6. AUTOVALORI E AUTOVETTORI

Data una matrice quadrata A si definiscono autovalori ed autovettori rispettivamente quei valori reali o complessi e quei vettori non nulli che verificano l'equazione

$$Au = \lambda u \qquad [6.1]$$

Molte matrici possono essere ridotte ad una matrice diagonale tramite una decomposizione

$$A=V^{-1}DV \qquad [6.2]$$

A e D si dicono "simili", e vale che gli auto valori di A sono uguali a quelli di D: $\lambda(A) = \lambda(D)$.

Purtroppo non tutte le matrici sono diagonalizzabili. Esiste però sempre un'altra decomposizione che vale per matrici generiche:

$$A=MJM^{-1} \qquad [6.3]$$

dove J è detta "matrice di Jordan" ed ha la forma di una matrice diagonale a blocchi.

Gli autovalori di A sono allora quelli disposti lungo la diagonale di J. È possibile che lungo la diagonale un autovalore compaia più volte. Diamo allora le seguenti definizioni:

Def-6.1: si definisce "molteplicità geometrica (GM)" il numero di blocchi di Jordan in cui compare un autovalore.

Def-6.2: si definisce "molteplicità algebrica (AM)" il numero di volte che un dato autovalore compare sulla diagonale.

Esse ci consentono di scrivere il seguente:

Teorema-6.1: il numero di autovettori linearmente indipendenti associati ad un dato autovalore è dato dalla differenza tra la molteplicità algebrica e la molteplicità geometrica, cioè AM-GM.

Se un dato autovalore ha GM minore di AM, esso si dice "difettivo". Una matrice che possiede uno o più autovalori difettivi si dice anch'essa "matrice difettiva"

Esercizio 6.1

Sia data la seguente matrice di Jordan 4x4

$$J = \begin{bmatrix} 100 & & & \\ & 5 & & \\ & & 4 & \\ & & & 3 \end{bmatrix} \qquad [6.4]$$

ed $A=M^{-1}JM$ con M qualsiasi purché non singolare.

Si calcolino:

1) tutti gli autovalori di A;
2) l'autovalore di modulo massimo di A usando il metodo delle potenze dirette;
3) l'autovalore di modulo massimo di A^{-1} usando il metodo delle potenze dirette.

---

1) Possiamo definire J ed A in MATLAB:

```
>> J=diag([100,5,4,3])
>> M=rand(4);
>> A=M\J*M;
```

Gli autovalori sono calcolabili tramite il comando

```
>> eig(A)
```

2) Ordiniamo i quattro autovalori secondo la grandezza decrescente dei moduli; chiameremo quindi λ_1 l'autovalore di modulo massimo e λ_4 quello di modulo minimo. A questi, sono associati i corrispondenti autovettori.
Il metodo delle potenze dirette si basa su questa serie:

$$\begin{cases} x_{k+1} = Ax_k \\ x_0 \end{cases} \qquad [6.5]$$

Si dimostra che:

$$\lim_{x\to+\infty} x_k = u_1 \qquad [6.6]$$

cioè la serie converge all'autovettore associato all'autovalore di modulo massimo. In particolare, l'autovalore associato si calcola come:

$$\lambda_1 = \frac{{x_k}^T A x_k}{{x_k}^T x_k} \qquad [6.7]$$

Possiamo utlizzare la routine MATLAB potenze per implementare il metodo. Tale routine ha la seguente sintassi:

```
>>  [lam1,x1,j]=potenze(A,x0,n°_max_iterazioni,tolleranza)
```

e come si vede restituisce il valore di λ_1, indicando anche dopo quante iterazioni vi è giunto.

È possibile calcolare il fattore di abbattimento dell'errore come:

$$r = \frac{|\lambda_2|}{|\lambda_1|} \qquad [6.8]$$

r è inversamente proporzionale alla velocità di convergenza del metodo.

3) Il metodo delle potenze inverse permette di calcolare l'autovalore di modulo massimo della matrice inversa di A, e analogamente quello di modulo minimo di A, tenuto conto del fatto che $\lambda(A^{-1}) = \frac{1}{\lambda(A)}$

Il metodo utilizza questa serie:

$$\begin{cases} Ax_{k+1} = x_k \\ x_0 \end{cases} \qquad [6.5]$$

In MATLAB, possiamo usare la routine:

```
>>  [lam1,x1,j]=potinv(A,x0,fattore_accelerazione,n°_max_iterazioni,tolleranza)
```

Il fattore di accelerazione è normalmente posto a zero. In altri casi può essere calcolato (si veda il prossimo esercizio), permettendo di accelerare il metodo e giungere più velocemente al risultato.

Se calcoliamo tramite A^{-1} il fattore $r_{p.i.}$, notiamo che nel caso di potenze inverse esse è più piccolo di quello calcolato con il metodo delle potenze dirette. Pertanto la velocità di convergenza è inferiore e quindi il metodo delle potenze inverse è più lento (cioè richiede un numero maggiore di iterazioni per giungere al risultato). L'inserimento del fattore di accelerazione opportuno può migliorare la situazione e quindi ridurre $r_{p.i.}$.

Osserviamo anche che il metodo delle potenze inverse converge tanto più velocemente quanto più distanti sono fra loro in modulo gli autovalori contenuti sulla diagonale della matrice di Jordan. Se gli autovalori sono simili in modulo, il metodo risulta più oneroso in termini di iterazioni.

Esercizio 6.2
Si consideri il polinomio

$$p(\lambda) = det(\lambda I - A) \qquad [6.6]$$

che ha radici (10,7,4,2,1).

Come si osserva dalla sua forma, le radici di $p(\lambda)$ non sono altro che gli autovalori della matrice A.

Si chiede di:

1) costruire la matrice A che ha $p(\lambda)$ come polinomio caratteristico;
2) calcolare l'autovalore di modulo minimo di A usando il metodo delle potenze inverse, e determinare il fattore $r_{p.i.}$;
3) accelerare il metodo calcolando un giusto fattore di accelerazione.

---

1) Inseriamo le radici in un vettore v:

```
>> v=[10,7,4,2,1]
```

Calcoliamo i coefficienti del polinomio caratteristico usando la routine poly:

```
>>coeff_p=poly(v)
```

Tale routine prende il vettore v e genera il polinomio come:

$$p(\lambda) = (x-10)(x-7)(x-4)(x-2)(x-1) \qquad [6.7]$$

Dopo le varie moltiplicazioni, il nostro polinomio avrà una generica forma del tipo:

$$p(\lambda) = a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0 \qquad [6.8]$$

La matrice A richiesta dall'esercizio è detta "companion matrix", e può essere costuita in MATLAB con il comando:

```
>> A=compan(coeff_p)
```

Se si calcolano gli autovalori dei questa matrice, essi sono proprio le radici di $p(\lambda)$. Infatti la companion matrix è quella che possiede $p(\lambda)$ come polinomio caratteristico.

2) Occorre eseguire la routine `potinv`. Si osservi che comunque è banale calcolare $\lambda(A^{-1}) = \left[1, \frac{1}{2}, \frac{1}{4}, \frac{1}{7}, \frac{1}{10}\right]$ e dunque $r_{p.i.} = \frac{1}{2}$

3) Per calcolare il fattore di accelerazione α, occorre eseguire la routine `potinv` a bassa tolleranza (inserendo p.es. il valore 1e-1). Il valore `lam1` restituito da `potinv` è utilizzabile come fattore di accelerazione α . Vale allora:

$$\lambda(A - \alpha I) = \lambda(A) - \alpha \qquad [6.9]$$

Nel nostro esempio, con i nostri numeri, troviamo $\alpha = 0.9$. Dobbiamo eseguire uno shifting pari ad αsugli autovalori di A per trovare gli autovalori con il metodo accelerato: [10-0.9,7-0.9,4-0.9,2-0.9,1-0.9].

Se uso l'inversa di $(A - \alpha I)$, gli autovalori di $\lambda((A - \alpha I)^{-1})$ saranno [10,0.1,……]. Allora, calcolando $r_{p.i.} = \frac{0.1}{10} = 0.01$, osservo come il fattore di accelerazione abbia aumentato la velocità di convergenza, riducendo il fattore di abbattimento dell'errore rispetto al caso non accelerato.

Metodo globale (QR)

Data la matrice $A \epsilon R^{mxn}$, poniamo $A_0 = A$ e calcoliamo ricorsivamente

$$\begin{cases} A_k = Q_k R_k \\ A_{k+1} = R_k Q_k \end{cases} \qquad [6.10]$$

È dimostrato che

$$\lim_{x \to +\infty} A_k = R \qquad [6.11]$$

cioè il metodo converge ad una matrice triangolare alta.

Supponiamo di avere una matrice A. Per prima cosa le diamo forma di Hessemberg:

```
>>A=hess(A)
```

Per applicare il metodo occorre iterare un po' di volte i seguenti passi:

```
>> [Q,R]=qr(A);
>>A1=R*Q;
>>[Q,R]=qr(A1);
>>A2=R*Q;
ecc…ecc…
```

Dopo un po' si dovrebbe giungere ad una matrice che ha l'elemento in ultima riga/penultima colonna che è molto prossimo a zero. Quando questo succede, allora l'elemento in ultima riga/ultima colonna è l'autovalore di modulo massimo. Se questo invece non accade, cioè se l'elemento in ultima riga/penultima colonna non si approssima mai a zero, allora occorre considerare tutto il blocchetto 2x2 più in basso a destra e calcolarne gli autovalori: questi saranno quello di modulo massimo e quello immediatamente più piccolo.