Optimització de la qualitat d'una malla

Tant en modelització numèrica com en visualització, és habitual aproximar un domini continu Ω per la unió disjunta d'elements (triangles, quadrilàters, tetraedres, o altres), anomenada malla. En aquest exercici, es considera una malla composta per triangles, donada per una matriu $\mathbf{X}$, que conté les coordenades dels vèrtexs, $\mathbf{X}(i,:) = (x_i, y_i)$ per $i = 1 \ldots N$, i l'anomenada matriu de connectivitats $\mathbf{T}$. Cada fila e de la matriu de connectivitats conté els números dels nodes del triàngle e-èsim. Per exemple, per a la malla a la Figura 1 les matrius són

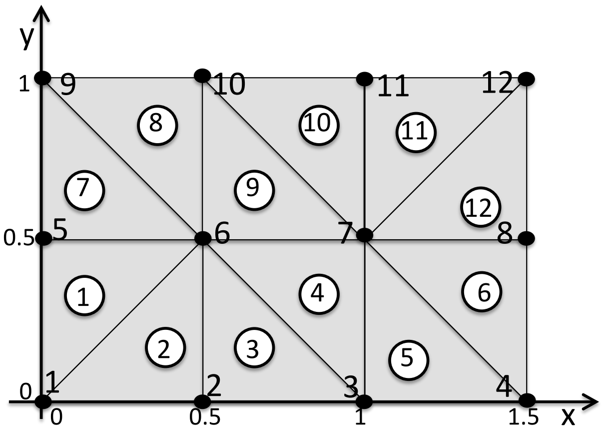

Figura 1: Malla amb 12 vèrtexs (cercles negres) i 12 elements triangulars (cercles blancs) per al domini $\Omega = (0, 1.5) \times (0, 1)$

$$\mathbf{X} = \begin{bmatrix} 0 & 0 \\ 0.5 & 0 \\ 1 & 0 \\ 1.5 & 0 \\ 0 & 0.5 \\ 0.5 & 0.5 \\ 1 & 0.5 \\ 1.5 & 0.5 \\ 0 & 1 \\ 0.5 & 1 \\ 1 & 1 \\ 1.5 & 1 \end{bmatrix} \qquad \mathbf{T} = \begin{bmatrix} 1 & 6 & 5 \\ 1 & 2 & 6 \\ 2 & 3 & 6 \\ 3 & 7 & 6 \\ 3 & 4 & 7 \\ 4 & 8 & 7 \\ 5 & 6 & 9 \\ 6 & 10 & 9 \\ 6 & 7 & 10 \\ 7 & 11 & 10 \\ 7 & 12 & 11 \\ 7 & 8 & 12 \end{bmatrix}$$

Amb aquestes definicions, $\mathbf{X}_e := \mathbf{X}(\mathbf{T}(e,:),:) \in \mathbb{R}^{3\times 2}$ és una matriu amb les coordenades dels 3 vèrtexs del triangle e-èsim.

Per a la resolució numèrica d'Equacions en Derivades Parcials (EDPs) és convenient que els elements siguin poc distorsionats; és a dir, que siguin el més semblants possible a un triangle equilàter. En aquest problema considerem la següent funció per a mesurar la distorsió d'un triangle

$$\eta_{tri}(\mathbf{X}_e) = \frac{\|\mathbf{D}\phi\|_F^2}{2|\det(\mathbf{D}\phi)|}, \tag{1}$$

on ϕ és la transformació afí del triangle equilàter de vèrtexs $(0,0)$, $(1,0)$ i $(\sqrt{3}/2, 1/2)$ al triangle de vèrtexs donats per $\mathbf{X}_e$, $\mathbf{D}\phi$ és la matriu diferencial de ϕ, i $\|\cdot\|_F$ és la norma de Frobenius (`np.linalg.norm(DPhi,'fro')` a Python). Amb aquestes definicions, la matriu diferencial $\mathbf{D}\phi$

es pot calcular com

$$\mathbf{D}\phi = \begin{pmatrix} \mathbf{X}_e(2,:) - \mathbf{X}_e(1,:) \\ \mathbf{X}_e(3,:) - \mathbf{X}_e(1,:) \end{pmatrix}^T \begin{pmatrix} 1 & -\frac{\sqrt{3}}{3} \\ 0 & \frac{2\sqrt{3}}{3} \end{pmatrix},$$

Per a mesurar la distorsió de la malla, es pot avaluar la funció

$$\eta(\mathbf{X}, \mathbf{T}) = \sqrt{\sum_{e=1}^{N_{tri}} [\eta_{tri}(\ \mathbf{X}(\mathbf{T}(e,:),:)\ )]^2}, \tag{2}$$

on N_{tri} és el nombre de triangles de la malla.

L'objectiu d'aquest exercici és, donada la malla de la Figura 2 trobar la posició dels vèrtexs interiors que minimitza la distorsió de la malla, mantenint fixos els vèrtexs del contorn. La malla es pot carregar amb la crida:

```
from mesh import X,T,Nint
```

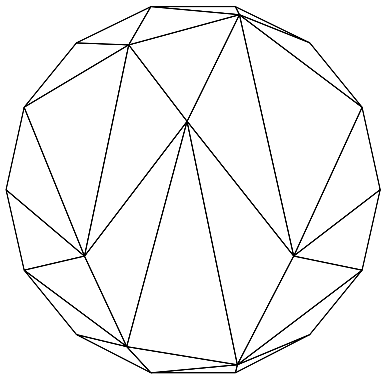

Figura 2: Malla a guardada a l'arxiu `mesh.py`

1. Completa la funció `calculaDistorsioMalla` per a calcular la distorsió (2). Quin és el valor de la distorsió de la malla inicial?

Els nodes de la malla estan numerats de manera que les `Nint` primeres files de la matriu $\mathbf{X}$ corresponen a nodes interiors. El vector d'incògnites es pot obtenir cridant

```
    y = coordsToDofs(X)
```

que agafa aquestes primeres fileres i crea un vector de graus de llibertat de dimensió $2Nint$. Observa que la funció $F : \mathbb{R}^{2Nint} \to \mathbb{R}$ definida com

```
def F(y):
    return calculaDistorsioMalla( dofsToCoords(y), T)
```

permet avaluar la distorsió per qualsevol posició dels nodes interiors (escrits com un vector $y \in \mathbb{R}^{2Nint}$). Per determinar la posició dels vèrtexs interiors que minimitza la distorsió de la malla cal minimitzar aquesta funció `F`.

2. Planteja el problema de minimitzar la funció `F` com un problema de zeros de funcions. Escriu clarament el sistema que cal resoldre (residu).

3. Si el problema es vol resoldre mitjançant el mètode de Newton-Raphson, cal fer servir la Jacobiana del residu. Com s'escriu aquesta matriu en funció de `F`?

Completa el codi de l'arxiu `main.py` per resoldre el problema plantejat mitjançant el mètode de Newton[1]. Per comprovar que el càlcul del residu i la Jacobiana són correctes, tingues en compte que prenent com a vector `y0` la posició dels nodes en la malla inicial, la primera component del vector residu i de la matriu jacobiana són

$$r[0] = 0.1457 \ \text{ i } \ J[0,0] = 6.4706.$$

4. Dibuixa la malla inicial i la malla final que has obtingut[2]. Quina és la posició del primer node interior en la malla final? I el valor de la distorsió?

5. Dibuixa la gràfica de convergència, prenent com a mida de l'error $r^k = \dfrac{\|\mathbf{y}^k - \mathbf{y}^{k+1}\|}{\|\mathbf{y}^{k+1}\|}$. Quantes iteracions han calgut per obtenir el resultat amb un error menor que $0.5 \cdot 10^{-7}$?

6. En aquest cas, s'observa que el mètode de Newton te convergència quadràtica? Per què?

[1] Pots fer servir les funcions `derivadaNumerica` i `hessianaNumerica` si necessites calcular primeres o segones derivades de `F`:

```python
from differentiation import derivadaNumerica, hessianaNumerica
```

[2] Per obtenir les coordenades finals dels nodes en funció dels graus de llibertat òptims, pot usar la funció

```python
X = dofsToCoords(y)
```