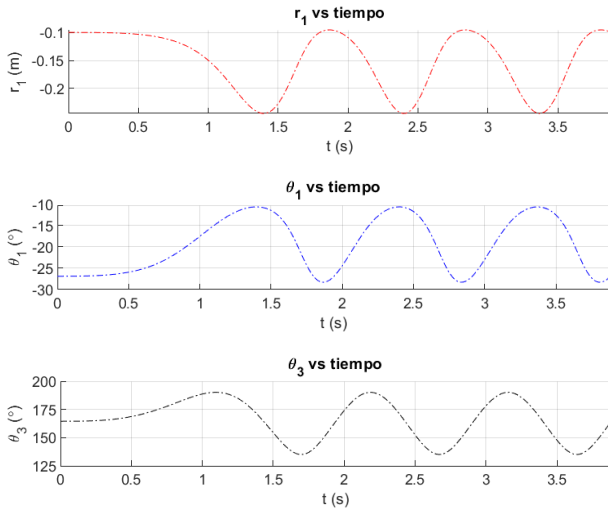


Figura 26: Posiciones obtenidas en Autodesk Inventor

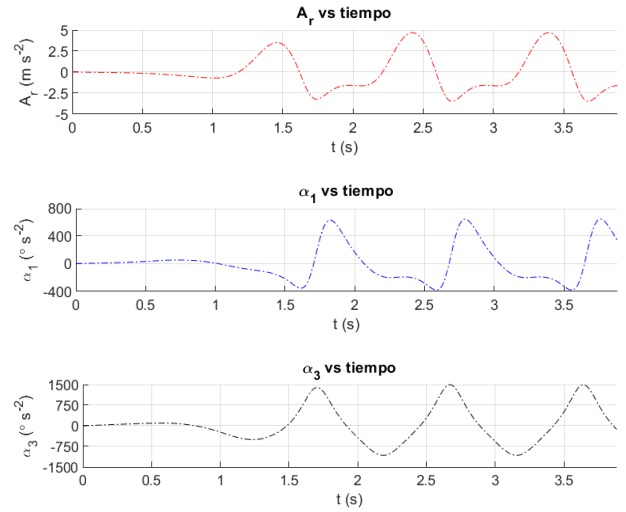


Figura 28: Aceleraciones obtenidas en Autodesk Inventor.

C. Jacobianos Inversos

C.1. Posición:

$$J_{pos}^{-1} = \begin{pmatrix} \frac{1}{\sin(\theta_1)} & -\frac{r_1 \cos(\theta_1)}{r_{1y} \sin(\theta_1) \sigma_1} & \frac{r_1 \cos(\theta_1) \sin(\theta_3)}{r_{1y} \cos(\theta_3) \sin(\theta_1) \sigma_1} \\ 0 & \frac{1}{r_{1y} \sigma_1} & -\frac{\sin(\theta_3)}{r_{1y} \cos(\theta_3) \sigma_1} \\ 0 & 0 & -\frac{1}{r_3 \cos(\theta_3)} \end{pmatrix}$$

Donde:

$$\sigma_1 = \cot(\theta_1)^2 + 1$$

C.2. Velocidad:

$$J_{vel}^{-1} = \begin{pmatrix} -\frac{\cos(\theta_1)}{\cos(\theta_1)^2 + \sin(\theta_1)^2} & \frac{\sigma_4}{\sigma_2} & -\frac{\sigma_4 - \cos(\theta_3) \sin(\theta_1)}{\sigma_2} \\ \frac{\sin(\theta_1)}{r_1 \cos(\theta_1)^2 + r_1 \sin(\theta_1)^2} & -\frac{\sigma_3}{\sigma_1} & \frac{\cos(\theta_1) \cos(\theta_3) + \sigma_3}{\sigma_1} \\ 0 & -\sigma_5 & \sigma_5 \end{pmatrix}$$

Donde:

$$\sigma_1 = r_1 \cos(\theta_3) \cos(\theta_1)^2 + r_1 \cos(\theta_3) \sin(\theta_1)^2$$

$$\sigma_2 = \cos(\theta_3) \cos(\theta_1)^2 + \cos(\theta_3) \sin(\theta_1)^2$$

$$\sigma_3 = \sin(\theta_1) \sin(\theta_3)$$

$$\sigma_4 = \cos(\theta_1) \sin(\theta_3)$$

$$\sigma_5 = \frac{1}{r_3 \cos(\theta_3)}$$

C.3. Aceleración:

$$J_{ac}^{-1} = \begin{pmatrix} -\frac{\sigma_4 - \cos(\theta_3) \sin(\theta_1)}{\sigma_2} & -\frac{\cos(\theta_1)}{\cos(\theta_1)^2 + \sin(\theta_1)^2} & \frac{\sigma_4}{\sigma_2} \\ \frac{\cos(\theta_1) \cos(\theta_3) + \sigma_3}{\sigma_1} & \frac{\sin(\theta_1)}{r_1 \cos(\theta_1)^2 + r_1 \sin(\theta_1)^2} & -\frac{\sigma_3}{\sigma_1} \\ \sigma_5 & 0 & -\sigma_5 \end{pmatrix}$$

Donde:

$$\sigma_1 = r_1 \cos(\theta_3) \cos(\theta_1)^2 + r_1 \cos(\theta_3) \sin(\theta_1)^2$$

$$\sigma_2 = \cos(\theta_3) \cos(\theta_1)^2 + \cos(\theta_3) \sin(\theta_1)^2$$

$$\sigma_3 = \sin(\theta_1) \sin(\theta_3)$$

$$\sigma_4 = \cos(\theta_1) \sin(\theta_3)$$

$$\sigma_5 = \frac{1}{r_3 \cos(\theta_3)}$$

D. Jacobiano del sistema simultaneo:

Col. 1-3:

$$\begin{pmatrix} \sin(\theta_1) & r_1 \cos(\theta_1) & 0 \\ 0 & r_{1y}(\cot(\theta_1)^2 + 1) & \sigma_4 \\ 0 & 0 & \sigma_5 \\ \omega_1 \sin(\theta_1) & V_r \sin(\theta_1) + \omega_1 r_1 \cos(\theta_1) & -\omega_3 r_3 \cos(\theta_3) \\ \omega_1 \cos(\theta_1) & \sigma_6 & \omega_3 r_3 \sin(\theta_3) \\ \omega_1 \cos(\theta_1) & \sigma_6 & 0 \\ \sigma_3 & \sigma_1 & 0 \\ \cos(\theta_1) \omega_1^2 + \alpha_1 \sin(\theta_1) & -r_1 \sin(\theta_1) \omega_1^2 + 2 V_r \cos(\theta_1) \omega_1 + a_r \sin(\theta_1) + \alpha_1 r_1 \cos(\theta_1) & \omega_3^2 r_3 \sin(\theta_3) - \alpha_3 r_3 \cos(\theta_3) \\ \sigma_3 & \sigma_1 & r_3 \cos(\theta_3) \omega_3^2 + \alpha_3 r_3 \sin(\theta_3) \end{pmatrix}$$

Col. 4-6:

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -\cos(\theta_1) & r_1 \sin(\theta_1) & \sigma_4 \\ \sin(\theta_1) & r_1 \cos(\theta_1) & \sigma_5 \\ \sin(\theta_1) & r_1 \cos(\theta_1) & 0 \\ \sigma_7 & \sigma_2 & 0 \\ 2 \omega_1 \sin(\theta_1) & 2 V_r \sin(\theta_1) + 2 \omega_1 r_1 \cos(\theta_1) & -2 \omega_3 r_3 \cos(\theta_3) \\ \sigma_7 & \sigma_2 & 2 \omega_3 r_3 \sin(\theta_3) \end{pmatrix}$$

Col. 7-9:

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \sigma_7 & \cos(\theta_1) & 0 \\ -\cos(\theta_1) & r_1 \sin(\theta_1) & \sigma_4 \\ \sin(\theta_1) & r_1 \cos(\theta_1) & \sigma_5 \end{pmatrix}$$

Donde:

$$\sigma_1 = -r_1 \cos(\theta_1) \omega_1^2 - 2 V_r \sin(\theta_1) \omega_1 + a_r \cos(\theta_1) - \alpha_1 r_1 \sin(\theta_1)$$

$$\sigma_2 = 2 V_r \cos(\theta_1) - 2 \omega_1 r_1 \sin(\theta_1)$$

$$\sigma_3 = \alpha_1 \cos(\theta_1) - \omega_1^2 \sin(\theta_1)$$

$$\sigma_4 = -r_3 \sin(\theta_3)$$

$$\sigma_5 = -r_3 \cos(\theta_3)$$

$$\sigma_6 = V_r \cos(\theta_1) - \omega_1 r_1 \sin(\theta_1)$$

$$\sigma_7 = 2 \omega_1 \cos(\theta_1)$$

E. Códigos de Matlab del desarrollo

A continuación se expone el listado de códigos generados para la solución del proyecto y la función que estos realizan:

1. **Perfil cinemático:** Enfocado en la realización del perfil cinemático del motor, el cual se utiliza como datos de entrada al sistema.
2. **Solución por Newton Raphson consecutivo:** Solución para un único punto para el método de Newton Raphson solucionando la posición, velocidad y aceleración de manera consecutiva.
3. **Solución por Newton Raphson simultáneo:** Solución para un único punto para el método de Newton Raphson solucionando la posición, velocidad y aceleración de manera simultánea.
4. **Solución por método de Newton Raphson:** Solución a todo el perfil cinemático del motor para el mecanismo. Se implementan ambas versiones del NR y se comparan entre si.
5. **Solución analítica:** Se realiza la solución analítica para todo el perfil cinemático del motor para el mecanismo.
6. **Resultados NR simultáneo vs solución analítica:** Comparar los resultados obtenidos por el método de NR con la solución analítica.
7. **Resultados NR simultáneo vs simulación:** Comparar los resultados obtenidos por el método de NR con los resultados de simulación.

Cabe resaltar que la mayoría de archivos utilizan datos externos en formato .txt, generados por los mismos u otro archivo de Matlab, por lo que es importante verificar la ruta de acceso a estos archivos.

E.1. Perfil cinemático

```

1 clear
2 clc
3 syms a1(t) a2(t) a3(t) a4(t) a5(t) T A
4 syms v1(t) v2(t) v3(t) v4(t) v5(t)
5 syms x1(t) x2(t) x3(t) x4(t) x5(t)
6 assume(T>0)
7 assume(A>0)
8
9 A=5;
10
11 Aceleracion
12 a1(t)=3*A/T*t;
13 a2(t)=A;
14 a3(t)=-3*A/T*t;
15 Ca3=A-a3(2*T/3);
16 a3(t)=a3(t)+Ca3;
17 a4(t)=0;
18 fprintf('Funcion de aceleracion:')
19 aceleracion(t)=piecewise((0<=t<T/3),a1,(T/3<=t<2*T/3),a2,(2*T/3<=t<T),a3,(T<=t),a4)
20
21 Velocidad
22 v1(t)=int(a1);
23 v2(t)=int(a2);
24 Cv2=v1(T/3)-v2(T/3);
25 v2(t)=v2(t)+Cv2;
26 v3(t)=int(a3);
27 Cv3=v2(2*T/3)-v3(2*T/3);
28 v3(t)=v3(t)+Cv3;
29 v4(t)=int(a4);
30 Cv4=v3(T)-v4(T);
31 v4(t)=v4(t)+Cv4;
32 fprintf('Funcion de velocidad:')
33 velocidad(t)=piecewise((0<=t<T/3),v1,(T/3<=t<2*T/3),v2,(2*T/3<=t<T),v3,(t<=T),v4)
34
35 Desplazamiento
36 z1(t)=int(v1);
37 z2(t)=int(v2);
38 Cz2=z1(T/3)-z2(T/3);
39 z2(t)=z2(t)+Cz2;
40 z3(t)=int(v3);
41 Cz3=z2(2*T/3)-z3(2*T/3);
42 z3(t)=z3(t)+Cz3;
43 z4(t)=int(v4);
44 Cz4=z3(T)-z4(T);
45 z4(t)=z4(t)+Cz4;
46 fprintf('Funcion de desplazamiento:')
47 posicion(t)=piecewise((0<=t<T/3),z1,(T/3<=t<2*T/3),z2,(2*T/3<=t<T),z3,(t<=T),z4)
48
49 Solucion de tMax
50 f=2*pi==posicion(T);
51 Tgiro=solve(f,T);
52 aceleracion=subs(aceleracion,T,Tgiro);
53 velocidad=subs(velocidad,T,Tgiro);
54 posicion=subs(posicion,T,Tgiro);

```



```

55 fprintf('Tiempo de primera vuelta %.3f s',Tgiro)
56
57 Graficas
58 ts=linspace(0,2*Tgiro,1000);
59 Vmax=double(velocidad(Tgiro));
60
61 subplot(3,1,1)
62 plot(ts,aceleracion(ts))
63 title('Aceleracion angular vs tiempo')
64 grid on
65 xlabel('t (s)')
66 ylabel('\alpha (rad s^{-2})')
67 ylim([0 6])
68 xlim([0 double(2*Tgiro)])
69 yticks(0:2:6)
70
71 subplot(3,1,2)
72 plot(ts,velocidad(ts))
73 title('Velocidad angular vs tiempo')
74 grid on
75 xlabel('t (s)')
76 ylabel('\omega (rad s^{-1})')
77 ylim([0 Vmax*1.1])
78 xlim([0 double(2*Tgiro)])
79 yticks(0:2:8)
80
81 subplot(3,1,3)
82 plot(ts,posicion(ts))
83 title('Posicion angular vs tiempo')
84 grid on
85 xlabel('t (s)')
86 ylabel('\theta (rad)')
87 xlim([0 double(2*Tgiro)])
88 yticks(0:5:20)
89
90 TablaCinematica=table(double(ts)',double(aceleracion(ts))',double...
91 (velocidad(ts))',double(posicion(ts))');
92 TablaCinematica.Properties.VariableNames=["Tiempo","Aceleracion",...
93 "Velocidad","Posicion"]
94 writetable(TablaCinematica,'Perfil_cinematica.txt','delimiter',' ', ...
95 'WriteVariableNames', true)
96 TablaTXT=table(double(ts)',double(velocidad(ts)*180/pi)');
97 writetable(TablaTXT,'Perfil_Velocidad.txt','delimiter',' ', ...
98 'WriteVariableNames', false)

```

E.2. Solución por Newton Raphson Consecutivo

```

1 clear
2 syms t Theta_1(t) Theta_2(t) Theta_3(t) R_1(t)
3 syms r_1 r_1y r_2 r_3 x
4 syms V_r omega_1 omega_2 omega_3 theta_1 theta_2 theta_3
5 syms a_r alpha_1 alpha_2 alpha_3
6
7 Ecuaciones
8 Vr_2=0.075
9 Vr_3=0.17
10 Vr_1y=0.045
11
12 Vtheta_2=0
13 Vomega_2=0
14 Valpha_2=0
15
16 Fc=sin(Theta_1)*R_1-r_1y;
17 Fc=simplify(Fc,'Steps',5)
18
19 fc=subs(Fc,[Theta_1, R_1],[theta_1, r_1]);
20 SolR_1=solve(fc,r_1);
21
22 Fx=-R_1*cos(Theta_1)+r_2*cos(Theta_2)+r_3*cos(Theta_3);
23 Fx=simplify(Fx,'Steps',5)
24
25 fx=formula(subs(Fx,[Theta_1, Theta_2, Theta_3, R_1],[theta_1, theta_2, ...
    theta_3, SolR_1]))
26 fx=simplify(fx,'Steps',5)
27 SolTheta_1=solve(fx,theta_1);
28
29
30 Fy=R_1*sin(Theta_1)-r_2*sin(Theta_2)-r_3*sin(Theta_3);
31 Fy=simplify(Fy,'Steps',5)
32
33 fy=formula(subs(Fy,[Theta_1, Theta_2, Theta_3, R_1],[theta_1, theta_2, ...
    theta_3, SolR_1]))
34
35 SolTheta_3=solve(fy,theta_3);
36 SolTheta_3=SolTheta_3(1)
37
38 DFcDt=diff(Fc);
39 DFDt=diff(Fx);
40 DFyDt=diff(Fy);
41
42 F1=formula(subs(DFcDt,[diff(R_1),diff(Theta_1),diff(Theta_2), ...
    diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[V_r,omega_1,omega_2, ...
    omega_3,theta_1,theta_2,theta_3,r_1]))
43
44 F2=formula(subs(DFyDt,[diff(R_1),diff(Theta_1),diff(Theta_2), ...
    diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[V_r,omega_1,omega_2, ...
    omega_3,theta_1,theta_2,theta_3,r_1]))
45
46 F3=formula(subs(DFcDt,[diff(R_1),diff(Theta_1),diff(Theta_2), ...
    diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[V_r,omega_1,omega_2, ...
    omega_3,theta_1,theta_2,theta_3,r_1]))
47
48
49
50
51
52

```

```

53 D2FcDt2=diff(DFcDt);
54 D2FxDt2=diff(DFxDt);
55 D2FyDt2=diff(DFyDt);
56
57 F4=formula(subs(D2FcDt2,[diff(diff(R_1)),diff(diff(Theta_1)), ...
58     diff(diff(Theta_2)),diff(diff(Theta_3)),diff(R_1),diff(Theta_1), ...
59     diff(Theta_2),diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[a_r, ...
60     alpha_1,alpha_2,alpha_3,V_r,omega_1,omega_2,omega_3,theta_1, ...
61     theta_2,theta_3,r_1]))
62 F5=formula(subs(D2FxDt2,[diff(diff(R_1)),diff(diff(Theta_1)), ...
63     diff(diff(Theta_2)),diff(diff(Theta_3)),diff(R_1),diff(Theta_1), ...
64     diff(Theta_2),diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[a_r, ...
65     alpha_1,alpha_2,alpha_3,V_r,omega_1,omega_2,omega_3,theta_1, ...
66     theta_2,theta_3,r_1]))
67 F6=formula(subs(D2FyDt2,[diff(diff(R_1)),diff(diff(Theta_1)), ...
68     diff(diff(Theta_2)),diff(diff(Theta_3)),diff(R_1),diff(Theta_1), ...
69     diff(Theta_2),diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[a_r, ...
70     alpha_1,alpha_2,alpha_3,V_r,omega_1,omega_2,omega_3,theta_1, ...
71     theta_2,theta_3,r_1]))
72
73 Pi=[-0.1,-0.5,3,-0.5,0,0.5,-0.5,0,0.5];
74 %Pi=[-0.0997,-0.4684,2.8737,0,0,0,0,0,0];
75
76 Posición
77
78 fP(r_1,theta_1,theta_3)=[fc;fx;fy];
79 fP=subs(fP,[theta_2,r_1y,r_2,r_3],[Vtheta_2,Vr_1y,Vr_2,Vr_3]);
80 fP=simplify(fP,'Steps',5);
81
82 valTheta_3=subs(SolTheta_3,[theta_2,r_2,r_3,r_1y],[Vtheta_2,Vr_2,Vr_3,Vr_1y]);
83 ValTheta_1=subs(SolTheta_1,[theta_2,r_2,r_3,r_1y,theta_3],[Vtheta_2,Vr_2, ...
84     Vr_3,Vr_1y,valTheta_3]);
85 valR_1=subs(SolR_1,[r_1y,theta_1],[Vr_1y,ValTheta_1]);
86
87 JacP=jacobian(fP,[r_1,theta_1,theta_3]);
88 JacInvP=JacP^-1;
89
90
91 pP=[Pi(1),Pi(2),Pi(3)]';
92 ValoresPNR(1,:)=pP';
93 iteraciones=15;
94 for i=1:iteraciones
95     pP=double(pP-JacInvP(pP(1),pP(2),pP(3))*fP(pP(1),pP(2),pP(3)));
96     ValoresPNR(i+1,:)=double(pP)';
97     ErrorNRP(i,:)=abs(ValoresPNR(i,:)-ValoresPNR(i+1,:));
98     NormErrorNRP(i)=norm(ErrorNRP(i,:));
99     if NormErrorNRP(i)<0.000001
100         break;
101     end
102 end
103 SolPosNR=double(pP);
104
105 Velocidad
106
107 fV(V_r,omega_1,omega_3)=[F1;F2;F3];
108 fV=subs(fV,[theta_2,r_1y,r_2,r_3,r_1,theta_1,theta_3,omega_2],[Vtheta_2, ...
109     Vr_1y,Vr_2,Vr_3,SolPosNR(1),SolPosNR(2),SolPosNR(3),Vomega_2]);

```

```

110 fV=simplify(fV,'Steps',5);
111
112 JacV=jacobian(fV, [V_r,omega_1,omega_3]);
113 JacInvV=JacV^-1;
114
115 pV=[Pi(4),Pi(5),Pi(6)]';
116 ValoresVNR(1,:)=pV';
117
118 iteraciones=15;
119 for i=1:iteraciones
120     pV=double(pV-JacInvV(pV(1),pV(2),pV(3))*fV(pV(1),pV(2),pV(3)));
121     ValoresVNR(i+1,:)=double(pV)';
122     ErrorNRV(i,:)=abs(ValoresVNR(i,:)-ValoresVNR(i+1,:));
123     NormErrorNRV(i)=norm(ErrorNRV(i,:));
124     if NormErrorNRV(i)<0.000001
125         break;
126     end
127 end
128
129 SolVelNR=double(pV);
130
131 Aceleración
132 fA(a_r,alpha_1,alpha_3)=[F4;F5;F6];
133 fA=subs(fA,[theta_2,r_1y,r_2,r_3,r_1,theta_1,theta_3,omega_2,V_r,omega_1,...
134     omega_3,alpha_2],[Vtheta_2,Vr_1y,Vr_2,Vr_3,SolPosNR(1),SolPosNR(2),...
135     SolPosNR(3),Vomega_2,SolVelNR(1),SolVelNR(2),SolVelNR(3),Valpha_2]);
136 fA=simplify(fA,'Steps',5);
137 JacA=jacobian(fA, [a_r,alpha_1,alpha_3]);
138 JacInvA=JacA^-1;
139
140 PiA=[Pi(7),Pi(8),Pi(9)]';
141 ValoresANR(1,:)=PiA';
142
143 iteraciones=15;
144 for i=1:iteraciones
145     PiA=double(PiA-JacInvA(PiA(1),PiA(2),PiA(3))*fA(PiA(1),PiA(2),PiA(3)));
146     ValoresANR(i+1,:)=double(PiA)';
147     ErrorNRA(i,:)=abs(ValoresANR(i,:)-ValoresANR(i+1,:));
148     NormErrorNRA(i)=norm(ErrorNRA(i,:));
149     if NormErrorNRA(i)<0.000001
150         break;
151     end
152 end
153 SolAceNR=double(PiA);
154
155 Gráficas
156
157 figure()
158 plot(0:(length(ValoresPNR(:,1))-1),ValoresPNR)
159 title('Solucion posicion metodo de Newton Raphson')
160 legend("r_1","\theta_1","\theta_3","Location","best")
161 ylabel('Solucion')
162 xlabel('Indice')
163 xticks(0:length(ValoresPNR(:,1)))
164 grid on
165
166 figure()

```

```

167 subplot(2,1,1)
168 plot(ErrorNRP)
169 title('Error de posicion metodo de Newton Raphson')
170 legend("r_1","\theta_1","\theta_3","Location","best")
171 ylabel('Error')
172 xlabel('Iteraciones')
173 xticks(1:(length(ValoresPNR(:,1))-1))
174 grid on
175 subplot(2,1,2)
176 plot(NormErrorNRP)
177 title('Error total posicion metodo de Newton Raphson')
178 xlabel('Iteraciones')
179 xticks(1:(length(ValoresPNR(:,1))-1))
180 grid on
181
182
183 figure()
184 plot(0:(length(ValoresVNR(:,1))-1),ValoresVNR)
185 title('Solucion velocidad metodo de Newton Raphson')
186 legend("V_r","\omega_1","\omega_3","Location","best")
187 ylabel('Solucion')
188 xlabel('Indice')
189 xticks(0:length(ValoresVNR(:,1)))
190 grid on
191
192
193 figure()
194 subplot(2,1,1)
195 plot(ErrorNRV)
196 title('Error de velocidad metodo de Newton Raphson')
197 legend("V_r","\omega_1","\omega_3","Location","best")
198 ylabel('Error')
199 xlabel('Iteraciones')
200 xticks(1:(length(ValoresVNR(:,1))-1))
201 grid on
202 subplot(2,1,2)
203 plot(NormErrorNRV)
204 title('Error total velocidad metodo de Newton Raphson')
205 xlabel('Iteraciones')
206 xticks(1:(length(ValoresVNR(:,1))-1))
207 grid on
208
209 figure()
210 plot(0:(length(ValoresANR(:,1))-1),ValoresANR)
211 title('Solucion aceleracion metodo de Newton Raphson')
212 legend("a_r","\alpha_1","\alpha_3","Location","best")
213 ylabel('Solucion')
214 xlabel('Indice')
215 xticks(0:length(ValoresANR(:,1)))
216 grid on
217
218 figure()
219 subplot(2,1,1)
220 plot(ErrorNRA)
221 title('Error de aceleracion metodo de Newton Raphson')
222 legend("a_r","\alpha_1","\alpha_3","Location","best")
223 ylabel('Error')
224 xlabel('Iteraciones')

```

```
225 xticks(1:(length(ValoresANR(:,1))-1))
226 grid on
227 subplot(2,1,2)
228 plot(NormErrorNRA)
229 title('Error total aceleracion metodo de Newton Raphson')
230 xlabel('Iteraciones')
231 xticks(1:(length(ValoresANR(:,1))-1))
232 grid on
233
234 format long
235 Solucion=[SolPosNR;SolVelNR;SolAceNR]
```

E.3. Solución por Newton Raphson simultaneo

```

1 clear
2 clc
3 syms t Theta_1(t) Theta_2(t) Theta_3(t) R_1(t)
4
5
6 syms r_1 r_1y r_2 r_3 x
7
8 syms V_r omega_1 omega_2 omega_3 theta_1 theta_2 theta_3
9
10 syms a_r alpha_1 alpha_2 alpha_3
11
12 Vr_2=0.075
13 Vr_3=0.17
14 Vr_1y=0.045
15
16 Vtheta_2=0
17 Vomega_2=0
18 Valpha_2=0
19
20 Fc=sin(Theta_1)*R_1-r_1y;
21 Fc=simplify(Fc, 'Steps',5)
22 fc=subs(Fc,[Theta_1, R_1],[theta_1, r_1])
23 SolR_1=solve(fc,r_1)
24 Fx=-R_1*cos(Theta_1)+r_2*cos(Theta_2)+r_3*cos(Theta_3);
25 Fx=simplify(Fx, 'Steps',5)
26 fx=formula(subs(Fx,[Theta_1, Theta_2, Theta_3, R_1],[theta_1, theta_2, ...
    theta_3, SolR_1]))
27 fx=simplify(fx, 'Steps',5)
28 SolTheta_1=solve(fx,theta_1);
29 Fy=R_1*sin(Theta_1)-r_2*sin(Theta_2)-r_3*sin(Theta_3);
30 Fy=simplify(Fy, 'Steps',5);
31 fy=formula(subs(Fy,[Theta_1, Theta_2, Theta_3, R_1],[theta_1, theta_2, ...
    theta_3, SolR_1]))
32 SolTheta_3=solve(fy,theta_3);
33 SolTheta_3=SolTheta_3(1);
34
35 DFcDt=diff(Fc);
36 DFxDt=diff(Fx);
37 DFyDt=diff(Fy);
38
39 F1=formula(subs(DFxDt,[diff(R_1),diff(Theta_1),diff(Theta_2), ...
    diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[V_r,omega_1,omega_2, ...
    omega_3,theta_1,theta_2,theta_3,r_1]))
40
41 F2=formula(subs(DFyDt,[diff(R_1),diff(Theta_1),diff(Theta_2), ...
    diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[V_r,omega_1,omega_2, ...
    omega_3,theta_1,theta_2,theta_3,r_1]))
42
43 F3=formula(subs(DFcDt,[diff(R_1),diff(Theta_1),diff(Theta_2), ...
    diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[V_r,omega_1,omega_2, ...
    omega_3,theta_1,theta_2,theta_3,r_1]))
44
45
46
47
48
49
50 D2FcDt2=diff(DFcDt);
51 D2FxDt2=diff(DFxDt);
52 D2FyDt2=diff(DFyDt);
53

```

```

54 F4=formula(subs(D2FcDt2,[diff(diff(R_1)),diff(diff(Theta_1)), ...
55     diff(diff(Theta_2)),diff(diff(Theta_3)),diff(R_1),diff(Theta_1),
56     diff(Theta_2),diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[a_r, ...
57     alpha_1,alpha_2,alpha_3,V_r,omega_1,omega_2,omega_3,theta_1,theta_2, ...
58     theta_3,r_1]))
59 F5=formula(subs(D2FxDt2,[diff(diff(R_1)),diff(diff(Theta_1)), ...
60     diff(diff(Theta_2)),diff(diff(Theta_3)),diff(R_1),diff(Theta_1), ...
61     diff(Theta_2),diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[a_r, ...
62     alpha_1,alpha_2,alpha_3,V_r,omega_1,omega_2,omega_3,theta_1,theta_2, ...
63     theta_3,r_1]))
64 F6=formula(subs(D2FyDt2,[diff(diff(R_1)),diff(diff(Theta_1)), ...
65     diff(diff(Theta_2)),diff(diff(Theta_3)),diff(R_1),diff(Theta_1), ...
66     diff(Theta_2),diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[a_r, ...
67     alpha_1,alpha_2,alpha_3,V_r,omega_1,omega_2,omega_3,theta_1,theta_2, ...
68     theta_3,r_1]))
69
70
71 f(r_1,theta_1,theta_3,V_r,omega_1,omega_3,a_r,alpha_1,alpha_3)=[fc;fx;fy;
72     F1;F2;F3;F4;F5;F6]
73 f=subs(f,[theta_2,r_1y,r_2,r_3,omega_2,alpha_2],[Vtheta_2,Vr_1y,Vr_2, ...
74     Vr_3,Vomega_2,Valpha_2]);
75 f=simplify(f,'Steps',5);
76
77 Jac=jacobian(f, ...
78     [r_1,theta_1,theta_3,V_r,omega_1,omega_3,a_r,alpha_1,alpha_3])
79 JacInv=Jac^-1
80
81
82
83 p1=[-0.1,-0.5,3,-0.5,0,0.5,-0.5,0,0.5]';
84 %p1=[-0.0997,-0.4684,2.8737,0,0,0,0,0,0]';
85
86 ValoresNR(1,:)=p1';
87
88 iteraciones=15;
89 for i=1:iteraciones
90     p1=double(p1-JacInv(p1(1),p1(2),p1(3),p1(4),p1(5),p1(6),p1(7),p1(8), ...
91     p1(9))*f(p1(1),p1(2),p1(3),p1(4),p1(5),p1(6),p1(7),p1(8),p1(9)))
92     ValoresNR(i+1,:)=double(p1)';
93     ErrorNR(i,:)=abs(ValoresNR(i,:)-ValoresNR(i+1,:));
94     NormErrorNR(i)=norm(ErrorNR(i,:));
95     if NormErrorNR(i)<0.000001
96         break;
97     end
98 end
99 figure()
100 plot(0:(length(ValoresNR(:,1))-1),ValoresNR)
101 title('Solucion metodo conjunto de Newton Raphson')
102 legend("r_1","\theta_1","\theta_3","V_r","\omega_1","\omega_3","a_r", ...
103     "\alpha_1","\alpha_3","Location","best")
104
105 ylabel('Solucion')
106 xlabel('Indice')
107 xticks(0:length(ValoresNR))
108 grid on
109 figure()
110 subplot(2,1,1)

```



```
111 plot(ErrorNR)
112 title('Error absoluto metodo conjunto de Newton Raphson')
113 legend("r_1","\theta_1","\theta_3","V_r","\omega_1","\omega_3","a_r", ...
114         "\alpha_1","\alpha_3","Location","east")
115 ylabel('Error')
116 xlabel('Iteraciones')
117 xticks(1:(length(ValoresNR)-1))
118 grid on
119 subplot(2,1,2)
120 plot(NormErrorNR)
121 title('Error total metodo conjunto de Newton Raphson')
122 xlabel('Iteraciones')
123 xticks(1:(length(ValoresNR)-1))
124 grid on
125
126
127
128 format long
129 SolNR=double(p1)
130 ErrorNRFinal=double(NormErrorNR(end))
```

E.4. Solución por método de Newton Raphson

```

1 clear
2 clc
3 Desarrollo teórico
4 syms Theta_1(t) Theta_2(t) Theta_3(t) R_1(t) t
5 syms r_1 r_1y r_2 r_3
6 syms theta_1 theta_2 theta_3
7 syms V_r omega_1 omega_2 omega_3
8 syms a_r alpha_1 alpha_2 alpha_3
9
10 Vr_2=0.075
11 Vr_3=0.17
12 Vr_1y=0.045
13
14 Fc=sin(Theta_1)*R_1-r_1y;
15 fc=subs(Fc,[Theta_1, R_1],[theta_1, r_1]);
16
17 Fx=-R_1*cos(Theta_1)+r_2*cos(Theta_2)+r_3*cos(Theta_3);
18 Fx=simplify(Fx,'Steps',5);
19 fx=formula(subs(Fx,[Theta_1, Theta_2, Theta_3, R_1],[theta_1, theta_2, ...
    theta_3, r_1]));
20
21 Fy=R_1*sin(Theta_1)-r_2*sin(Theta_2)-r_3*sin(Theta_3);
22 Fy=simplify(Fy,'Steps',5);
23 fy=formula(subs(Fy,[Theta_1, Theta_2, Theta_3, R_1],[theta_1, theta_2, ...
    theta_3, r_1]));
24
25 DFcDt=diff(Fc);
26 DFxDt=diff(Fx);
27 DFyDt=diff(Fy);
28 F1=formula(subs(DFxDt,[diff(R_1),diff(Theta_1),diff(Theta_2),diff ...
29     (Theta_3),Theta_1,Theta_2,Theta_3,R_1],[V_r,omega_1,omega_2, ...
30     omega_3,theta_1,theta_2,theta_3,r_1]));
31 F2=formula(subs(DFyDt,[diff(R_1),diff(Theta_1),diff(Theta_2),diff ...
32     (Theta_3),Theta_1,Theta_2,Theta_3,R_1],[V_r,omega_1,omega_2, ...
33     omega_3,theta_1,theta_2,theta_3,r_1]));
34 F3=formula(subs(DFcDt,[diff(R_1),diff(Theta_1),diff(Theta_2),diff ...
35     (Theta_3),Theta_1,Theta_2,Theta_3,R_1],[V_r,omega_1,omega_2, ...
36     omega_3,theta_1,theta_2,theta_3,r_1]));
37
38 D2FcDt2=diff(DFcDt);
39 D2FxDt2=diff(DFxDt);
40 D2FyDt2=diff(DFyDt);
41 F4=formula(subs(D2FcDt2,[diff(diff(R_1)),diff(diff(Theta_1)), ...
42     diff(diff(Theta_2)),diff(diff(Theta_3)),diff(R_1),diff(Theta_1), ...
43     diff(Theta_2),diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1], ...
44     [a_r,alpha_1,alpha_2,alpha_3,V_r,omega_1,omega_2,omega_3,theta_1, ...
45     theta_2,theta_3,r_1]));
46 F5=formula(subs(D2FxDt2,[diff(diff(R_1)),diff(diff(Theta_1)), ...
47     diff(diff(Theta_2)),diff(diff(Theta_3)),diff(R_1),diff(Theta_1), ...
48     diff(Theta_2),diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1], ...
49     [a_r,alpha_1,alpha_2,alpha_3,V_r,omega_1,omega_2,omega_3,theta_1, ...
50     theta_2,theta_3,r_1]));
51 F6=formula(subs(D2FyDt2,[diff(diff(R_1)),diff(diff(Theta_1)), ...
52     diff(diff(Theta_2)),diff(diff(Theta_3)),diff(R_1),diff(Theta_1), ...

```

```

53     diff(Theta_2), diff(Theta_3), Theta_1, Theta_2, Theta_3, R_1], ...
54     [a_r, alpha_1, alpha_2, alpha_3, V_r, omega_1, omega_2, omega_3, theta_1, ...
55     theta_2, theta_3, r_1])));
56
57 Tabla_Cinematica=readtable("Perfil_cinematica.txt", "Delimiter", " ");
58
59 Solución NR Simultaneo
60 f(r_1, theta_1, theta_3, V_r, omega_1, omega_3, a_r, alpha_1, alpha_3)=
61 [fc; fx; fy; F1; F2; F3; F4; F5; F6];
62 f=subs(f, [r_1y, r_2, r_3], [Vr_1y, Vr_2, Vr_3]);
63 f=simplify(f, 'Steps', 5);
64 JacInv=jacobian(f, [r_1, theta_1, theta_3, V_r, omega_1, omega_3, a_r, ...
65     alpha_1, alpha_3])^-1;
66
67
68 p1=[-0.0997, -0.4684, 2.8737, 0, 0, 0, 0, 0, 0];
69
70 for i=1:length(Tabla_Cinematica.Tiempo)
71     tic
72     [p1, error_NR(i)]=NR_Sistema(f, JacInv, p1, Tabla_Cinematica.Posicion(i), ...
73     Tabla_Cinematica.Velocidad(i), Tabla_Cinematica.Aceleracion(i));
74     ValoresSolucion_NR(i, :)=p1;
75     tiepoEjeNR(i)=toc;
76 end
77
78 for i=1:length(tiepoEjeNR)
79     tiempoTotalNr(i)=sum(tiepoEjeNR(1:i));
80 end
81
82 Tabla_Cinematica.R_1=ValoresSolucion_NR(:, 1);
83 Tabla_Cinematica.theta_1=ValoresSolucion_NR(:, 2);
84 Tabla_Cinematica.theta_3=ValoresSolucion_NR(:, 3);
85 Tabla_Cinematica.V_1=ValoresSolucion_NR(:, 4);
86 Tabla_Cinematica.omega_1=ValoresSolucion_NR(:, 5);
87 Tabla_Cinematica.omega_3=ValoresSolucion_NR(:, 6);
88 Tabla_Cinematica.a_1=ValoresSolucion_NR(:, 7);
89 Tabla_Cinematica.alpha_1=ValoresSolucion_NR(:, 8);
90 Tabla_Cinematica.alpha_3=ValoresSolucion_NR(:, 9);
91 writetable(Tabla_Cinematica, 'Resultados.txt', 'delimiter', ' ', ...
92     'WriteVariableNames', true)
93
94 Solución NR Consecutivo
95 p1=[-0.0997, -0.4684, 2.8737, 0, 0, 0, 0, 0, 0];
96 fP(r_1, theta_1, theta_3)=[fc; fx; fy];
97 fP=subs(fP, [r_1y, r_2, r_3], [Vr_1y, Vr_2, Vr_3]);
98 fP=simplify(fP, 'Steps', 5);
99 JacP=jacobian(fP, [r_1, theta_1, theta_3]);
100 JacInvP=JacP^-1;
101 fV(V_r, omega_1, omega_3)=[F1; F2; F3];
102 fV=subs(fV, [r_1y, r_2, r_3], [Vr_1y, Vr_2, Vr_3]);
103 fV=simplify(fV, 'Steps', 5);
104 JacV=jacobian(fV, [V_r, omega_1, omega_3]);
105 JacInvV=JacV^-1;
106 fA(a_r, alpha_1, alpha_3)=[F4; F5; F6];
107 fA=subs(fA, [r_1y, r_2, r_3], [Vr_1y, Vr_2, Vr_3]);
108 fA=simplify(fA, 'Steps', 5);
109 JacA=jacobian(fA, [a_r, alpha_1, alpha_3]);

```

```

109 JacInvA=JacA^-1;
110
111 for i=1:length(Tabla_Cinematica.Tiempo)
112     tic
113     [p1,Error_NR_EX(i)]= ...
114         NR_Sistema_extenso(fP,JacInvP,fV,JacInvV,fA,JacInvA,p1, ...
115             Tabla_Cinematica.Posicion(i),Tabla_Cinematica.Velocidad(i), ...
116             Tabla_Cinematica.Aceleracion(i));
117     ValoresSolucion_NR_EX(i,:)=p1;
118     tiempoEje_EX(i)=toc;
119 end
120
121 tabla_Cinematica_NR_Ext=array2table([Tabla_Cinematica.Tiempo, ...
122     Tabla_Cinematica.Aceleracion,Tabla_Cinematica.Velocidad, ...
123     Tabla_Cinematica.Posicion,ValoresSolucion_NR_EX]);
124 tabla_Cinematica_NR_Ext.Properties.VariableNames=
125     Tabla_Cinematica.Properties.VariableNames()
126 writetable(tabla_Cinematica_NR_Ext,'Resultados_NR_Extendido.txt', ...
127     'delimiter',' ','WriteVariableNames',true)
128
129 for i=1:length(tiempoEje_EX)
130     tiempoTotalNrEX(i)=sum(tiempoEje_EX(1:i));
131 end
132
133 Tiempo_NR=array2table([tiempoTotalNr',tiempoTotalNrEX',tiempoEjeNR', ...
134     tiempoEje_EX']);
135 Tiempo_NR.Properties.VariableNames=["Tiempo_total_NR","Tiempo_total_NR_EX",
136     "tiempo_puntos_NR","tiempo_puntos_NR_EX"]
137 writetable(Tiempo_NR,'Tiempor_NR.txt','delimiter',' ',' ...
138     'WriteVariableNames', ...
139     true)
140
141 Error_NR=array2table([error_NR',Error_NR_EX']);
142 Error_NR.Properties.VariableNames=["Error_NR","Error_NR_EX"]
143 writetable(Error_NR,'Error_NR.txt','delimiter',' ','WriteVariableNames', ...
144     true)
145

```

Gráficas de resultados

Tabla de resultados

```

146
147 TablaError=readtable("Error_NR.txt");
148 Tabla_Cinematica=readtable("Resultados.txt","Delimiter"," ");
149 tabla_Cinematica_NR_Ext=readtable('Resultados_NR_Extendido.txt', ...
150     'delimiter',' ');
151 tiempo=Tabla_Cinematica.Tiempo;
152

```

NR simultaneo

```

153
154 fprintf("Graficas de resultados del metodo de NR simultaneo")
155 figure()
156 subplot(3,1,1)
157 plot(tiempo,Tabla_Cinematica.R_1,"r")
158 title("r_1 vs tiempo")
159 xlabel("t (s)")
160 ylabel("r_1 (m)")
161 grid on
162 xlim([0 tiempo(end)])
163

```

```

164 subplot(3,1,2)
165 plot(tiempo, Tabla_Cinematica.theta_1*180/pi, "b")
166 title("\theta_1 vs tiempo")
167 xlabel("t (s)")
168 ylabel("\theta_1 (\circ)")
169 grid on
170 xlim([0 tiempo(end)])
171 yticks(-30:5:-10)
172
173 subplot(3,1,3)
174 plot(tiempo, Tabla_Cinematica.theta_3*180/pi, "k")
175 title("\theta_3 vs tiempo")
176 xlabel("t (s)")
177 ylabel("\theta_3 (\circ)")
178 grid on
179 xlim([0 tiempo(end)])
180 ylim([125 200])
181 yticks(125:25:200)
182
183 figure()
184 subplot(3,1,1)
185 plot(tiempo, Tabla_Cinematica.V_1, "r")
186 title("v_1 vs tiempo")
187 xlabel("t (s)")
188 ylabel("v_1 (m s^{-1})")
189 grid on
190 xlim([0 tiempo(end)])
191 ylim([-0.6 0.6])
192 yticks(-0.6:0.3:0.6)
193
194 subplot(3,1,2)
195 plot(tiempo, Tabla_Cinematica.omega_1*180/pi, "b")
196 title("\omega_1 vs tiempo")
197 xlabel("t (s)")
198 ylabel("\omega_1 (\circ s^{-1})")
199 grid on
200 xlim([0 tiempo(end)])
201 ylim([-80 80])
202 yticks(-80:40:80)
203
204
205 subplot(3,1,3)
206 plot(tiempo, Tabla_Cinematica.omega_3*180/pi, "k")
207 title("\omega_3 vs tiempo")
208 xlabel("t (s)")
209 ylabel("\omega_3 (\circ s^{-1})")
210 grid on
211 xlim([0 tiempo(end)])
212 yticks([-200:100:200])
213
214 figure()
215 subplot(3,1,1)
216 plot(tiempo, Tabla_Cinematica.a_1, "r")
217 title("a_1 vs tiempo")
218 xlabel("t (s)")
219 ylabel("a_1 (m s^{-2})")
220 grid on
221 xlim([0 tiempo(end)])

```

```

222 yticks(-5:2.5:5)
223
224 subplot(3,1,2)
225 plot(tiempo,Tabla_Cinematica.alpha_1*180/pi,"b")
226 title("\alpha_1 vs tiempo")
227 xlabel("t (s)")
228 ylabel("\alpha_1 (\circ s^{-2})")
229 grid on
230 xlim([0 tiempo(end)])
231 ylim([-400 800])
232 yticks([-400:400:800])
233
234 subplot(3,1,3)
235 plot(tiempo,Tabla_Cinematica.alpha_3*180/pi,"k")
236 title("\alpha_3 vs tiempo")
237 xlabel("t (s)")
238 ylabel("\alpha_3 (\circ s^{-2})")
239 grid on
240 xlim([0 tiempo(end)])
241 ylim([-1500 1500])
242 yticks([-1500:750:1500])
243
244
245 figure()
246 subplot(2,1,1)
247 plot(tiempo,TablaError.Error_NR)
248 title("Error del metodo por Newton Raphson simultaneo")
249 xlim([0 tiempo(end)])
250 grid on
251
252 NR consecutivo
253 fprintf("Graficas de resultados del metodo de NR consecutivo")
254 figure()
255 subplot(3,1,1)
256 plot(tiempo,tabla_Cinematica_NR_Ext.R_1,"r--")
257 title("r_1 vs tiempo")
258 xlabel("t (s)")
259 ylabel("r_1 (m)")
260 grid on
261 xlim([0 tiempo(end)])
262
263 subplot(3,1,2)
264 plot(tiempo,tabla_Cinematica_NR_Ext.theta_1*180/pi,"b--")
265 title("\theta_1 vs tiempo")
266 xlabel("t (s)")
267 ylabel("\theta_1 (\circ)")
268 grid on
269 xlim([0 tiempo(end)])
270 yticks(-30:5:-10)
271
272 subplot(3,1,3)
273 plot(tiempo,tabla_Cinematica_NR_Ext.theta_3*180/pi,"k--")
274 title("\theta_3 vs tiempo")
275 xlabel("t (s)")
276 ylabel("\theta_3 (\circ)")
277 grid on
278 xlim([0 tiempo(end)])

```

```

279 ylim([125 200])
280 yticks(125:25:200)
281
282 figure()
283 subplot(3,1,1)
284 plot(tiempo,tabla_Cinematica_NR_Ext.V_1,"r--")
285 title("v_1 vs tiempo")
286 xlabel("t (s)")
287 ylabel("v_1 (m s^{-1})")
288 grid on
289 xlim([0 tiempo(end)])
290 ylim([-0.6 0.6])
291 yticks(-0.6:0.3:0.6)
292
293 subplot(3,1,2)
294 plot(tiempo,tabla_Cinematica_NR_Ext.omega_1*180/pi,"b--")
295 title("\omega_1 vs tiempo")
296 xlabel("t (s)")
297 ylabel("\omega_1 (\circ s^{-1})")
298 grid on
299 xlim([0 tiempo(end)])
300 ylim([-80 80])
301 yticks(-80:40:80)
302
303
304 subplot(3,1,3)
305 plot(tiempo,tabla_Cinematica_NR_Ext.omega_3*180/pi,"k--")
306 title("\omega_3 vs tiempo")
307 xlabel("t (s)")
308 ylabel("\omega_3 (\circ s^{-1})")
309 grid on
310 xlim([0 tiempo(end)])
311 yticks([-200:100:200])
312
313 figure()
314 subplot(3,1,1)
315 plot(tiempo,tabla_Cinematica_NR_Ext.a_1,"r--")
316 title("a_1 vs tiempo")
317 xlabel("t (s)")
318 ylabel("a_1 (m s^{-2})")
319 grid on
320 xlim([0 tiempo(end)])
321 yticks(-5:2.5:5)
322
323 subplot(3,1,2)
324 plot(tiempo,tabla_Cinematica_NR_Ext.alpha_1*180/pi,"b--")
325 title("\alpha_1 vs tiempo")
326 xlabel("t (s)")
327 ylabel("\alpha_1 (\circ s^{-2})")
328 grid on
329 xlim([0 tiempo(end)])
330 ylim([-400 800])
331 yticks([-400:400:800])
332
333 subplot(3,1,3)
334 plot(tiempo,tabla_Cinematica_NR_Ext.alpha_3*180/pi,"k--")
335 title("\alpha_3 vs tiempo")
336 xlabel("t (s)")

```

```

337 ylabel("\alpha_3 (\circ s^{-2})")
338 grid on
339 xlim([0 tiempo(end)])
340 ylim([-1500 1500])
341 yticks(-1500:750:1500)
342
343 figure()
344 subplot(2,1,1)
345 plot(tiempo, TablaError.Error_NR_EX)
346 title("Error del metodo por Newton Raphson consecutivo")
347 xlim([0 tiempo(end)])
348 grid on
349
350

```

Comparación entre ambas versiones de Newton Raphson

Error Resultados

```

353 ValoresSolucion_NR=table2array(Tabla_Cinematica);
354 ValoresSolucion_NR=ValoresSolucion_NR(:, (2:end));
355 ValoresSolucion_NR_EX=table2array(tabla_Cinematica_NR_Ext);
356 ValoresSolucion_NR_EX=ValoresSolucion_NR_EX(:, (2:end));
357
358
359 for i=1:length(ValoresSolucion_NR)
360     Error_M(i)=norm(abs(ValoresSolucion_NR(i,:) - ValoresSolucion_NR_EX(i,:)));
361 end
362 figure()
363 subplot(2,1,1)
364 plot(Tabla_Cinematica.Tiempo, Error_M)
365 writematrix(Error_M, "Error_entre_NR")
366 xlim([0 max(Tabla_Cinematica.Tiempo)])
367 title("Error entre ambos metodos NR")
368 xlabel("tiempo (s)")
369 ylabel("Error")
370 grid on
371

```

Tiempo de ejecución

```

372 tablaTiempo=readtable("Tiempor_NR.txt");
373 tiempoTotalNr=tablaTiempo.Tiempo_total_NR;
374 tiempoTotalNrEX=tablaTiempo.Tiempo_total_NR_EX;
375 figure()
376 plot(tiempoTotalNr)
377 hold on
378 plot(tiempoTotalNrEX)
379 hold off
380 title("Tiempo de ejecucion vs Numero de puntos")
381 legend("NR Simultaneo", "NR Consecutivo", "Location", "best")
382 grid on
383 ylabel("t (s)")
384 xlabel("Numero de puntos")
385 Diferencia=abs(tiempoTotalNr-tiempoTotalNrEX);
386 DiferenciaRelativa=Diferencia./tiempoTotalNrEX*100;
387 figure()
388 subplot(2,1,1)
389 plot(Diferencia)
390 title("Diferencia temporal entre metodos")
391 xlabel("Numero de puntos")
392 ylabel("t (s)")
393

```



```

394 yticks(0:20:100)
395 grid on
396 subplot(2,1,2)
397 plot(DiferenciaRelativa)
398 title("Porcentaje de optimizacion")
399 ylabel("Porcentaje (%)")
400 xlabel("Numero de puntos")
401 ylim([0 30])
402 yticks(0:6:30)
403 grid on
404
405 function [Sol,Error]= NR_Sistema(F,FJacInv,Pi,Ftheta_2,Fomega_2,Falpha_2)
406 syms omega_2 theta_2 alpha_2
407 f=subs(F,[theta_2,omega_2,alpha_2],[Ftheta_2,Fomega_2,Falpha_2]);
408 JacInv=subs(FJacInv,[theta_2,omega_2,alpha_2],[Ftheta_2,Fomega_2,Falpha_2]);
409 p=Pi';
410 ValoresNR(1,:)=p';
411 iteraciones=50;
412 for i=1:iteraciones
413     p=double(p-JacInv(p(1),p(2),p(3),p(4),p(5),p(6),p(7),p(8),p(9))*f(p(1), ...
414         ...
415         p(2),p(3),p(4),p(5),p(6),p(7),p(8),p(9)));
416     ValoresNR(i+1,:)=double(p)';
417     ErrorNR(i,:)=abs(ValoresNR(i,:)-ValoresNR(i+1,:));
418     NormErrorNR(i)=norm(ErrorNR(i,:));
419     if NormErrorNR(i)<0.000001
420         break;
421     end
422 end
423 Error=NormErrorNR(end);
424 Sol=p';
425
426
427 function [Sol,Error]= ...
428     NR_Sistema_extenso(fP,JacInvP,fV,JacInvV,fA,JacInvA,Pi,Vtheta_2, ...
429     Vomega_2,Valpha_2)
430 syms r_1y r_1 r_2 r_3 theta_3 theta_2 theta_1 V_r omega_1 omega_2 ...
431     omega_3 a_r alpha_1 alpha_2 alpha_3
432 fP=subs(fP,theta_2,Vtheta_2);
433 PiP=[Pi(1),Pi(2),Pi(3)]';
434 ValoresNR(1,:)=PiP';
435 iteraciones=50;
436 for i=1:iteraciones
437     PiP=double(PiP-JacInvP(PiP(1),PiP(2),PiP(3))*fP(PiP(1), ...
438         PiP(2),PiP(3)));
439     ValoresNR(i+1,:)=double(PiP)';
440     ErrorNR_P(i,:)=abs(ValoresNR(i,:)-ValoresNR(i+1,:));
441     NormErrorNR_P(i)=norm(ErrorNR_P(i,:));
442     if NormErrorNR_P(i)<0.000001
443         break;
444     end
445 end
446 SolPosNR=PiP;
447 ErrorP=NormErrorNR_P(end);
448 fV=subs(fV,[theta_2,r_1,theta_1,theta_3,omega_2],[Vtheta_2,SolPosNR(1), ...
449     SolPosNR(2),SolPosNR(3),Vomega_2]);
450 JacInvV=subs(JacInvV,[theta_2,r_1,theta_1,theta_3,omega_2],[Vtheta_2, ...

```

```

449     SolPosNR(1), SolPosNR(2), SolPosNR(3), Vomega_2]);
450 PiV=[Pi(4), Pi(5), Pi(6)]';
451 ValoresVNR(1,:)=PiV';
452 iteraciones=50;
453 for i=1:iteraciones
454     PiV=double(PiV-JacInvV(PiV(1), PiV(2), PiV(3))*fV(PiV(1), PiV(2), PiV(3)));
455     ValoresVNR(i+1,:)=double(PiV)';
456     ErrorNR_V(i,:)=abs(ValoresVNR(i,:)-ValoresVNR(i+1,:));
457     NormErrorNR_V(i)=norm(ErrorNR_V(i,:));
458     if NormErrorNR_V(i)<0.000001
459         break;
460     end
461 end
462 SolVelNR=double(PiV);
463 ErrorV=NormErrorNR_V(end);
464 fA=subs(fA, [theta_2, r_1, theta_1, theta_3, omega_2, V_r, omega_1, omega_3, ...
465     alpha_2], [Vtheta_2, SolPosNR(1), SolPosNR(2), SolPosNR(3), Vomega_2, ...
466     SolVelNR(1), SolVelNR(2), SolVelNR(3), Valpha_2]);
467 JacInvA=subs(JacInvA, [theta_2, r_1, theta_1, theta_3, omega_2, V_r, omega_1, ...
468     omega_3, alpha_2], [Vtheta_2, SolPosNR(1), SolPosNR(2), SolPosNR(3), ...
469     Vomega_2, SolVelNR(1), SolVelNR(2), SolVelNR(3), Valpha_2]);
470 PiA=[Pi(7), Pi(8), Pi(9)]';
471 ValoresVNR(1,:)=PiA';
472 iteraciones=50;
473 for i=1:iteraciones
474     PiA=double(PiA-JacInvA(PiA(1), PiA(2), PiA(3))*fA(PiA(1), PiA(2), PiA(3)));
475     ValoresVNR(i+1,:)=double(PiA)';
476     ErrorNR_A(i,:)=abs(ValoresVNR(i,:)-ValoresVNR(i+1,:));
477     NormErrorNR_A(i)=norm(ErrorNR_A(i,:));
478     if NormErrorNR_A(i)<0.000001
479         break;
480     end
481 end
482 SolAceNR=double(PiA);
483 ErrorA=NormErrorNR_A(end);
484 Sol=[SolPosNR', SolVelNR', SolAceNR'];
485 Error=norm([ErrorP, ErrorV, ErrorA]);
486 end

```

E.5. Solución analítica

```

1 clear
2 clc
3
4 syms Theta_1(t) Theta_2(t) Theta_3(t) R_1(t) t
5 syms r_1 r_1y r_2 r_3
6 syms theta_1 theta_2 theta_3
7 syms V_r omega_1 omega_2 omega_3
8 syms a_r alpha_1 alpha_2 alpha_3
9
10 Vr_2=0.075
11 Vr_3=0.17
12 Vr_1y=0.045
13 Vtheta_2=0
14 Vomega_2=0
15 Valpha_2=0
16
17 Fc=sin(Theta_1)*R_1-r_1y;
18 fc=formula(subs(Fc,[Theta_1, R_1],[theta_1, r_1]));
19
20 Fx=-R_1*cos(Theta_1)+r_2*cos(Theta_2)+r_3*cos(Theta_3);
21 Fx=simplify(Fx,'Steps',5);
22 fx=formula(subs(Fx,[Theta_1, Theta_2, Theta_3, R_1],[theta_1, theta_2, ...
    theta_3, r_1]));
23
24 Fy=R_1*sin(Theta_1)-r_2*sin(Theta_2)-r_3*sin(Theta_3);
25 Fy=simplify(Fy,'Steps',5);
26 fy=formula(subs(Fy,[Theta_1, Theta_2, Theta_3, R_1],[theta_1, theta_2, ...
    theta_3, r_1]));
27
28 DFcDt=diff(Fc);
29 DFxDt=diff(Fx);
30 DFyDt=diff(Fy);
31 F1=formula(subs(DFxDt,[diff(R_1),diff(Theta_1),diff(Theta_2), ...
    diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[V_r,omega_1,omega_2, ...
    omega_3,theta_1,theta_2,theta_3,r_1]));
32
33 F2=formula(subs(DFyDt,[diff(R_1),diff(Theta_1),diff(Theta_2), ...
    diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[V_r,omega_1,omega_2, ...
    omega_3,theta_1,theta_2,theta_3,r_1]));
34
35 F3=formula(subs(DFcDt,[diff(R_1),diff(Theta_1),diff(Theta_2), ...
    diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[V_r,omega_1,omega_2, ...
    omega_3,theta_1,theta_2,theta_3,r_1]));
36
37
38
39
40
41 D2FcDt2=diff(DFcDt);
42 D2FxDt2=diff(DFxDt);
43 D2FyDt2=diff(DFyDt);
44 F4=formula(subs(D2FcDt2,[diff(diff(R_1)),diff(diff(Theta_1)), ...
    diff(diff(Theta_2)),diff(diff(Theta_3)),diff(R_1),diff(Theta_1), ...
    diff(Theta_2),diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[a_r, ...
    alpha_1,alpha_2,alpha_3,V_r,omega_1,omega_2,omega_3,theta_1, ...
    theta_2,theta_3,r_1]));
45
46
47
48
49 F5=formula(subs(D2FxDt2,[diff(diff(R_1)),diff(diff(Theta_1)), ...
    diff(diff(Theta_2)),diff(diff(Theta_3)),diff(R_1),diff(Theta_1), ...
    diff(Theta_2),diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[a_r, ...
    alpha_1,alpha_2,alpha_3,V_r,omega_1,omega_2,omega_3,theta_1, ...
    theta_2,theta_3,r_1]));
50
51
52
53

```

```

54 F6=formula(subs(D2FyDt2,[diff(diff(R_1)),diff(diff(Theta_1)), ...
55     diff(diff(Theta_2)),diff(diff(Theta_3)),diff(R_1),diff(Theta_1), ...
56     diff(Theta_2),diff(Theta_3),Theta_1,Theta_2,Theta_3,R_1],[a_r, ...
57     alpha_1,alpha_2,alpha_3,V_r,omega_1,omega_2,omega_3,theta_1, ...
58     theta_2,theta_3,r_1]));
59
60
61 SolR_1=solve(fc,r_1)
62 SolTheta_1=solve(simplify(subs(fx,r_1,SolR_1),"Steps",15),theta_1)
63 SolTheta_3=solve(fy-fc,theta_3);
64 SolTheta_3=SolTheta_3(2)
65
66
67 SolOmega_3=solve(F3-F2,omega_3)
68
69 SolOmega_1=solve(F3,omega_1);
70 SolOmega_1=simplify(SolOmega_1,"Steps",10)
71
72 SolV_r=solve(subs(F1,omega_1,SolOmega_1),V_r);
73 SolV_r=simplify(SolV_r,"Steps",10)
74
75 SolAlpha_3=solve(F6-F4,alpha_3);
76 SolAlpha_3=simplify(SolAlpha_3,"Steps",20)
77
78 SolAlpha_1=solve(F4,alpha_1);
79 SolAlpha_1=simplify(SolAlpha_1,"Steps",20)
80
81 SolA_r=solve(subs(F5,alpha_1,SolAlpha_1),a_r);
82 SolA_r=simplify(SolA_r,"Steps",20)
83
84
85 SolR_1=subs(SolR_1,[r_1y,r_2,r_3],[Vr_1y,Vr_2,Vr_3]);
86 SolTheta_1=subs(SolTheta_1,[r_1y,r_2,r_3],[Vr_1y,Vr_2,Vr_3]);
87 SolTheta_3=subs(SolTheta_3,[r_1y,r_2,r_3],[Vr_1y,Vr_2,Vr_3]);
88 SolV_r=subs(SolV_r,[r_1y,r_2,r_3],[Vr_1y,Vr_2,Vr_3]);
89 SolOmega_1=subs(SolOmega_1,[r_1y,r_2,r_3],[Vr_1y,Vr_2,Vr_3]);
90 SolOmega_3=subs(SolOmega_3,[r_1y,r_2,r_3],[Vr_1y,Vr_2,Vr_3]);
91 SolA_r=subs(SolA_r,[r_1y,r_2,r_3],[Vr_1y,Vr_2,Vr_3]);
92 SolAlpha_1=subs(SolAlpha_1,[r_1y,r_2,r_3],[Vr_1y,Vr_2,Vr_3]);
93 SolAlpha_3=subs(SolAlpha_3,[r_1y,r_2,r_3],[Vr_1y,Vr_2,Vr_3]);
94
95
96
97 Tabla_Cinematica=readtable("Perfil_cinematica.txt","Delimiter"," ");
98 for i=1:length(Tabla_Cinematica.Tiempo)
99     Sol=Fanalitica(SolR_1,SolTheta_1,SolTheta_3,SolV_r,SolOmega_1, ...
100     SolOmega_3,SolA_r,SolAlpha_1,SolAlpha_3, ...
101     Tabla_Cinematica.Posicion(i),Tabla_Cinematica.Velocidad(i), ...
102     Tabla_Cinematica.Aceleracion(i));
103     ValoresSolucion_NR(i,:)=Sol;
104 end
105
106 Tabla_Cinematica.R_1=ValoresSolucion_NR(:,1);
107 Tabla_Cinematica.theta_1=ValoresSolucion_NR(:,2);
108 Tabla_Cinematica.theta_3=ValoresSolucion_NR(:,3);
109 Tabla_Cinematica.V_1=ValoresSolucion_NR(:,4);
110 Tabla_Cinematica.omega_1=ValoresSolucion_NR(:,5);
111 Tabla_Cinematica.omega_3=ValoresSolucion_NR(:,6);

```

```

112 Tabla_Cinematica.a_1=ValoresSolucion_NR(:,7);
113 Tabla_Cinematica.alpha_1=ValoresSolucion_NR(:,8);
114 Tabla_Cinematica.alpha_3=ValoresSolucion_NR(:,9);
115 writetable(Tabla_Cinematica,'ResultadosAnaliticos.txt','delimiter',' ', ...
    'WriteVariableNames', true)
116
117
118 function ...
    solAnalitica=Fanalitica(SolR_1,SolTheta_1,SolTheta_3,SolV_r,SolOmega_1, ...
    ...
119     SolOmega_3,SolA_r,SolAlpha_1,SolAlpha_3,Vtheta_2,Vomega_2,Valpha_2)
120 syms r_1 theta_1 theta_2 theta_3 V_r omega_1 omega_2 omega_3 a_r alpha_1 ...
    alpha_2 alpha_3
121 Vtheta_3=double(subs(SolTheta_3,theta_2,Vtheta_2));
122 Vtheta_1=double(subs(SolTheta_1,[theta_2,theta_3],[Vtheta_2,Vtheta_3]));
123 Vr_1=double(subs(SolR_1,theta_1,Vtheta_1));
124 Vomega_3=double(subs(SolOmega_3,[theta_2,theta_3,omega_2],[Vtheta_2, ...
125     Vtheta_3,Vomega_2]));
126 Vv_r=double(subs(SolV_r,[theta_1,theta_2,theta_3,omega_2,omega_3], ...
127     [Vtheta_1,Vtheta_2,Vtheta_3,Vomega_2,Vomega_3]));
128 Vomega_1=double(subs(SolOmega_1,[r_1,theta_1,V_r],[Vr_1,Vtheta_1,Vv_r]));
129 Valpha_3=double(subs(SolAlpha_3,[theta_2,theta_3,omega_2, ...
130     omega_3,alpha_2],[Vtheta_2,Vtheta_3,Vomega_2,Vomega_3,Valpha_2]));
131 Va_r=double(subs(SolA_r,[r_1,theta_1,theta_2,theta_3,omega_1,omega_2, ...
132     omega_3,alpha_2,alpha_3],[Vr_1,Vtheta_1,Vtheta_2,Vtheta_3,Vomega_1, ...
133     Vomega_2,Vomega_3,Valpha_2,Valpha_3]));
134 Valpha_1=double(subs(SolAlpha_1,[r_1,theta_1,V_r,omega_1,a_r],[Vr_1, ...
135     Vtheta_1,Vv_r,Vomega_1,Va_r]));
136 solAnalitica=[Vr_1,Vtheta_1,Vtheta_3,Vv_r,Vomega_1,Vomega_3,Va_r,Valpha_1,
137     Valpha_3];
138 end

```

E.6. Resultados NR simultaneo vs solución analítica

```

1 clear
2 clc
3
4 Resultados_Analiticos=readtable("ResultadosAnaliticos.txt","Delimiter",' ');
5 Resultados_NR=readtable("Resultados.txt","Delimiter",' ');
6
7 ErrorR1=abs(Resultados_Analiticos.R_1-Resultados_NR.R_1);
8 ErrorTheta1=abs(Resultados_Analiticos.theta_1-Resultados_NR.theta_1);
9 ErrorTheta3=abs(Resultados_Analiticos.theta_3-Resultados_NR.theta_3);
10 ErrorV1=abs(Resultados_Analiticos.V_1-Resultados_NR.V_1);
11 ErrorW1=abs(Resultados_Analiticos.omega_1-Resultados_NR.omega_1);
12 ErrorW3=abs(Resultados_Analiticos.omega_3-Resultados_NR.omega_3);
13 ErrorA1=abs(Resultados_Analiticos.a_1-Resultados_NR.a_1);
14 ErrorAlpha1=abs(Resultados_Analiticos.alpha_1-Resultados_NR.alpha_1);
15 ErrorAlpha3=abs(Resultados_Analiticos.alpha_3-Resultados_NR.alpha_3);
16
17 figure()
18 subplot(3,1,1)
19 plot(ErrorR1)
20 title("Error valores de R_1")
21 ylim([0 10^-15])
22
23 subplot(3,1,2)
24 plot(ErrorTheta1)
25 title("Error valores \theta_1")
26 ylim([0 10^-15])
27
28 subplot(3,1,3)
29 plot(ErrorTheta3)
30 title("Error valores \theta_3")
31 ylim([0 10^-15])
32
33 figure()
34 subplot(3,1,1)
35 plot(ErrorV1)
36 title("Error valores V_r")
37 ylim([0 10^-15])
38
39 subplot(3,1,2)
40 plot(ErrorW1)
41 title("Error valores \omega_1")
42 ylim([0 10^-15])
43
44 subplot(3,1,3)
45 plot(ErrorW3)
46 title("Error valores \omega_3")
47 ylim([0 10^-15])
48
49 figure()
50 subplot(3,1,1)
51 plot(ErrorA1)
52 title("Error valores A_r")
53 ylim([0 10^-15])
54
55 subplot(3,1,2)

```

```
56 plot(ErrorAlpha1)
57 ylim([0 10^-15])
58 title("Error valores \alpha_1")
59
60 subplot(3,1,3)
61 plot(ErrorAlpha3)
62 title("Error valores \alpha_3")
63 ylim([0 10^-15])
```

E.7. Resultados NR simultaneo VS simulación

```

1 clear
2 clc
3 Tabla_Cinematica=readtable("Resultados.txt","Delimiter",' ')
4
5 t = xlsread("Resultados Inventor.xlsx");
6 TablaExcel=array2table(t);
7 TablaExcel.Properties.VariableNames=["Tiempo","Theta2","w2","A2","Theta3",
8   "W3","A3","R1","Vr1","Ar1","Theta1","w1","A1"]
9
10 tiempo=Tabla_Cinematica.Tiempo;
11 ExTiempo=TablaExcel.Tiempo;
12

```

Gráficas

Perfil cinemático

```

16
17 Vmax=double(max(Tabla_Cinematica.Velocidad))*180/pi
18 figure()
19 subplot(3,1,1)
20 plot(tiempo,Tabla_Cinematica.Aceleracion*180/pi,"r-.")
21 hold on
22 plot(ExTiempo,TablaExcel.A2,"k--")
23 hold off
24 title('Aceleracion vs tiempo')
25 grid on
26 xlabel('t (s)')
27 ylabel('a (\circ s^{-2})')
28 xlim([0 ExTiempo(end)])
29
30 subplot(3,1,2)
31 plot(tiempo,Tabla_Cinematica.Velocidad*180/pi,"r-.")
32 hold on
33 plot(ExTiempo,TablaExcel.w2,"k--")
34 hold off
35 title('Velocidad vs tiempo')
36 grid on
37 xlabel('t (s)')
38 ylabel('v (\circ s^{-1})')
39 ylim([0 Vmax*1.1])
40 xlim([0 ExTiempo(end)])
41
42 subplot(3,1,3)
43 plot(tiempo,Tabla_Cinematica.Posicion*180/pi,"r-.")
44 hold on
45 plot(ExTiempo,TablaExcel.Theta2,"k--")
46 hold off
47 title('Posicion vs tiempo')
48 grid on
49 xlabel('t (s)')
50 ylabel('z (\circ)')
51 xlim([0 ExTiempo(end)])
52

```

Gráficas de simulación y NR

```

53 figure()
54

```



```

55
56 subplot(3,1,1)
57 plot(tiempo, Tabla_Cinematica.R_1, "r-.")
58 hold on
59 plot(ExTiempo, -TablaExcel.R1, "k--")
60 hold off
61 title("r_1 vs tiempo")
62 xlabel("t (s)")
63 ylabel("r_1 (m)")
64 grid on
65 xlim([0 tiempo(end)])
66 legend("Newton Raphson", "Simulacion", "Location", "east", 'FontSize', 7)
67
68 subplot(3,1,2)
69 plot(tiempo, Tabla_Cinematica.theta_1*180/pi, "r-.")
70 hold on
71 plot(ExTiempo, TablaExcel.Theta1, "k--")
72 hold off
73 title("\theta_1 vs tiempo")
74 xlabel("t (s)")
75 ylabel("\theta_1 (\circ)")
76 grid on
77 xlim([0 tiempo(end)])
78 yticks(-30:5:-10)
79
80 subplot(3,1,3)
81 plot(tiempo, Tabla_Cinematica.theta_3*180/pi, "r-.")
82 hold on
83 plot(ExTiempo, TablaExcel.Theta3+180, "k--")
84 hold off
85 title("\theta_3 vs tiempo")
86 xlabel("t (s)")
87 ylabel("\theta_3 (\circ)")
88 grid on
89 xlim([0 ExTiempo(end)])
90 ylim([125 200])
91 yticks(125:25:200)
92
93
94 figure()
95 subplot(3,1,1)
96 plot(tiempo, Tabla_Cinematica.V_1, "r-.")
97 hold on
98 plot(ExTiempo, TablaExcel.Vr1, "k--")
99 hold off
100 title("V_r vs tiempo")
101 xlabel("t (s)")
102 ylabel("V_r (m s^{-1})")
103 grid on
104 xlim([0 ExTiempo(end)])
105 ylim([-0.6 0.6])
106 yticks(-0.6:0.3:0.6)
107 legend("Newton Raphson", "Simulacion", "Location", "east", 'FontSize', 7)
108
109 subplot(3,1,2)
110 plot(tiempo, Tabla_Cinematica.omega_1*180/pi, "r-.")
111 hold on
112 plot(ExTiempo, TablaExcel.w1, "k--")

```

```

113 hold off
114 title("\omega_1 vs tiempo")
115 xlabel("t (s)")
116 ylabel("\omega_1 (\circ s^{-1})")
117 grid on
118 xlim([0 ExTiempo(end)])
119 ylim([-80 80])
120 yticks(-80:40:80)
121
122
123 subplot(3,1,3)
124 plot(tiempo, Tabla_Cinematica.omega_3*180/pi, "r-.")
125 hold on
126 plot(ExTiempo, TablaExcel.W3, "k--")
127 hold off
128 title("\omega_3 vs tiempo")
129 xlabel("t (s)")
130 ylabel("\omega_3 (\circ s^{-1})")
131 grid on
132 xlim([0 ExTiempo(end)])
133 yticks([-200:100:200])
134
135 figure()
136 subplot(3,1,1)
137 plot(tiempo, Tabla_Cinematica.a_1, "r-.")
138 hold on
139 plot(ExTiempo, TablaExcel.Ar1, "k--")
140 hold off
141 title("A_r vs tiempo")
142 xlabel("t (s)")
143 ylabel("A_r (m s^{-2})")
144 grid on
145 xlim([0 ExTiempo(end)])
146 yticks(-5:2.5:5)
147 legend("Newton Raphson", "Simulacion", "Location", "east", 'FontSize', 7)
148
149 subplot(3,1,2)
150 plot(tiempo, Tabla_Cinematica.alpha_1*180/pi, "r-.")
151 hold on
152 plot(ExTiempo, TablaExcel.A1, "k--")
153 hold off
154 title("\alpha_1 vs tiempo")
155 xlabel("t (s)")
156 ylabel("\alpha_1 (\circ s^{-2})")
157 grid on
158 xlim([0 ExTiempo(end)])
159 ylim([-400 800])
160 yticks([-400:400:800])
161
162 subplot(3,1,3)
163 plot(tiempo, Tabla_Cinematica.alpha_3*180/pi, "r-.")
164 hold on
165 plot(ExTiempo, TablaExcel.A3, "k--")
166 hold off
167 title("\alpha_3 vs tiempo")
168 xlabel("t (s)")
169 ylabel("\alpha_3 (\circ s^{-2})")
170 grid on

```

```
171 xlim([0 ExTiempo(end)])  
172 ylim([-1500 1500])  
173 yticks(-1500:750:1500)
```