



Nell'equazione precedente, se  $\eta \geq 0$  si sceglie la radice  $t = 1/(\eta + \sqrt{1 + \eta^2})$  altrimenti  $t = -1/(-\eta + \sqrt{1 + \eta^2})$ . Quindi  $c$  e  $s$  risultano

$$c = \frac{1}{\sqrt{1 + t^2}}, \quad s = ct.$$

La convergenza del metodo si verifica calcolando la seguente quantità, valida per una generica matrice  $M$

$$\Phi(M) = \left( \sum_{i,j=1,i \neq j}^n m_{ij}^2 \right)^{1/2} = \left( \|M\|_F^2 - \sum_{i=1}^n m_{ii}^2 \right)^{1/2}. \quad (4)$$

Il metodo garantisce che  $\Phi(A^{(k)}) \leq \Phi(A^{(k-1)})$ ,  $\forall k$ .

Una strategia ottimale di scelta degli indici  $p, q$  tale che  $\Phi(A^{(k)}) \leq \Phi(A^{(k-1)})$  sia sempre verificata, è quella di scegliere l'elemento di  $A^{(k-1)}$  tale che

$$|a_{pq}^{(k-1)}| = \max_{i \neq j} |a_{ij}^{(k-1)}|.$$

Una M-function che implementa il metodo di Jacobi, data  $A$  e una tolleranza  $tol$  e che restituisce la matrice diagonale  $D$  degli autovalori, il numero di iterazioni effettuate e la quantità  $\Phi(\cdot)$ , potrebbe essere come segue.

```
function [D,iter,phi]=symJacobi(A,tol)
%-----
% Data la matrice simmetrica A e una tolleranza tol, determina,
% mediante il metodo di Jacobi tutti i suoi autovalori che memorizza
% nella matrice diagonale D. Determina inoltre il numero di iterazioni,
% iter, e la quantit phi che contiene la norma degli elementi extra-diagonali
% e come converge il metodo.
%-----
n=max(size(A)); D=A; phiD=norm(A,'fro'); epsi=tol*phiD;
phiD=phiNorm(D); iter=0; [phi]=phiD;
while (phiD > epsi)
    iter=iter+1;
    for p=1:n-1,
        for q=p+1:n
            [c,s]=calcoloCeS(D,p,q);
            D=calcoloProdottoGtDG(D,c,s,p,q);
        end;
    end
    phiD=phiNorm(D);
    phi=[phi; phiD];
end
return
```

1. Data la matrice di Hilbert di ordine 4 (in Matlab `hilb(4)`), i cui autovalori (arrotondati) sono  $\lambda_1 = 1.5002$ ,  $\lambda_2 = 0.1691$ ,  $\lambda_3 = 0.0067$ ,  $\lambda_4 = 0.0001$ . Calcolare detti autovalori usando il metodo di Jacobi con una tolleranza `tol=1.e-15`. In quante iterazioni converge il metodo? Calcolare anche ad ogni iterazione la quantità  $\Phi(A^{(k)})$  per verificare che decresce. (Sugg. Usare l'M-function `symJacobi` fornita: implementare le M-functions `calcoloCeS` e `calcoloProdottoGtDG`).

◇ ◇ ◇

## 2 Il metodo delle successioni di Sturm

Sia  $A \in \mathbb{R}^{n \times n}$  tridiagonale simmetrica. Indichiamo con  $\mathbf{d}$  e  $\mathbf{b}$  i vettori diagonale e extradiagonali di dimensioni  $n$  e  $n - 1$ , rispettivamente.

Sia  $A_i$  il minore principale di ordine  $i$  di  $A$ . Posto  $p_0(x) = 1$ , indichiamo con  $p_i(x) = \det(A_i - xI_i)$ . Si ha

$$\begin{aligned} p_1(x) &= d_1 - x; \\ p_i(x) &= (d_i - x)p_{i-1}(x) - b_{i-1}^2 p_{i-2}(x), \quad i = 2, \dots, n. \end{aligned} \quad (5)$$

Alla fine  $p_n(x) = \det(A)$ . La successione  $\{p_i(x)\}$  è detta una *successione di Sturm*. Vale il seguente fatto: **per  $i=2, \dots, n$  gli autovalori di  $A_{i-1}$  separano quelli di  $A_i$** , ovvero

$$\lambda_i(\mathbf{A}_i) < \lambda_{i-1}(\mathbf{A}_{i-1}) < \lambda_{i-1}(\mathbf{A}_i) < \dots < \lambda_2(\mathbf{A}_i) < \lambda_1(\mathbf{A}_{i-1}) < \lambda_1(\mathbf{A}_i). \quad (6)$$

Inoltre, per ogni reale  $\nu$ , definiamo

$$S_\nu = \{p_0(\nu), p_1(\nu), \dots, p_n(\nu)\}. \quad (7)$$

Allora  $s(\nu)$ , **il numero di cambiamenti di segno in  $S_\nu$ , indica il numero di autovalori di  $A$  strettamente minori di  $\nu$** .

Da un punto di vista implementativo, per costruire  $A$ , noti i vettori  $\mathbf{d}$  e  $\mathbf{b}$ , basta usare il comando Matlab `A=diag(d)+diag(b,1)+diag(b,-1)`. Quindi si può procedere come segue:

- Scelgo un valore reale  $\nu$  e costruisco l'insieme  $S_\nu$ . Qui bisogna implementare le formule (5), ottenendo in output un array che contiene i numeri  $p_i(\nu)$ ,  $i = 0, 1, \dots, n$ .
- Determino il numero  $s(\nu)$  che mi dirà, grazie alla (6), quanti autovalori di  $A$  sono minori in senso stretto, di  $\nu$ .
- Esiste un metodo, detto *metodo di Givens*, per determinare tutti gli autovalori di una matrice  $A$  tridiagonale simmetrica. Poniamo  $b_0 = b_n = 0$  allora l'intervallo  $I = [\alpha, \beta]$  con

$$\alpha = \min_{1 \leq i \leq n} (d_i - (|b_i| + |b_{i-1}|)), \quad \beta = \max_{1 \leq i \leq n} (d_i + (|b_i| + |b_{i-1}|)), \quad (8)$$

conterrà tutti gli autovalori di  $A$  (infatti  $\alpha$  e  $\beta$  indicano gli estremi dell'intervallo di Gerschgorin).

Per determinare l' $i$ -esimo autovalore di  $A$ ,  $\lambda_i$ , una volta determinato  $I = [\alpha, \beta]$ , con il metodo di bisezione lo si determina operando come segue. Si costruiscono le successioni  $a^{(i)}$  e  $b^{(i)}$  come segue:  $a^{(0)} = \alpha$ ,  $b^{(0)} = \beta$ , quindi si calcola  $c^{(0)} = (a^{(0)} + b^{(0)})/2$ , **grazie alla proprietà (6)**, se  $s(c^{(0)}) > n - i$  allora  $b^{(1)} = c^{(0)}$  altrimenti  $a^{(1)} = c^{(0)}$ . Si continua finchè ad un passo  $k^*$ ,  $|b^{(k^*)} - a^{(k^*)}| \leq \text{tol}(|a^{(k^*)}| + |b^{(k^*)}|)$ .

Procederemo poi in modo sistematico per calcolare tutti gli altri autovalori.

1. Data la matrice tridiagonale avente diagonale principale il vettore  $\mathbf{d}=\mathbf{ones}(1,n)$  e sopra e sotto diagonali il vettore  $\mathbf{b}=-2*\mathbf{ones}(1,n-1)$ . Con  $n = 4$ , calcolarne tutti gli autovalori usando il metodo di Givens delle successioni di Sturm. Per facilitare l'implementazione si descrive l'M-function che determina la successione di Sturm e i suoi cambiamenti di segno.

```
function [p,cs]=succSturm(d,b,x)
%-----
% Calcolo della successione di Sturm 'p' in x
% a partire dai vettori d e b
% e dei corrispondenti cambiamenti di segno 'cs'
%-----
n=length(d); p(1)=1; p(2)=d(1)-x; for i=2:n,
    p(i+1)=(d(i)-x)*p(i)-b(i-1)^2*p(i-1);
end
cs=0; %contatore cambi di segno
s=0; %contatore dei segni costanti
for i=2:length(p),
    if(p(i)*p(i-1)<=0),
        cs=cs+1;
    end
    if(p(i)==0),
        s=s+1;
    end
end
cs=cs-s;
return
```

**Tempo massimo: 2 ore.**