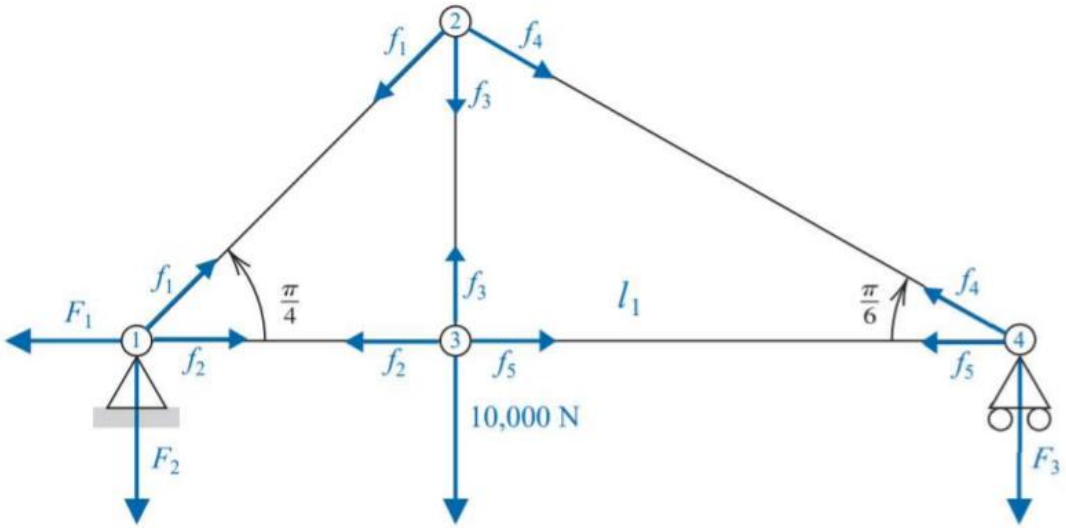
TALLER MÉTODOS NUMÉRICOS



Realizamos el balance de fuerzas en cada uno de estos nodos, donde encontraremos

Nodo 1

$$\sum F_x = f_1 \cdot \cos\left(\frac{\pi}{4}\right) + f_2 - F_1 = 0$$

$$\sum F_y = f_1 \cdot \sin\left(\frac{\pi}{4}\right) - F_2 = 0$$

Nodo 2

$$\sum F_x = f_4 \cdot \cos\left(-\frac{\pi}{6}\right) + f_1 \cdot \sin\left(\pi + \frac{\pi}{4}\right) = 0$$

$$\sum F_y = f_1 \cdot \sin\left(\pi + \frac{\pi}{4}\right) + f_4 \cdot \sin\left(-\frac{\pi}{6}\right) - f_3 = 0$$

Nodo 3

$$\sum F_x = f_5 - f_2 = 0$$

$$\sum F_y = f_3 - 10000 = 0$$

Nodo 4

$$\sum F_x = f_5 + f_2 = 0$$

$$\sum F_y = f_3 \cdot \sin\left(\pi - \frac{\pi}{6}\right) - F_3 = 0$$

Adquiriendo el sistema de ecuaciones escrito en forma matricial

$$\begin{bmatrix} \sqrt{2}/2 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ \sqrt{2}/2 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ -\sqrt{2}/2 & 0 & 0 & \sqrt{3}/2 & 0 & 0 & 0 & 0 \\ -\sqrt{2}/2 & 0 & -1 & -1/2 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & -\sqrt{3}/2 & -1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ F_1 \\ F_2 \\ F_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 10000 \\ 0 \\ 0 \end{bmatrix}$$

En este caso tenemos un sistema de la forma

$$A \cdot x = b$$

Del cual se puede emplear el método iterativo general como

$$x^{(k+1)} \coloneqq (I - B^{-1}A) \cdot x^{(k)} + B^{-1}b \quad \text{con } k = 0,1, \dots$$

Donde $x^{(0)} \in R^n$ es un vector inicial dado y $B = \left(b_{ij}\right)_{n \times n}$ es cualquier matriz no singular y sea

$$G \coloneqq I - B^{-1}A$$

Denominada matriz de iteración. Reescribiendo el método iterativo general a

$$x^{(k+1)} \coloneqq G \cdot x^{(k)} + B^{-1}b \quad \text{con } k = 0,1, \dots$$

Y se dice que convergerá si y sólo si

$$\rho(G) < 1$$

Definiendo el radio espectral $\rho(U)$ de una matriz U como

$$\rho(U) = \max_{\lambda \in S} |\lambda|$$

Donde S es el conjunto de todos los valores propios de U.

Para examinar los casos particulares del método iterativo general (MIG) podemos descomponer la matriz de coeficientes del sistema a

$$A = A_L + A_D + A_R$$

Donde

$$A_L \coloneqq \begin{pmatrix} 0 & 0 & \cdots & 0 \\ a_{21} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ a_{n1} & \cdots & a_{n,n-1} & 0 \end{pmatrix}, \qquad A_R \coloneqq \begin{pmatrix} 0 & a_{12} & \cdots & a_{1n} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & a_{n,n-1} \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

a) **Método de Jacobi.**

Para este método, que es un caso particular de MIG se debe tomar

$$B \coloneqq A_D$$

Sin embargo, es evidente que la diagonal de la matriz A tiene en sus elementos valores nulos, razón por la cual la matriz A_D es singular y no se puede emplear como matriz B, teniendo en cuenta que se necesita del cálculo de G que es función B^{-1}.

b) **Método de Gauss-Seidel.**

Para este método, que es un caso particular de MIG se debe tomar

$$B \coloneqq A_D + A_L$$

De manera análoga al punto anterior, en este caso también tendremos $\det B = 0$. Es decir que la matriz no es singular y por lo tanto no se puede emplear este método iterativo.

Código python

```
import numpy as np
import numpy.linalg as la

A = np.array([[np.sqrt(2)/2,1,0,0,0,-1,0,0],
              [np.sqrt(2)/2,0,0,0,0,0,-1,0],
              [-np.sqrt(2)/2,0,0,np.sqrt(3)/2,0,0,0,0],
              [-np.sqrt(2)/2,0,-1,-1/2,0,0,0,0],
              [0,-1,0,0,1,0,0,0],
              [0,0,1,0,0,0,0,0],
              [0,0,0,1/2,0,0,0,-1],
              [0,0,0,-np.sqrt(3)/2,-1,0,0,0]])


b = np.array([[0,0,0,0,0,10000,0,0]])
b = b.T

AL = np.zeros((8,8))
AD = np.zeros((8,8))
AR = np.zeros((8,8))
I = np.identity(8)

for i in range(0,7):
    for j in range(0,7):
        if j < i :
            AL[i,j] = A[i,j]
        elif i == j :
            AD[i,j] = A[i,j]
        elif j > i :
            AR[i,j] = A[i,j]


# Jacobi
B = AD
```

```
G = I - np.dot(la.inv(B),A)

eig_vals, eig_vecs = la.eig(G)

if max(eig_vals) > 1 :

    error = 10
    it = 0
    x = np.zeros((8,1))

    while error > 1e-5 or it < 1000:
        x = np.dot(G,x) + np.dot(la.inv(B),b)
        it = it + 1
        error = la.norm(A,2)

print(max(eig_vals))
print("El número de iteraciones es: ",it)
print("La solución es:")
print(x)

# Gauss-Seidel
B = AD + AL
G = I - la.inv(B)*A

eig_vals, eig_vecs = la.eig(G)
if max(eig_vals) > 1 :

    error = 10
    it = 0
    x = np.zeros((8,1))

    while error > 1e-5 or it < 1000:
        x = np.dot(G,x) + np.dot(la.inv(B),b)
        it = it + 1
        error = la.norm(A, 2)

print(max(eig_vals))
print("El número de iteraciones es: ",it)
print("La solución es:")
print(x)
```