Jan Matas Cantos
i Júlia Fité Araújo

PRÀCTICA 2 - EQUACIÓ DE LA CALOR

Exercici 0:

Considerant el sistema lineal Ax=b de dimensions 8×8 donat per:

$$A = \begin{pmatrix} 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 \end{pmatrix}, \qquad \boldsymbol{b} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

contesta, utilitzant els mètodes de Jacobi, Gauss-Seidel i ω-relaxació successiva emprant com a condició inicial $x_{(0)} = (1,1,1,1,1,1,1,1)^T$ i toleràncies relatives i absolutes de 10^{-10}:

a. Quantes iteracions cal fer si utilitzem el mètode de Jacobi?

Per calcular la quantitat d'iteracions necessària si emprem el mètode de Jacobi, comencem programant una funció *jacobiIter(A, b, x, abstol, reltol, maxIter)* que implementa el mètode de Jacobi i retorna la solució x del sistema i el nombre d'iteracions que han calgut per assolir per tal que $||x^{(k+1)} - x^{(k)}|| < \varepsilon =$ *abstol* = 1.e-10 i $||x^{(k+1)} - x^{(k)}||/||x^{(k+1)}|| < \varepsilon =$ *reltol* = 1.e-10 sempre que no s'assoleixi el nombre màxim d'iteracions *maxIter*. En cas contrari, la funció retorna *x, -1* si en arribar al màxim d'iteracions no s'ha assolit la tolerància absoluta, *x, -2* si no s'ha assolit la relativa i *x, -3* si no s'han assolit cap de les dues toleràncies.

En aplicar aquesta funció a la matriu A i els vectors x_0 i b de l'enunciat, obtenim que les iteracions necessàries per resoldre el sistema mitjançant el mètode de Jacobi són 374.

b. Quantes iteracions cal fer si utilitzem el mètode de Gauss-Seidel?

Per calcular la quantitat d'iteracions necessària si emprem el mètode de Gauss-Seidel cal recordar que implementant el mètode de ω-relaxació successiva amb ω = 1 obtenim el mètode de Gauss-Seidel. Per tant, comencem programant una funció *relaxacio_w(A, b, x, w, abstol, reltol, maxIter)* que retorna la solució x del sistema i el nombre d'iteracions que han calgut per assolir per tal que ||x(k+1) − x(k)|| < ε = abstol = 1.e-10 i ||x(k+1) − x(k)||/||x(k+1)|| < ε = reltol = 1.e-10 sempre que no s'assoleixi el nombre màxim d'iteracions *maxIter*. En cas contrari, la funció retorna *x, -1* si en arribar al màxim d'iteracions no s'ha assolit la tolerància absoluta, *x, -2* si no s'ha assolit la relativa i *x, -3* si no s'han assolit cap de les dues toleràncies.

En aplicar aquesta funció a la matriu A i els vectors x_0 i b de l'enunciat amb ω = 1, obtenim que les iteracions necessàries per resoldre el sistema mitjançant Gauss-Seidel són 194.

c. Quin és el valor ω òptim (amb 3 xifres decimals) del mètode de ω-relaxació successiva? Quantes iteracions es fan amb aquest mètode per aquest valor de ω que has trobat?

Per trobar el valor òptim de ω i el nombre d'iteracions necessàries per al valor de ω en qüestió implementem el següent codi, en què w_opt és la ω òptima i iter_relaxació_w el nombre d'iteracions:

```
eps = np.float64(1E-3)
l, r = np.float64(0 + eps), np.float64(2 - eps)
while r - l > eps:
    m1 = l + (r - l) / 3
    m2 = r - (r - l) / 3
    x1, iter1 = relaxacio_w(A, b, x0, m1, maxIter=1000)
    x2, iter2 = relaxacio_w(A, b, x0, m2, maxIter=1000)
    if iter1 > iter2:
        l = m1
    else:
        r = m2
w_opt = np.round(np.float64((l + r) / 2), 3)
x_relaxacio_w, iter_relaxacio_w = relaxacio_w(A, b, x0, w_opt, maxIter=1000)
```

Aprofitant que la funció d'iteracions de w_opt és convexa, el codi anterior permet trobar el valor òptim de ω pel mètode de relaxació successiva a través d'una cerca ternària, és a dir, dividint l'interval inicial (0,2) en tres parts en cada iteració, comparant el nombre d'iteracions necessàries per convergir per dos punts m_1 i m_2, i seleccionant el subinterval que conté el mínim. Aquest procés es repeteix fins a assolir una precisió epsilon = 1.e-3.

Finalment es calcula l'ω òptim com el punt mitjà de l'interval resultant i es resol el sistema amb aquest ω, obtenint la solució i el nombre d'iteracions necessàries per convergir amb aquest valor òptim.

Jan Matas Cantos
i Júlia Fité Araújo

Exercici 1:

a. Implementeu una funció que, donat els valor de l'interval [a,b], els valors de la solució a la frontera c_a i c_b, un valor N, una funció F, un valor optatiu *tol* i un número màxim d'iteracions *maxIter*, resolgui l'equació estacionària 1D

$$\begin{cases} u_{xx}(x) = -F(x), & \text{per } x \in [a,b] \\ u(a) = c_a, \\ u(b) = c_b. \end{cases}$$

utilitzant el mètode de Jacobi i les operacions vectoritzades de python:

Programem la funció *solEstEq1D(a, b, cA, cB, N, F, maxIter, tol)*, que retorna la solució *u* obtinguda i el nombre *niter* d'iteracions realitzades si s'ha assolit la tolerància en el nombre màxim d'iteracions o el nombre d'iteracions realitzades amb signe negatiu en cas contrari:

```
def solEstEq1D(a, b, cA, cB, N, F, maxIter=1.e5, tol=1.e-10):
    u = (cA + cB)/2 * np.ones(N + 1)
    u[0], u[N] = cA, cB
    h = (b - a)/N
    for niter in range(1, int(maxIter)+1):
        unou = u.copy()
        i = np.arange(1, N)
        unou[i] = (u[i - 1] + u[i + 1] + F(a + h*i)*h*h)/2
        dif = unou - u
        u = unou
        if np.max(dif) < tol:
            return u, niter
    return u, -maxIter
```

b. Resol el problema:

$$\begin{cases} \delta u(x) = -\sin^2(x), & \text{per } x \in [0, 2\pi] \\ u(0) = 0, \\ u(2\pi) = 1. \end{cases}$$

Emprant N=100, quin és el valor aproximat de u(π)?

Apliquem la funció *solEstEq1D(a, b, cA, cB, N, F, maxIter, tol)* de l'apartat anterior amb N = 100, a = 0, b = 2π, c_a = 0, c_b = 1 i f = -sin^2(x).

Observem que si N = 100 i b - a = 2π, aleshores l'índex apropiat és 50. Consultant el valor de u[50], obtenim que el valor aproximat de u(π) que estem buscant és 0.22197966271872244.

Jan Matas Cantos
i Júlia Fité Araújo

Exercici 2:

Calcula la temperatura mitjana de la PC2 utilitzant un model 3 dimensional de l'FME:

Comencem definint la funció *solEq3D(dim, T, maxIter, tol)*. Inicialment, aquesta funció estableix com a condició inicial que la temperatura en tot l'interior és de 25.5 graus i defineix explícitament la temperatura de les fronteres (assignant els valors de temperatures extrets del document *TemperaturaVora.pdf*). A part, també es defineixen les temperatures dels nodes a les interseccions de les fronteres com la mitjana de les temperatures corresponents.

A continuació, s'inicia un bucle que a cada iteració aplica el mètode de Jacobi actualitzant la temperatura de cadascun dels nodes interiors com la mitjana de temperatures dels seus "veïns". Seguidament, es corregeixen els errors comesos en les temperatures de les parets del pati i les seves interseccions, tornant a definir-les explícitament com a l'inici.

Aquest bucle s'atura i retorna *u* (la matriu en tres dimensions que conté la informació dels nodes) i *niter* (el nombre d'iteracions fetes) quan la diferència màxima entre dues iteracions consecutives és menor que el valor de tolerància definit inicialment (en aquest cas amb *tol* = 1.e-8). A partir de l'*u* retornat en el moment que s'assoleix la convergència, calculem la mitjana de les temperatures dels nodes que pertanyen a l'aula PC2.

Finalment, hem necessitat 1058 iteracions per assolir la temperatura mitjana de l'aula PC2, que ha resultat ser de 20.985754114469884 graus.