Análisis Dinámico

Dinámica de Maquinaria

Jonathan Camargo

jon-cama@uniandes.edu.co

Universidad de los Andes
Colombia

Temas

- Fuerzas y momentos
- Método de Newton – Euler
- Dinámica inversa
- Dinámica directa

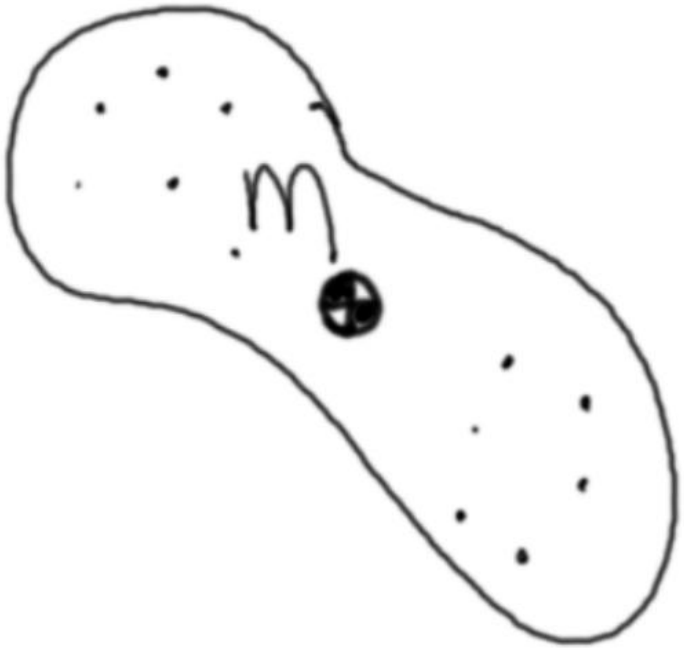
Fuerzas y momentos

$$\sum \mathbf{F} = 0 \Rightarrow \frac{dv}{dt} = 0$$

$$F = ma$$

$$F_A = -F_B$$

Fuerzas y momentos



$$\dot{H}_o = \Sigma M_o = \vec{r}_{cm} \times m\vec{a}_{cm} + I_{cm}\alpha$$

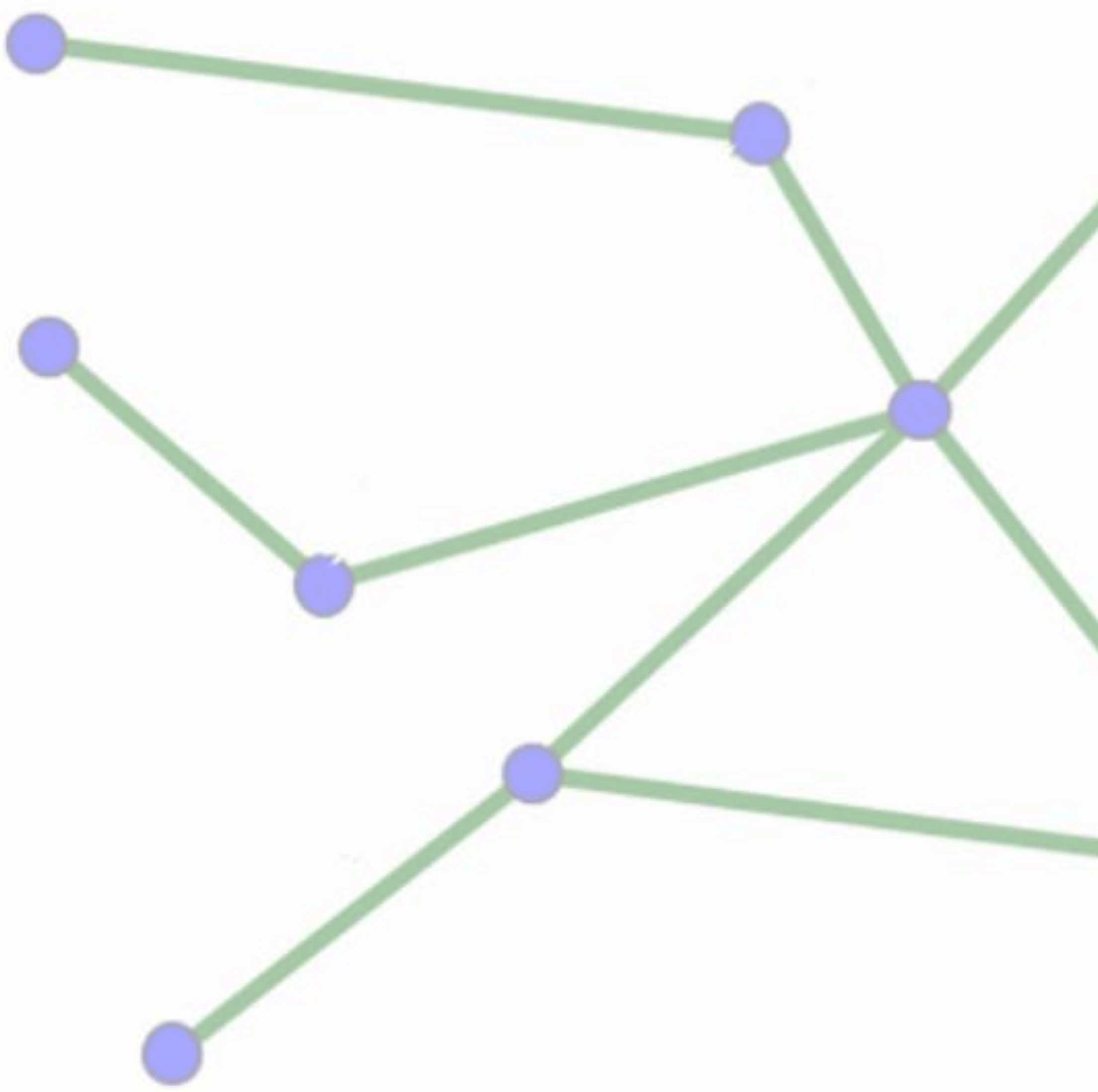
$$\dot{H}_{cm} = \Sigma M_{cm} = I_{cm}\alpha$$

$$\Sigma \vec{F} = m\vec{a}_{cm}$$

Método de Newton Euler

- Para cada nodo plantear:
  - Diagrama de cuerpo libre
  - **F=m*a, Mcm=I*alpha**

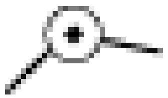
Se llega a un set de ecuaciones
Con variables:
qddot, reacciones, Fext, Mext



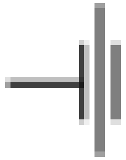
Universidad de los Andes Colombia

Método de Newton Euler

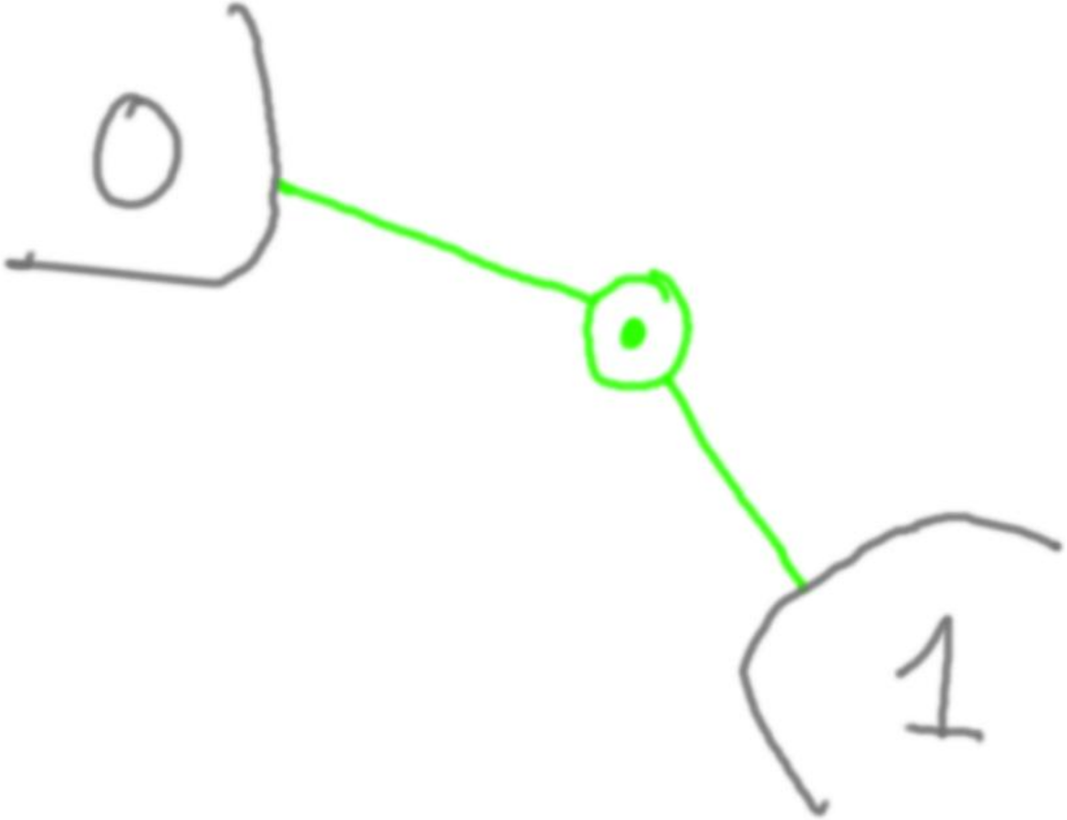
- Reacciones



Revolute joints

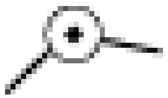


Prismatic joints

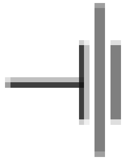


Método de Newton Euler

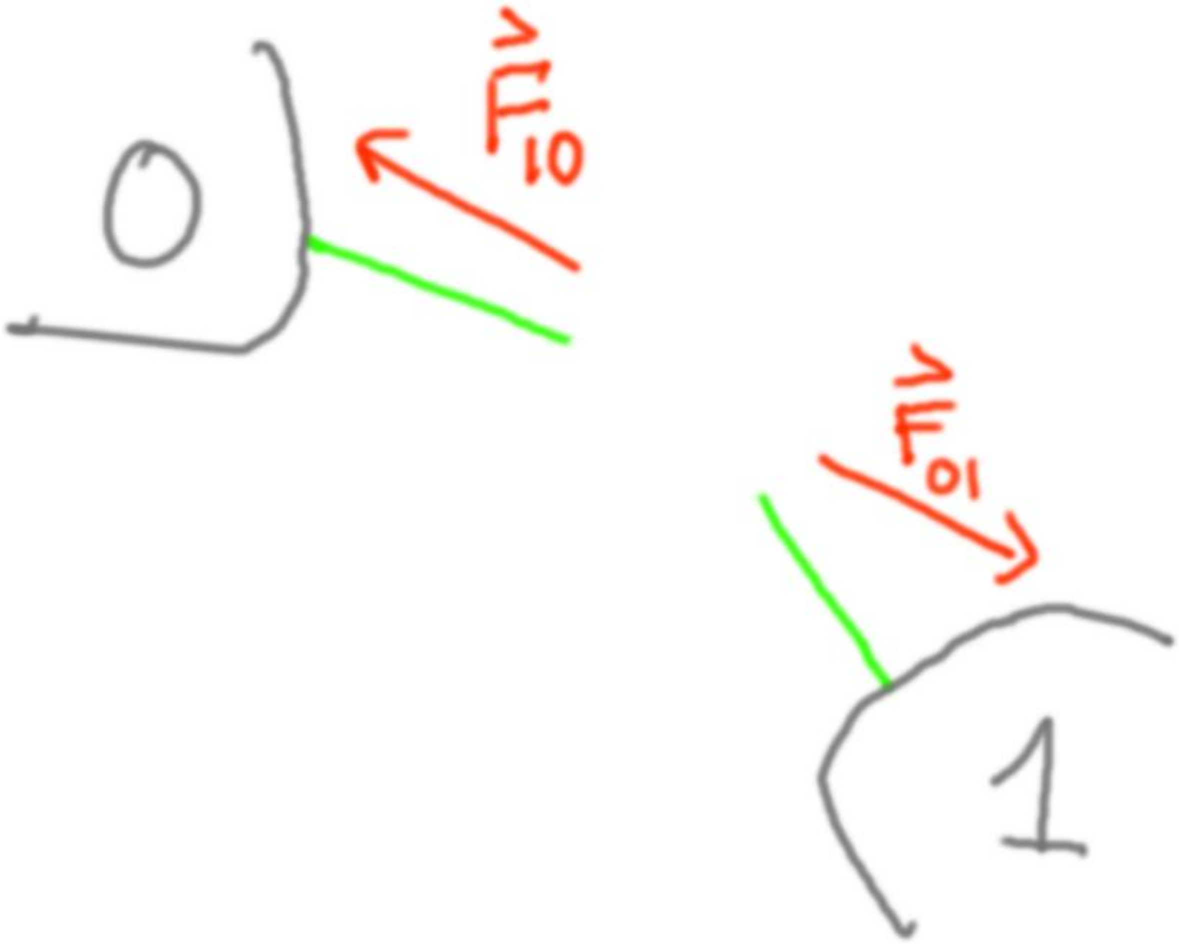
- Reacciones



Revolute joints

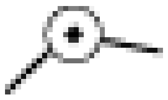


Prismatic joints

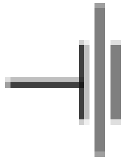


Método de Newton Euler

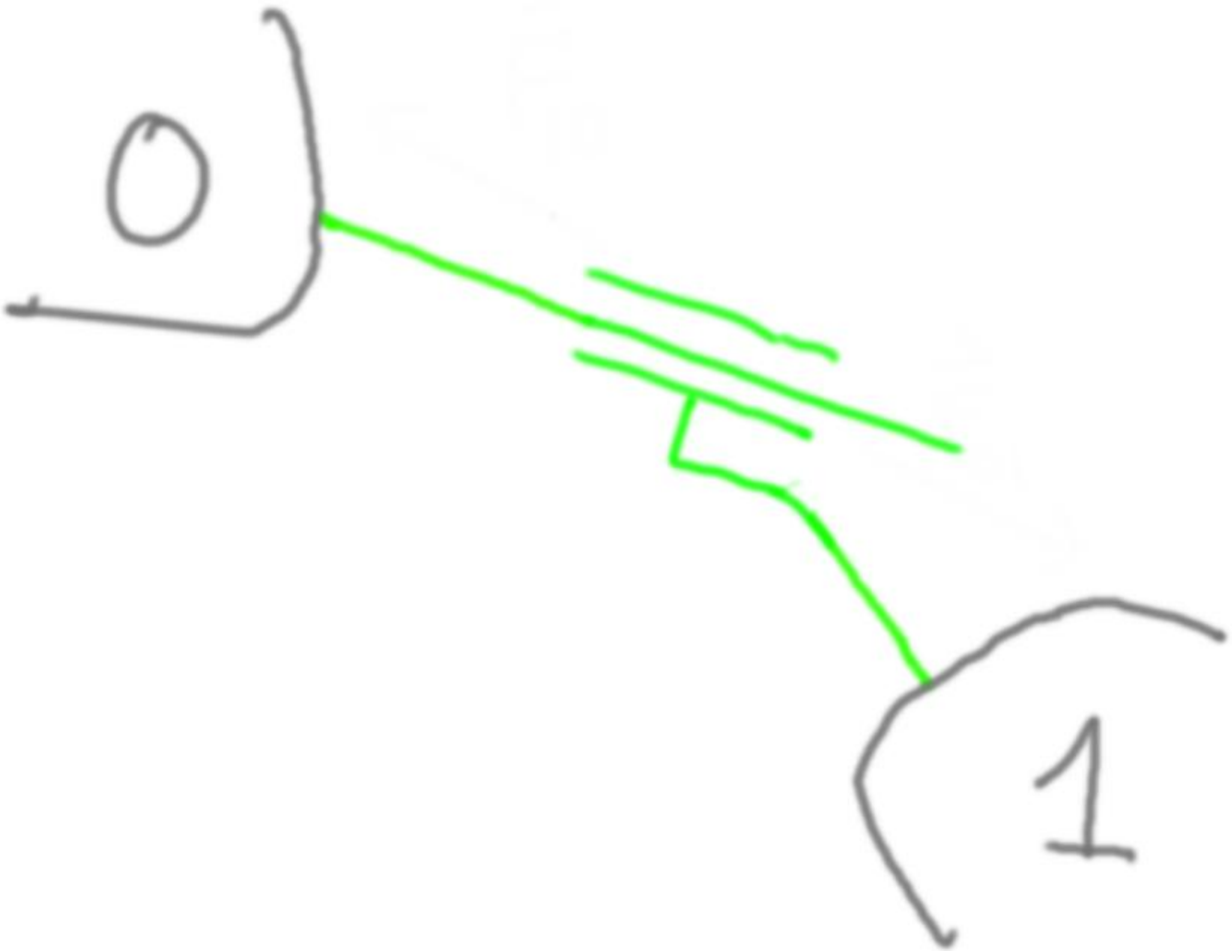
- Reacciones



Revolute joints

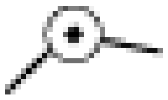


Prismatic joints

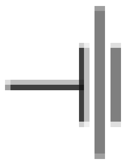


Método de Newton Euler

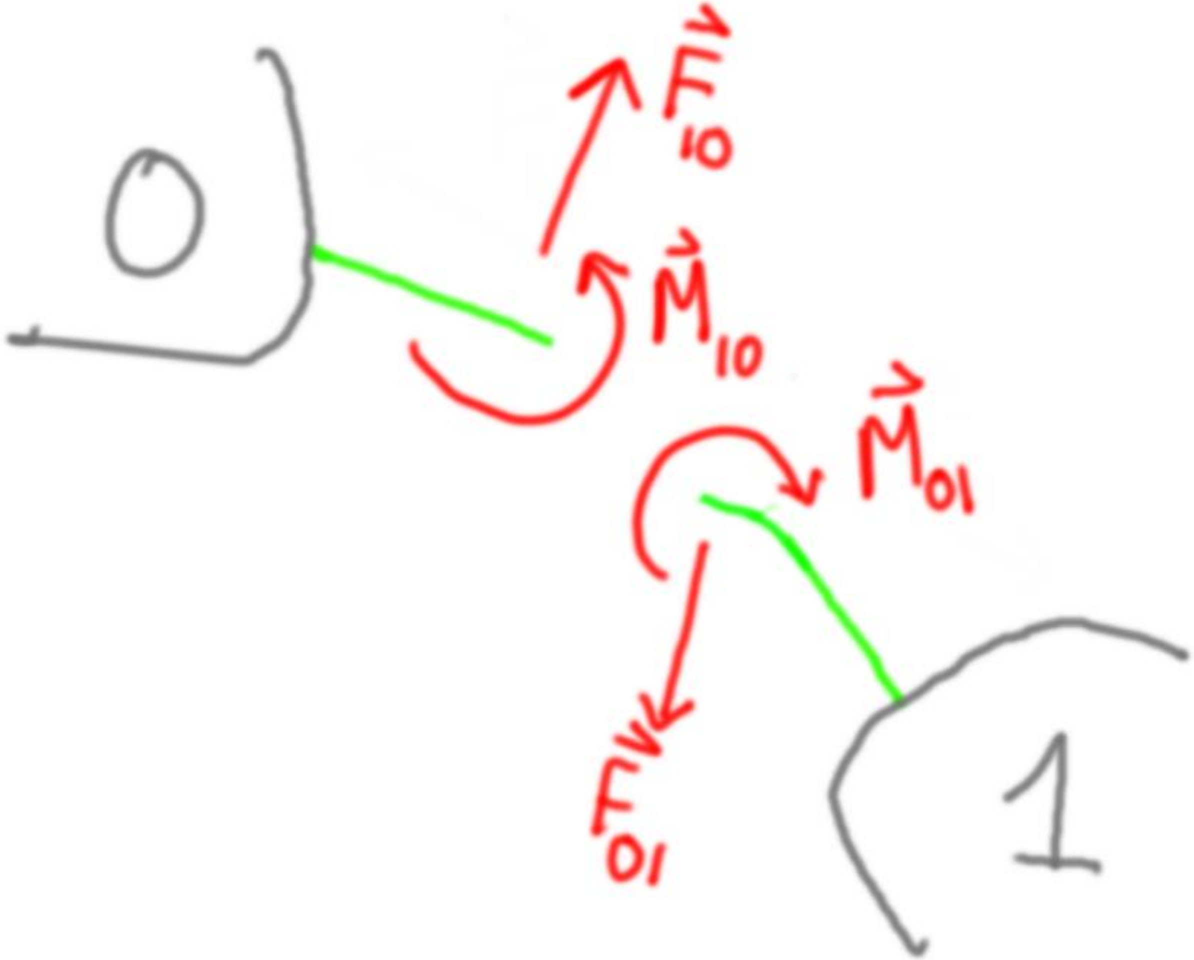
- Reacciones



Revolute joints



Prismatic joints



Método de Newton Euler

- Para cada nodo plantear:
  - Diagrama de cuerpo libre
  - **F**=m***a**, **Mcm**=I***alpha**

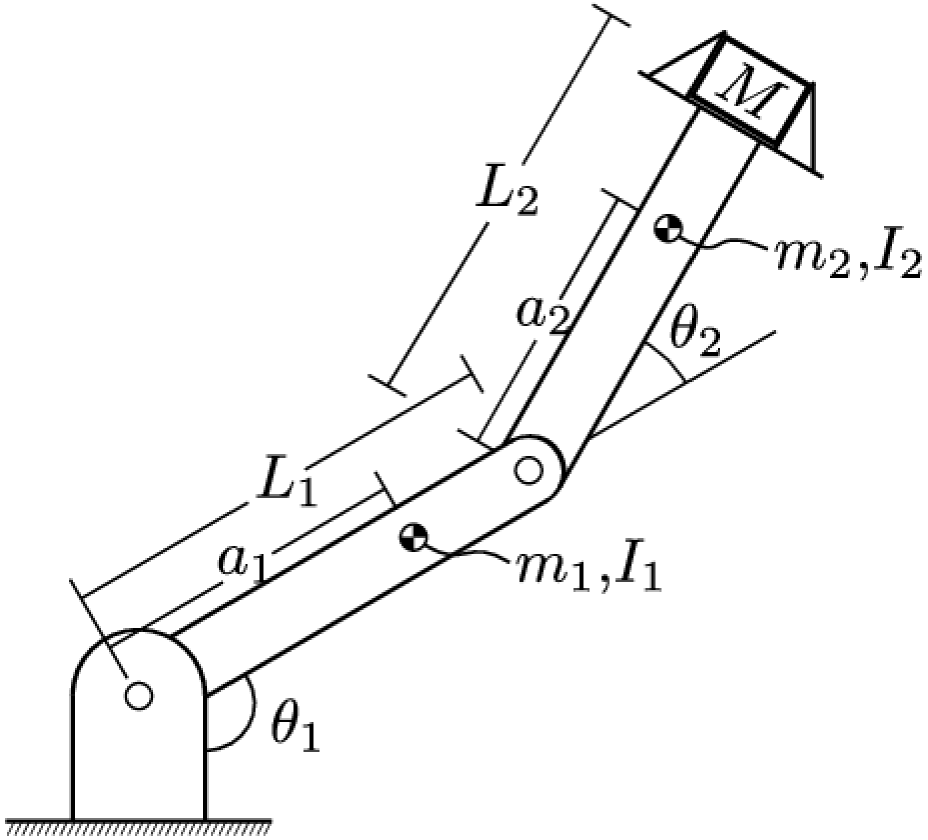
Se llega a un set de ecuaciones
Con variables:
qddot, reacciones, Fext, Mext

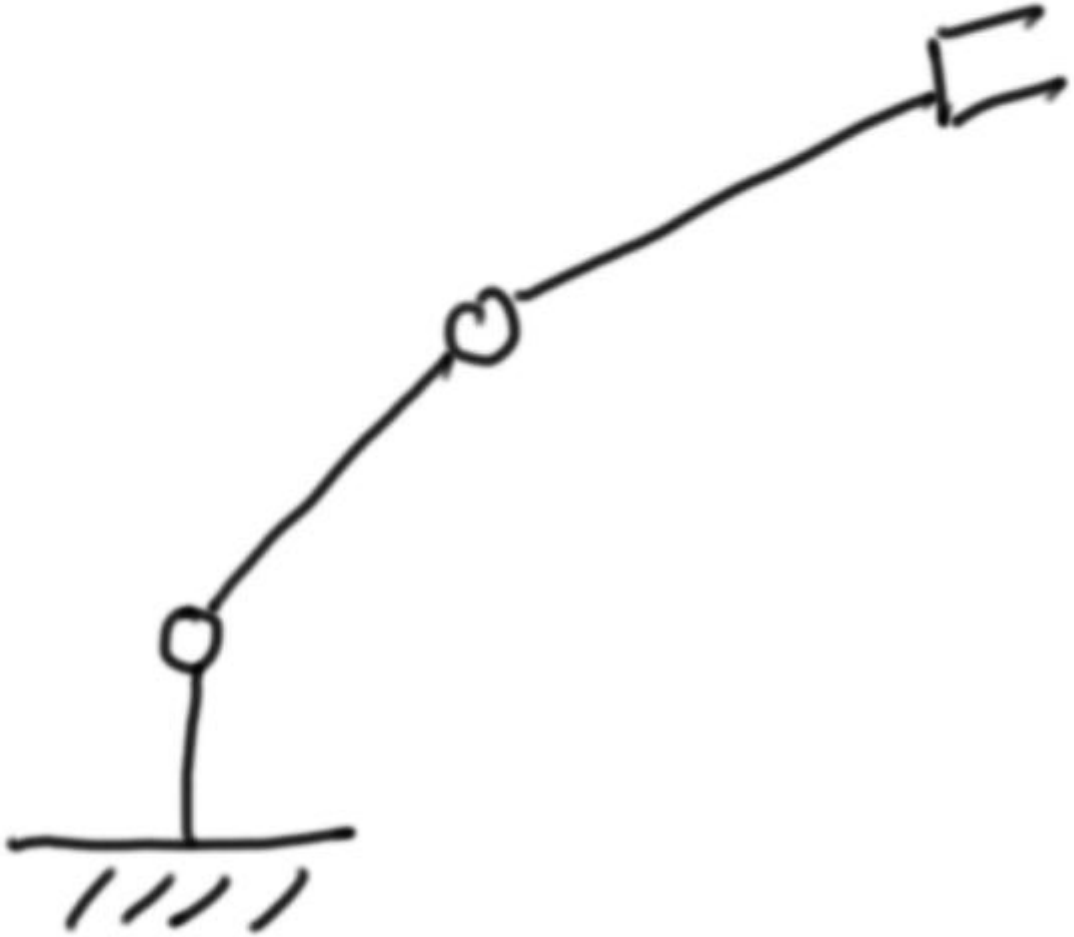
Pensar

Grafo con N nodos
Grafo con K uniones

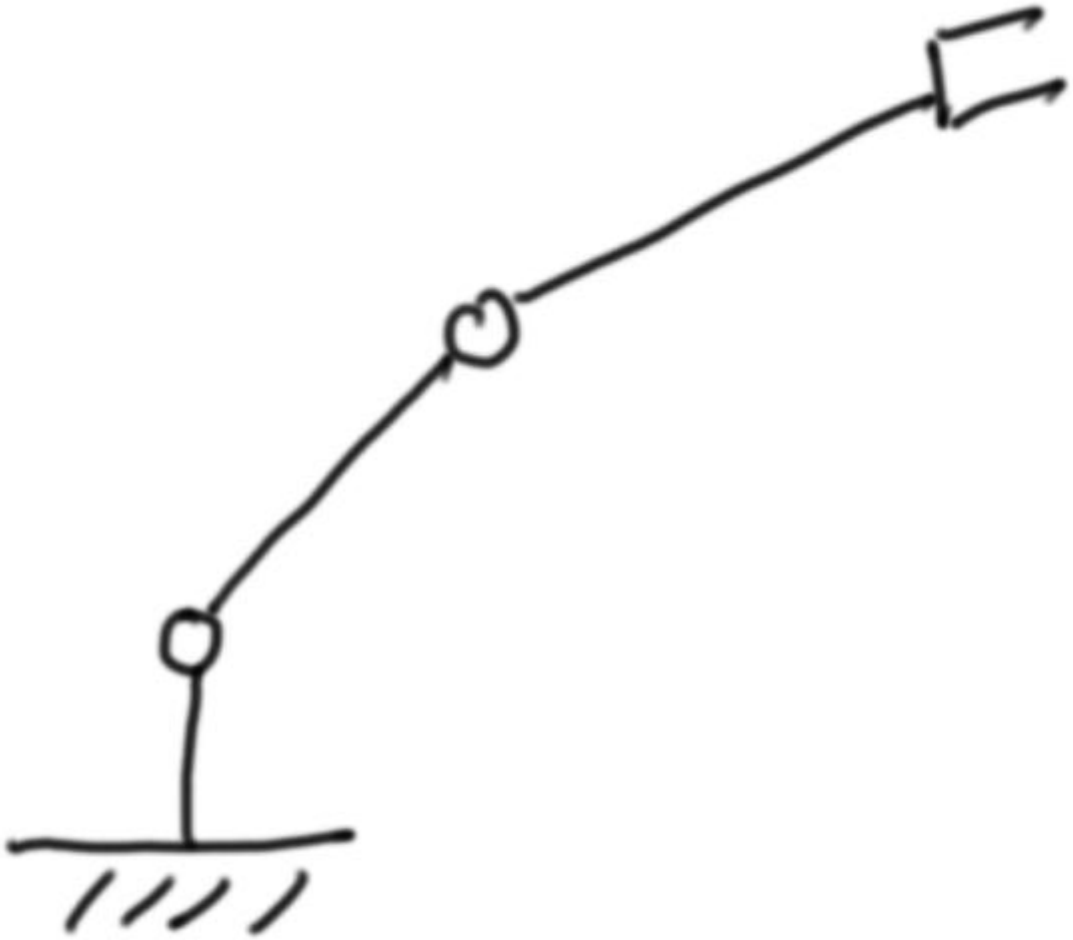
¿Cuántas ecuaciones salen?
¿Cuántas variables escalares en
qddot, reacciones, Fext, Mext ?

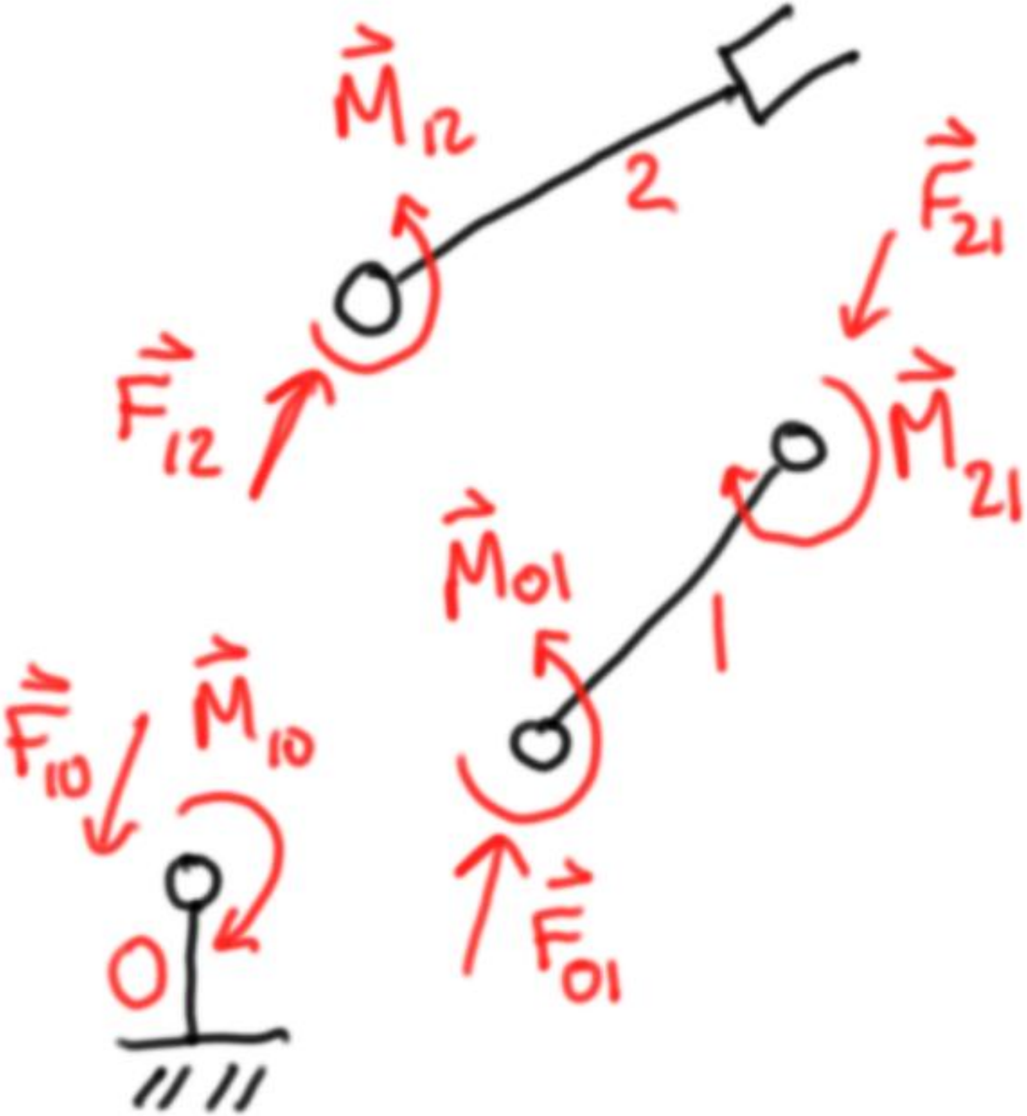
Ejemplo





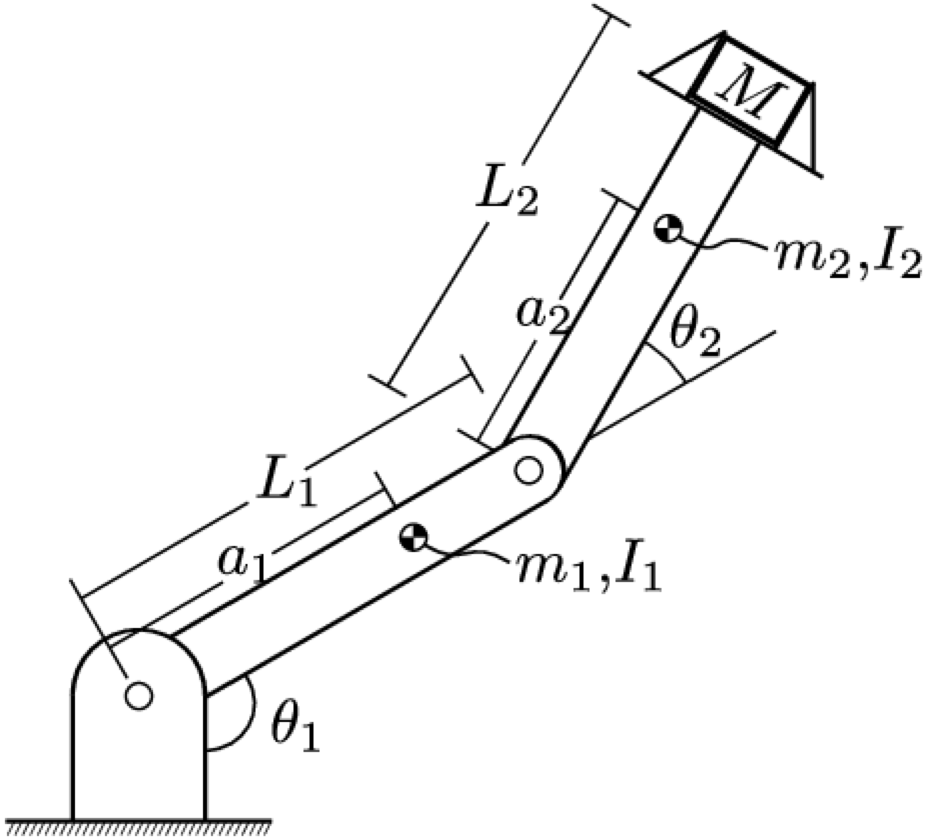
Ejemplo



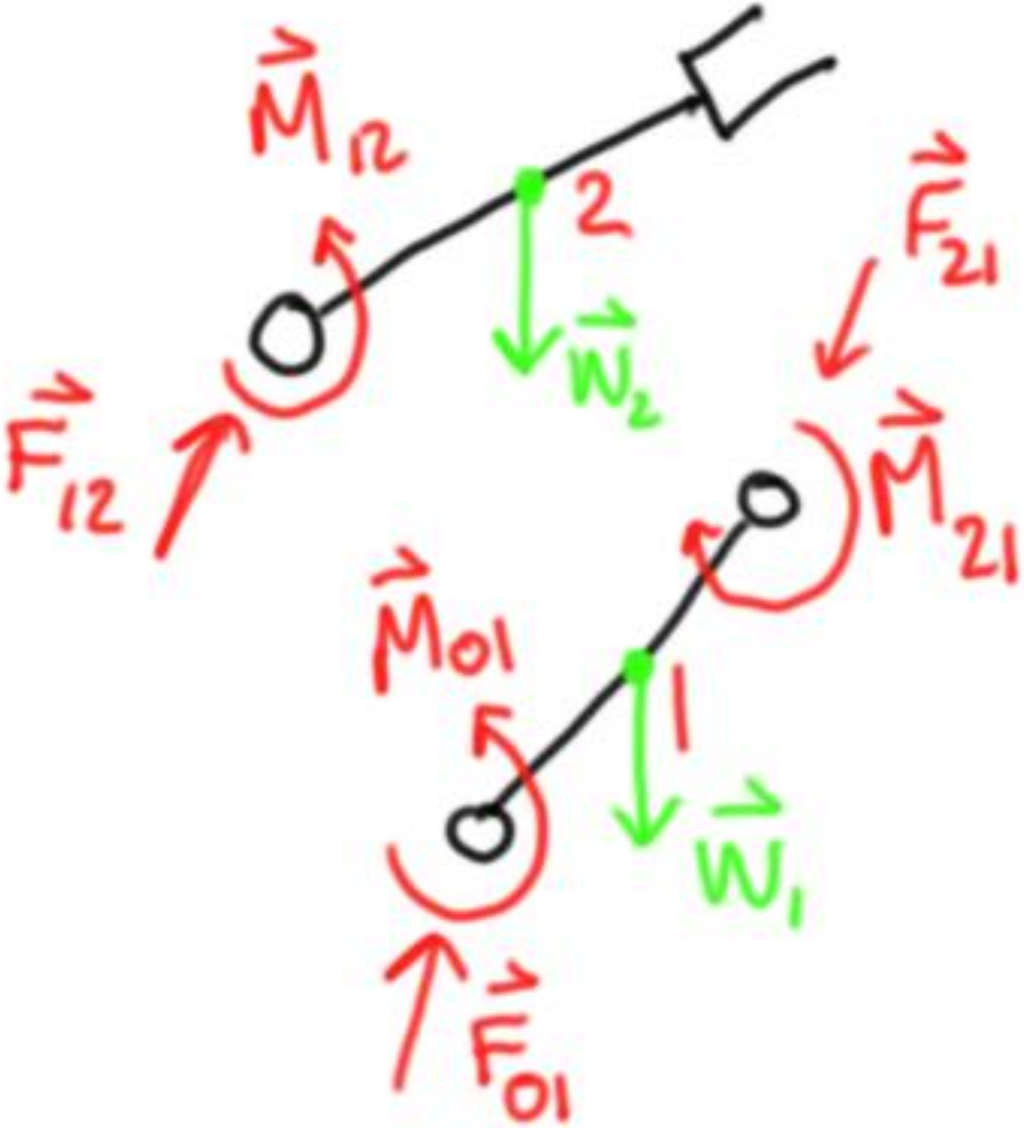


$\vec{M}_{12}$
2
$\vec{F}_{21}$
$\vec{F}_{12}$
$\vec{M}_{21}$
$\vec{M}_{01}$
1
$\vec{F}_{10}$
$\vec{M}_{10}$
0
$\vec{F}_{01}$

Ejemplo

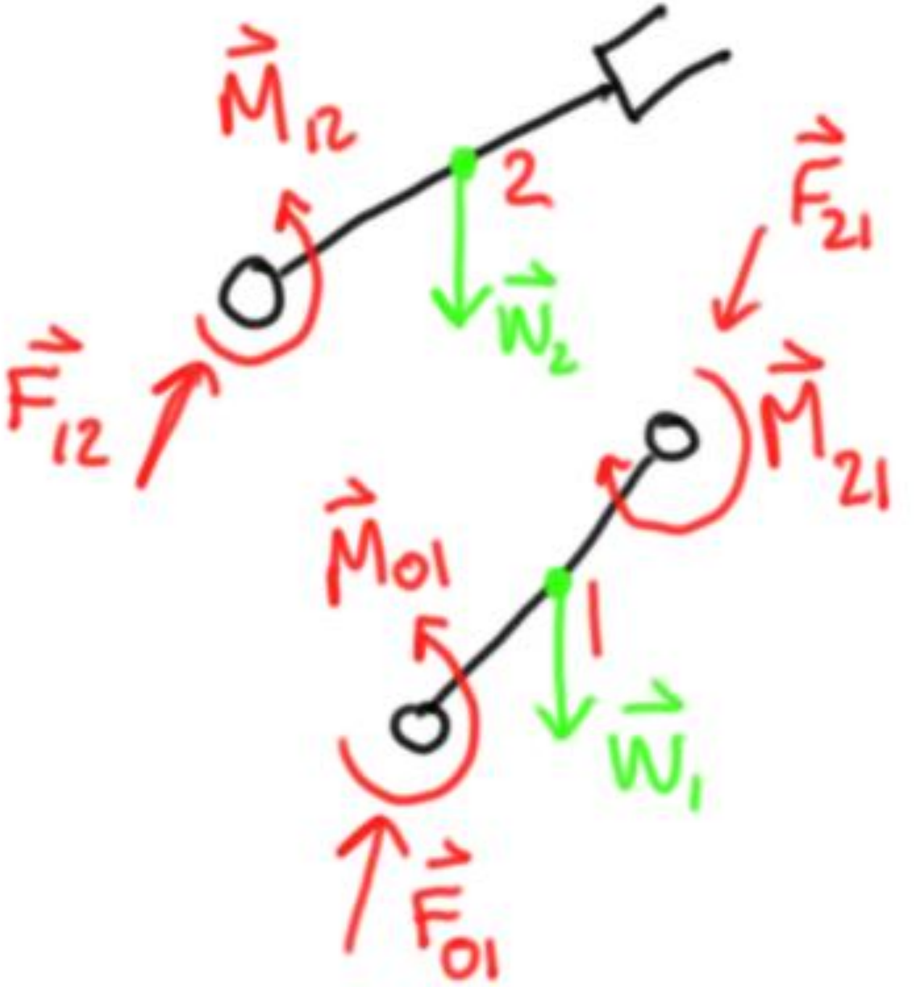


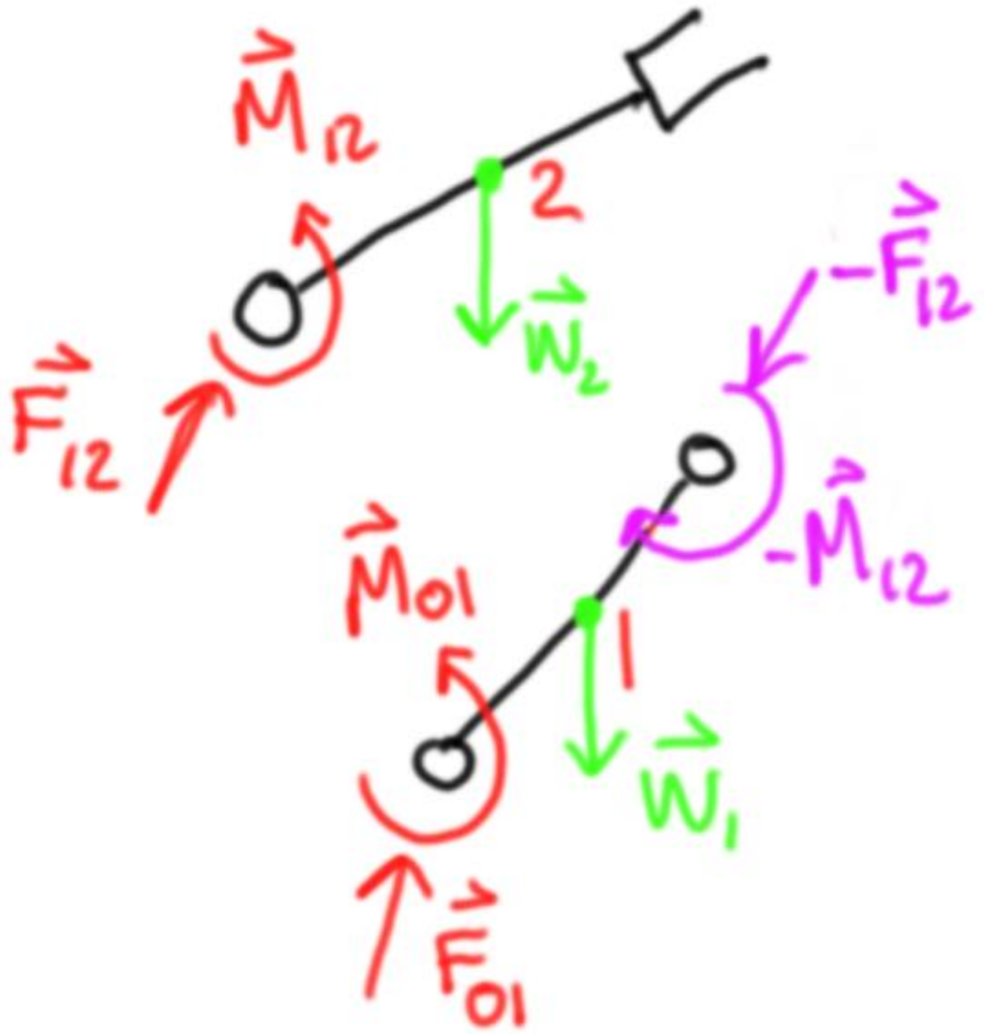
L_2
M
m_2, I_2
a_2
θ_2
L_1
a_1
m_1, I_1
θ_1



$\vec{M}_{12}$
2
$\vec{F}_{21}$
$\vec{F}_{12}$
$\vec{W}_2$
$\vec{M}_{21}$
$\vec{M}_{01}$
1
$\vec{W}_1$
$\vec{F}_{01}$

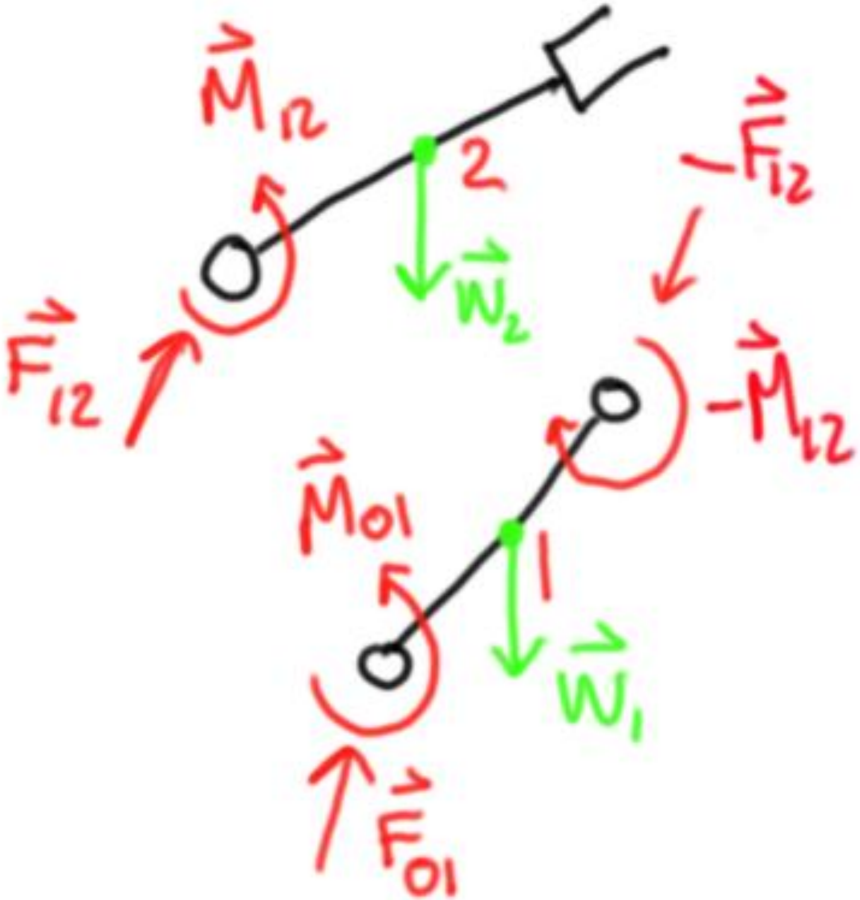
Ejemplo

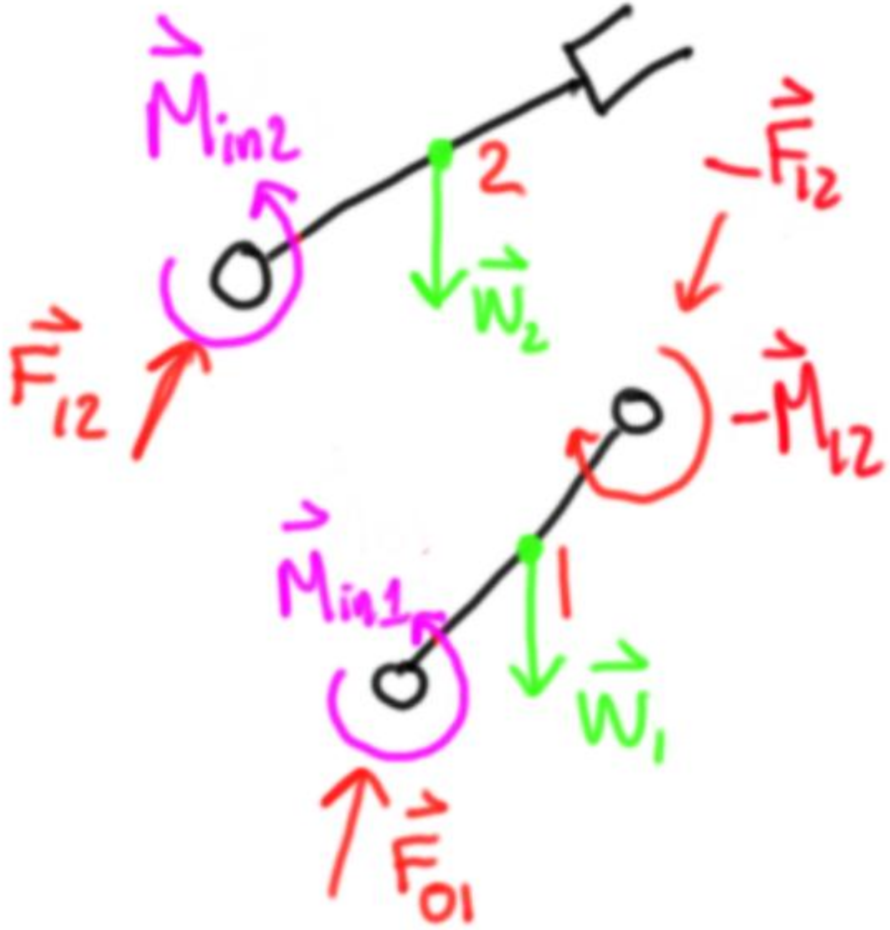




3era Ley de Newton

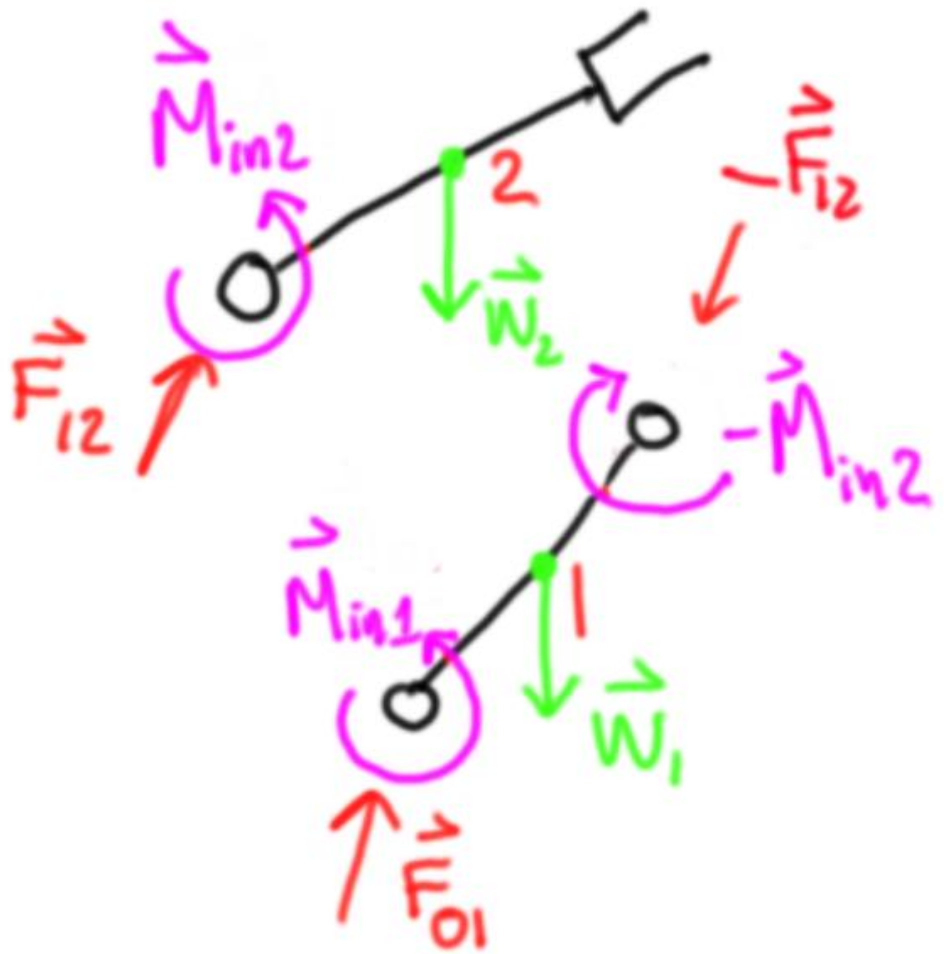
Ejemplo





Renombrar o relacionar:
Fuerzas – Momentos externos

Ejemplo

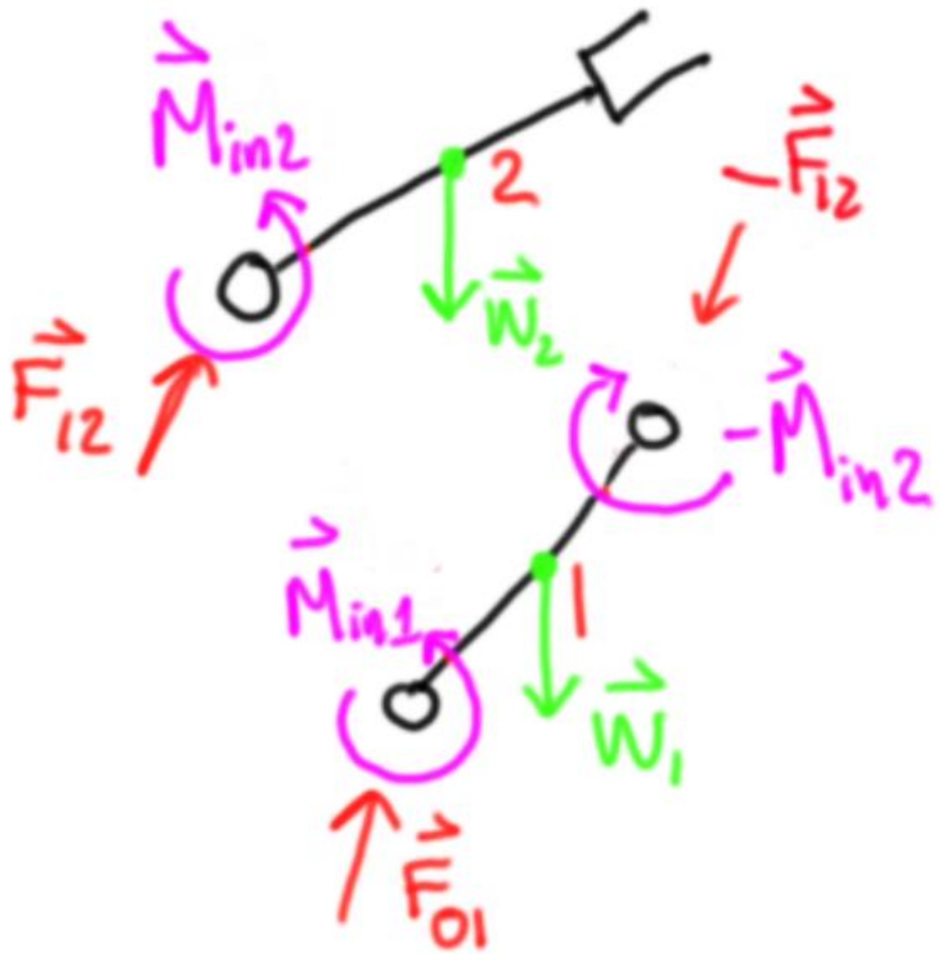


F=m*a, Mcm=I*alpha

Incógnitas vs ecuaciones

6 ecuaciones
¿Cuántas incógnitas?

Ejemplo



Kinematics (Again)

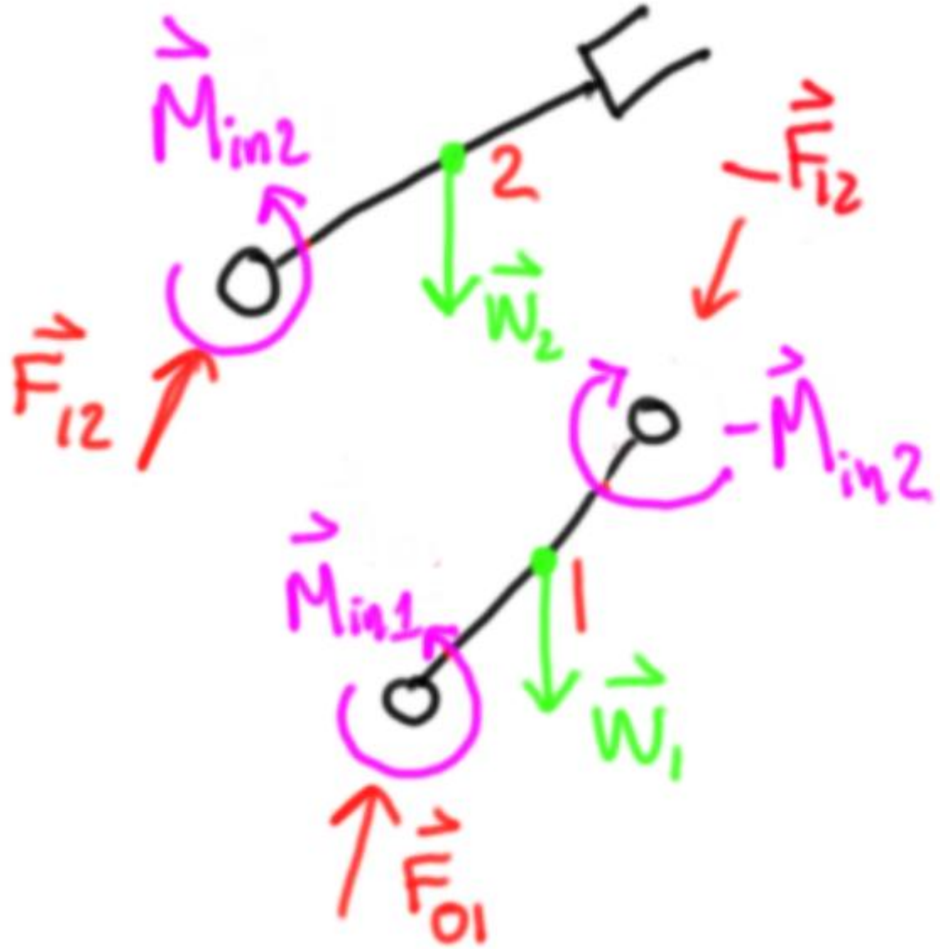
```
theta1,theta2=dynamicsymbols('theta1,theta2')
t,l1,l2,m1,m2,lc1,lc2,I1,I2=symbols('t,l1,l2,m1,m2,lc1,lc2,I1,I2')
params={l1:1,l2:1,m1:1,m2:1,lc1:0.5,lc2:0.5,I1:1,I2:1}
N=ReferenceFrame('N')
A=N.orientnew('A','Axis',(theta1-sympy.pi/2,N.z))
B=A.orientnew('B','Axis',(theta2,N.z))

# From kinematic analysis we found that we can
# describe any point from q [theta1,theta2] as:
rcm1=lc1*A.x
rcm2=l1*A.x+lc2*B.x
```

Cualquier posición puede ser definida a través de q

e.g. Centros de masa

Ejemplo

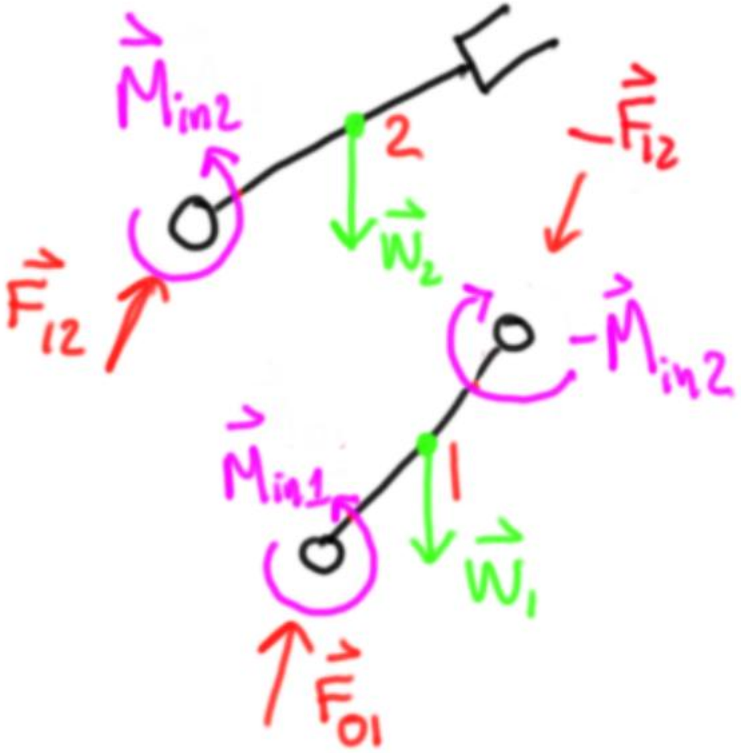



```
# Define force and moment variables
F01_x,F01_y,F12_x,F12_y,Min_1,Min_2=dynamicsymbols('F01_x F01_y F12_x F12_y Min_1,Min_2')

# Define vectors
F01=F01_x*A.x+F01_y*A.y
F12=F12_x*B.x+F12_y*B.y
```

Las reacciones se pueden escribir con sus componentes en Cualquier marco de referencia de interés.

Ejemplo




```
# Newton-Euler equations for each body

# Body 1 Forces and moments
eqF1=F01-m1*9.81*N.y-F12-m1*rcm1.diff(t,N).diff(t,N)
eqM1=Min_1*N.z-Min_2*N.z \
    -lc1*A.x.cross(F01) + (l1-lc1)*A.x.cross(-F12) \
    -I1*theta1.diff(t,t)*N.z

# Body 2 Forces and moments
eqF2=F12-m2*9.81*N.y-m2*rcm2.diff(t,N).diff(t,N)
eqM2=Min_2*N.z+(-lc2*B.x).cross(F12)-I2*(theta1+theta2).diff(t,t)*N.z

# 6 scalar equations
eqList=[eq.simplify() for eq in [eqF1.dot(N.x),eqF1.dot(N.y),eqF2.dot(N.x),eqF2.dot(N.y) \
        ,eqM1.dot(N.z),eqM2.dot(N.z)]]
#Observe each equation and understand role of each variable in the terms
eqList[0]
```

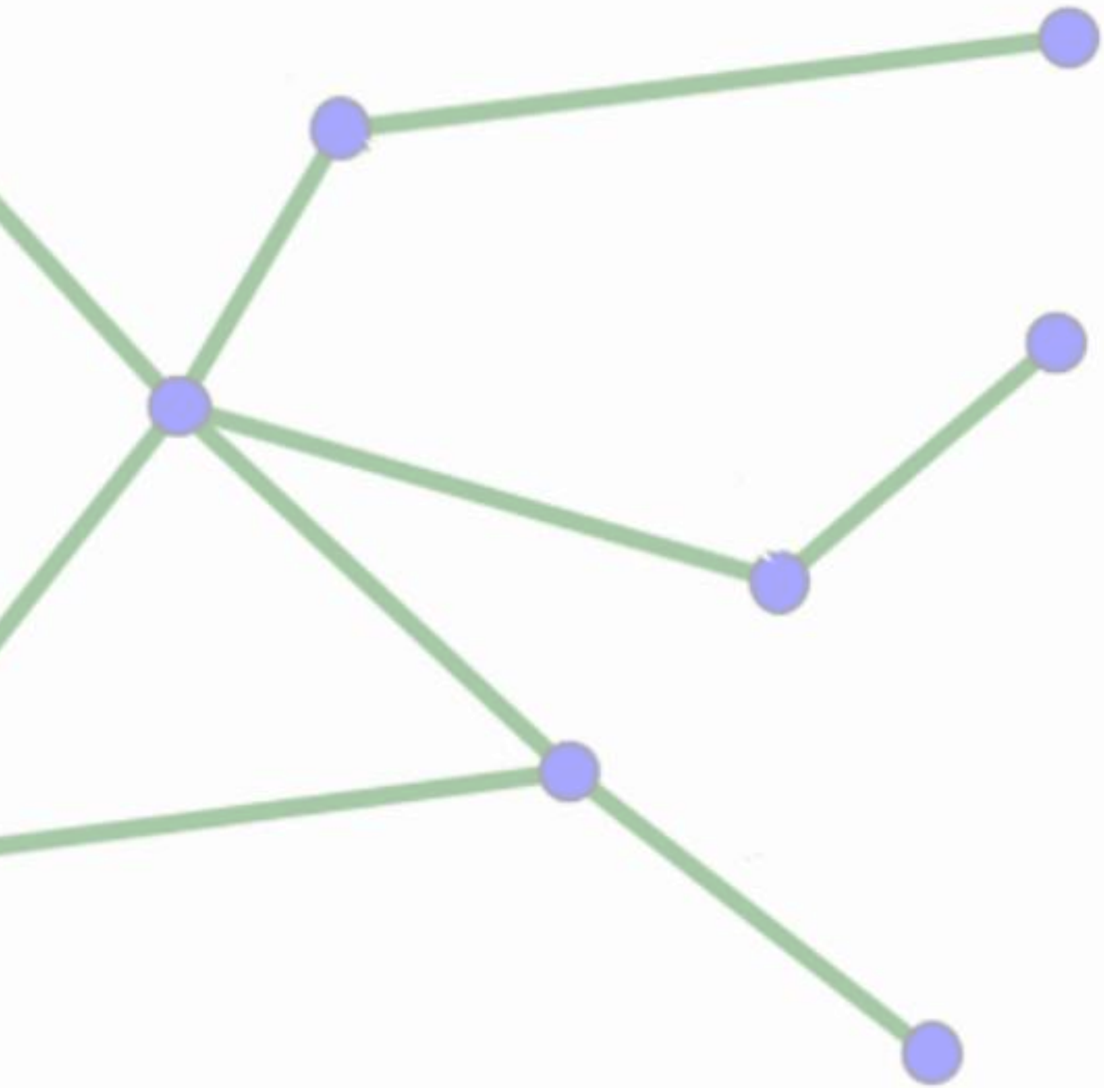
✓ 5.1s

$$lc_1 m_1 \left(\sin\left(\theta_1(t)\right)\left(\frac{d}{dt}\theta_1(t)\right)^2 - \cos\left(\theta_1(t)\right)\frac{d^2}{dt^2}\theta_1(t)\right) + F_{01x}(t)\sin\left(\theta_1(t)\right) + F_{01y}(t)\cos\left(\theta_1(t)\right) - F_{12x}(t)\sin\left(\theta_1(t)+\theta_2(t)\right) - F_{12y}(t)\cos\left(\theta_1(t)+\theta_2(t)\right)$$

Ecuaciones con términos q, qdot y qddot
… y reacciones
… y fuerzas o momentos externos

Dinámica Inversa

Dinámica Inversa

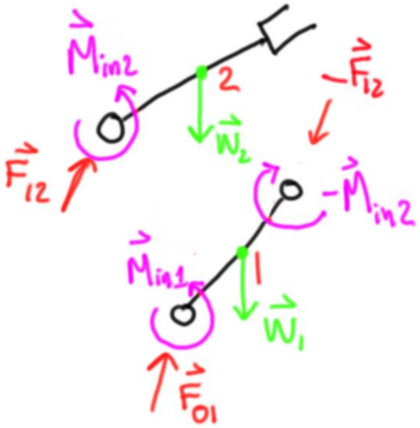


Se llegó a un set de ecuaciones
Con variables:
qddot, reacciones, Fext, Mext

Conocemos:
qddot

Despejamos:
reacciones, Fext, Mext

Dinámica inversa




```python
# Our unkowns are the forces and moments and we know the motion

[A,b]=sympy.linear_eq_to_matrix(eqList,[F01_x,F01_y,F12_x,F12_y,Min_1,Min_2])

A
```
✓ 0.0s

$$\begin{bmatrix} \sin(\theta_1(t)) & \cos(\theta_1(t)) & -\sin(\theta_1(t)+\theta_2(t)) & -\cos(\theta_1(t)+\theta_2(t)) & 0 & 0 \\ -\cos(\theta_1(t)) & \sin(\theta_1(t)) & \cos(\theta_1(t)+\theta_2(t)) & -\sin(\theta_1(t)+\theta_2(t)) & 0 & 0 \\ 0 & 0 & \sin(\theta_1(t)+\theta_2(t)) & \cos(\theta_1(t)+\theta_2(t)) & 0 & 0 \\ 0 & 0 & -\cos(\theta_1(t)+\theta_2(t)) & \sin(\theta_1(t)+\theta_2(t)) & 0 & 0 \\ 0 & -lc_1 & -(l_1-lc_1)\sin(\theta_2(t)) & -(l_1-lc_1)\cos(\theta_2(t)) & 1 & -1 \\ 0 & 0 & 0 & -lc_2 & 0 & 1 \end{bmatrix}$$

Sistema Lineal

Ax=b

```python
b
```
✓ 0.0s

$$-lc_1 m_1 \left(\sin(\theta_1(t))\left(\frac{d}{dt}\theta_1(t)\right)^2 - \cos(\theta_1(t))\frac{d^2}{dt^2}\theta_1(t)\right)$$

$$lc_1 m_1 \left(\sin(\theta_1(t))\frac{d^2}{dt^2}\theta_1(t) + \cos(\theta_1(t))\left(\frac{d}{dt}\theta_1(t)\right)^2\right) + 9.81 m_1$$

$$)\frac{d^2}{dt^2}\theta_1(t) + lc_2 (\sin(\theta_1(t)+\theta_2(t))\left(\frac{d}{dt}\theta_1(t)\right)^2 + 2\sin(\theta_1(t)+\theta_2(t))\frac{d}{dt}\theta_1(t)\frac{d}{dt}\theta_2(t) + \sin(\theta_1(t)+\theta_2(t))\left(\frac{d}{dt}\theta_2(t)\right)^2 - \cos($$

$$)^2 + lc_2 (\sin(\theta_1(t)+\theta_2(t))\frac{d^2}{dt^2}\theta_1(t) + \sin(\theta_1(t)+\theta_2(t))\frac{d^2}{dt^2}\theta_2(t) + \cos(\theta_1(t)+\theta_2(t))\left(\frac{d}{dt}\theta_1(t)\right)^2 + 2\cos(\theta_1(t)+\theta_2(t))\frac{d}{dt}$$

$$I_1\frac{d^2}{dt^2}\theta_1(t)$$

$$I_2\left(\frac{d^2}{dt^2}\theta_1(t) + \frac{d^2}{dt^2}\theta_2(t)\right)$$

Dinámica inversa

Evaluar numéricamente

e.g. theta1=45°, theta2=10°, theta1dot=0.1rad/s, theta2dot=0.25rad/s, theta1ddot=theta2ddot=0

```python
# Numerical evaluation:

values={theta1.diff(t,t):0,theta2.diff(t,t):0,theta1.diff(t):0.1, \
    theta2.diff(t):0.25,theta1:np.deg2rad(45),theta2:np.deg2rad(10)}

A_num=A.subs(params).subs(values).evalf()
A_num=np.array(A_num).astype(np.float64)
b_num=b.subs(params).subs(values).evalf()
b_num=np.array(b_num).astype(np.float64)

np.matmul(np.linalg.pinv(A_num),b_num)
```

✓ 0.0s Python

```
array([[-13.93875452],
       [ 13.8627991 ],
       [ -5.69788292],
       [  8.03761804],
       [ 14.41324935],
       [  4.01880902]])
```

Vector solución:

[F01_x,F01_y,F12_x,F12_y,Min_1,Min_2]

Dinámica inversa

Evaluar numéricamente (lambdify)

e.g. theta1=45°, theta2=10°, theta1dot=0.1rad/s, theta2dot=0.25rad/s, theta1ddot=theta2ddot=0

```python
# Numerical evaluation with lambdify:
values={theta1.diff(t,t):0,theta2.diff(t,t):0,theta1.diff(t):0.1, \
    theta2.diff(t):0.25,theta1:np.deg2rad(45),theta2:np.deg2rad(10)}

A_fun=sympy.lambdify(list(values.keys()),A.subs(params))
A_num=A_fun(*list(values.values()))
b_fun=sympy.lambdify(list(values.keys()),b.subs(params))
b_num=b_fun(*list(values.values()))

np.matmul(np.linalg.pinv(A_num),b_num)
```

✓ 0.0s Python

```
array([[-13.93875452],
       [ 13.8627991 ],
       [ -5.69788292],
       [  8.03761804],
       [ 14.41324935],
       [  4.01880902]])
```

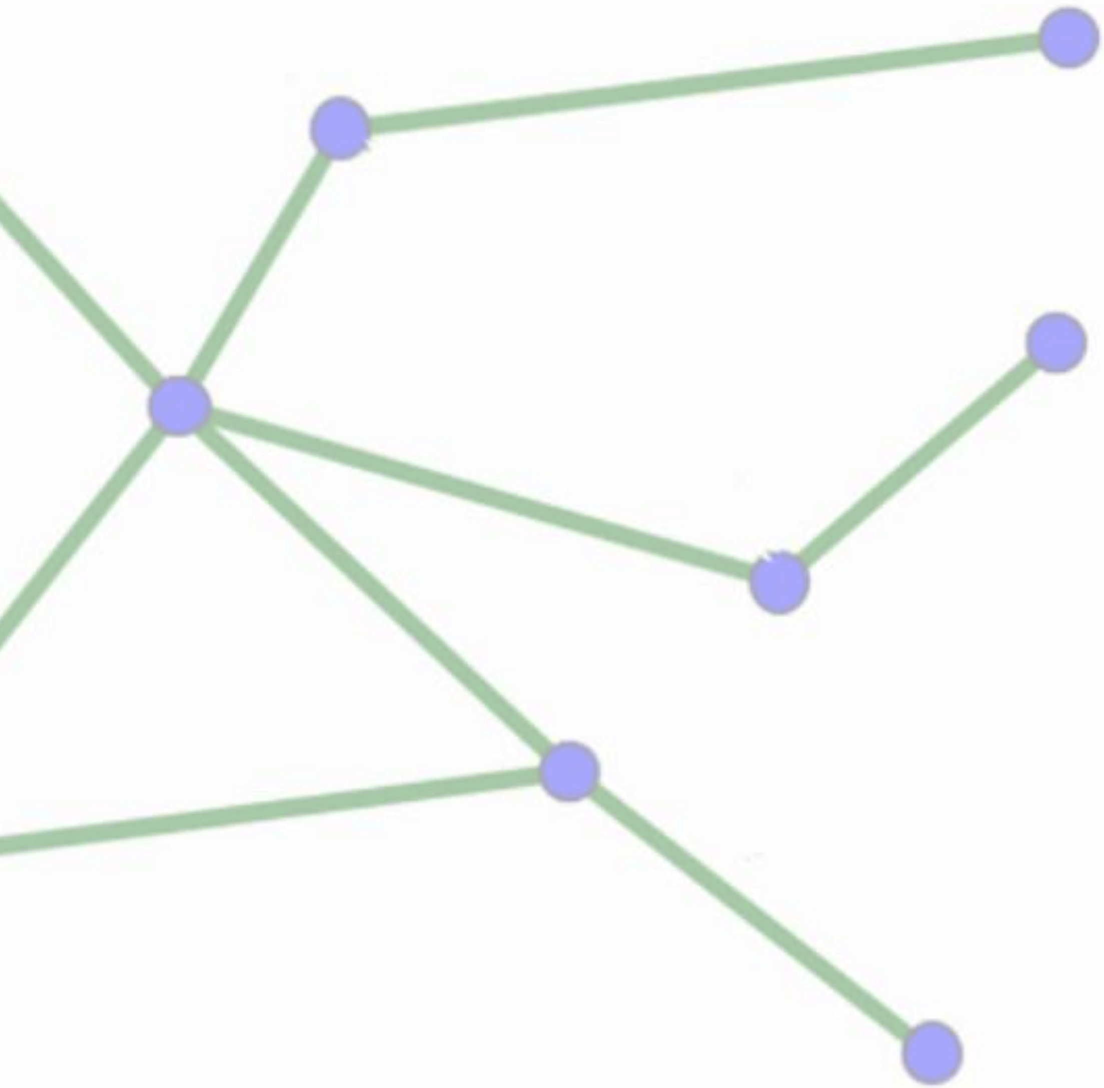
Vector solución:

[F01_x,F01_y,F12_x,F12_y,Min_1,Min_2]

Universidad de los Andes
Colombia

Dinámica Directa

Dinámica directa



Se llegó a un set de ecuaciones
Con variables:
qddot, reacciones, Fext, Mext

Conocemos:
Fext,Mext

Despejamos:
qddot, reacciones

Dinámica directa

Sistema Lineal

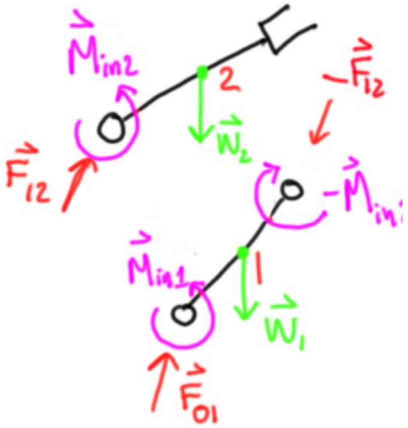
Ax=b

```python
# Our unkowns are the reactions and q accelerations and we know the external forces and moments

[A,b]=sympy.linear_eq_to_matrix(eqList,[F01_x,F01_y,F12_x,F12_y,theta1.diff(t,t),theta2.diff(t,t)])

A
```

✓ 0.0s Python



$$\left[\begin{array}{cccccc} \sin(\theta_1(t)) & \cos(\theta_1(t)) & -\sin(\theta_1(t)+\theta_2(t)) & -\cos(\theta_1(t)+\theta_2(t)) & -lc_1 m_1 \cos(\theta_1(t)) & 0 \\ -\cos(\theta_1(t)) & \sin(\theta_1(t)) & \cos(\theta_1(t)+\theta_2(t)) & -\sin(\theta_1(t)+\theta_2(t)) & -lc_1 m_1 \sin(\theta_1(t)) & 0 \\ 0 & 0 & \sin(\theta_1(t)+\theta_2(t)) & \cos(\theta_1(t)+\theta_2(t)) & m_2(-l_1\cos(\theta_1(t)) - lc_2\cos(\theta_1(t)+\theta_2(t))) & -lc_2 m_2 \cos(\theta_1(t)+\theta_2 \\ 0 & 0 & -\cos(\theta_1(t)+\theta_2(t)) & \sin(\theta_1(t)+\theta_2(t)) & -m_2(l_1\sin(\theta_1(t)) + lc_2\sin(\theta_1(t)+\theta_2(t))) & -lc_2 m_2 \sin(\theta_1(t)+\theta_2 \\ 0 & -lc_1 & -(l_1 - lc_1)\sin(\theta_2(t)) & -(l_1 - lc_1)\cos(\theta_2(t)) & -I_1 & 0 \\ 0 & 0 & 0 & -lc_2 & -I_2 & -I_2 \end{array}\right.$$

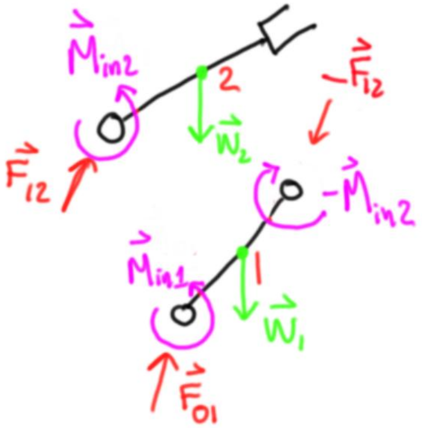
```python
b
```

Python

$$\left[\begin{array}{c} lc_1 m_1 \sin(\theta_1(t))\left(\frac{d}{dt}\theta_1(t)\right)^2 \\ -lc_1 m_1 \cos(\theta_1(t))\left(\frac{d}{dt}\theta_1(t)\right)^2 + 9.81 m_1 \\ -m_2\left(l_1\sin(\theta_1(t))\left(\frac{d}{dt}\theta_1(t)\right)^2 + lc_2\left(\sin(\theta_1(t)+\theta_2(t))\left(\frac{d}{dt}\theta_1(t)\right)^2 + 2\sin(\theta_1(t)+\theta_2(t))\frac{d}{dt}\theta_1(t)\frac{d}{dt}\theta_2(t) + \sin(\theta_1(t)+\theta_2(t))\left(\frac{d}{dt}\theta_2(t)\right)^2\right)\right) \\ m_2\left(l_1\cos(\theta_1(t))\left(\frac{d}{dt}\theta_1(t)\right)^2 + lc_2\left(\cos(\theta_1(t)+\theta_2(t))\left(\frac{d}{dt}\theta_1(t)\right)^2 + 2\cos(\theta_1(t)+\theta_2(t))\frac{d}{dt}\theta_1(t)\frac{d}{dt}\theta_2(t) + \cos(\theta_1(t)+\theta_2(t))\left(\frac{d}{dt}\theta_2(t)\right)^2\right)\right) + 9.81 m_2 \\ -\mathrm{Min}_1(t) + \mathrm{Min}_2(t) \\ -\mathrm{Min}_2(t) \end{array}\right]$$

Dinámica directa

Evaluar numéricamente (lambdify)
e.g. theta1=45°, theta2=10°, theta1dot=0.1rad/s, theta2dot=0.25rad/s, Min1=Min2=0




```
# Numerical evaluation with lambdify:
values={Min_1:0,Min_2:0,theta1.diff(t):0.1, \
    theta2.diff(t):0.25,theta1:np.deg2rad(45),theta2:np.deg2rad(10)}

A_fun=sympy.lambdify(list(values.keys()),A.subs(params))
A_num=A_fun(*list(values.values()))
b_fun=sympy.lambdify(list(values.keys()),b.subs(params))
b_num=b_fun(*list(values.values()))

np.matmul(np.linalg.pinv(A_num),b_num)
```

✓ 0.0s

```
array([[-13.81618735],
       [  6.68256361],
       [ -6.44207067],
       [  3.05369565],
       [ -4.28560647],
       [  2.75875864]])
```

Vector solución:

[F01_x,F01_y,F12_x,F12_y,theta1ddot,theta2ddot]

Discusión

Solución en un instante t_i

¿Cómo se realiza un análisis en el tiempo?