

Laboratorio di Analisi Numerica

Lezione 6

Gianna Del Corso <delcorso@di.unipi.it>

Federico Poloni <fpoloni@di.unipi.it>

6 Novembre 2012

Quantità di esercizi: in questa dispensa ci sono *più esercizi* di quanti uno studente medio riesce a farne durante una lezione di laboratorio, specialmente tenendo conto anche degli esercizi facoltativi. Questo è perché sono pensate per “tenere impegnati” per tutta la lezione anche quegli studenti che già hanno un solido background di programmazione. Quindi fate gli esercizi che riuscite, partendo da quelli *non* segnati come facoltativi, e non preoccupatevi se non li finite tutti!

1 Metodi di Jacobi e Gauss–Seidel

1.1 Matrici di test

Inserire in Octave le seguenti matrici.

$$A1 = \begin{bmatrix} 3 & 0 & 4 \\ 7 & 4 & 5 \\ -1 & -2 & 2 \end{bmatrix}, \quad A2 = \begin{bmatrix} -3 & 3 & -6 \\ -4 & 7 & -8 \\ 5 & 7 & -9 \end{bmatrix},$$
$$A3 = \begin{bmatrix} 4 & 1 & 1 \\ 2 & -9 & 0 \\ 0 & -8 & -6 \end{bmatrix}, \quad A4 = \begin{bmatrix} 7 & 6 & 9 \\ 4 & 5 & -4 \\ -7 & -3 & 8 \end{bmatrix}.$$

Vogliamo verificare che

- Su $A1$, il metodo di Jacobi non converge ma quello di Gauss–Seidel sì;
- Su $A2$, Jacobi converge (piano) ma Gauss–Seidel no;
- Su $A3$, convergono entrambi e Gauss–Seidel è più veloce;
- Su $A4$, convergono entrambi (piano) e Jacobi è più veloce.

1.2 Jacobi

La formula che definisce il metodo di Jacobi è

$$x_i^{(new)} = \frac{1}{A_{ii}} \left(b_i - \sum_{j=1}^{i-1} A_{ij} x_j^{(old)} - \sum_{j=i+1}^n A_{ij} x_j^{(old)} \right), \quad i = 1, \dots, n.$$

Esercizio 1. Scrivere una funzione `function x=jacobi(A,b,k)` che esegue `k` passi del metodo di Jacobi. Dopo ogni passo, scrivere sullo schermo la norma-2 del *residuo* con l'istruzione `norm(A*x-b)` (senza punto e virgola finale).

Suggerimento: attenzione che non potete “riutilizzare” la variabile `x` in un ciclo del tipo

```
for i=1:n
    ...
    x(i) = (formule che coinvolgono elementi di x);
    ...
endfor
```

perché in questo modo dopo la prima iterazione il vecchio valore di `x(1)` va perso, ma a voi serve ancora nelle formule che calcolano i nuovi `x(2)`, `x(3)`, ... Dovrete invece fare qualcosa del tipo

```
x_new=zeros(n,1);
for i=1:n
    ...
    x_new(i) = (formule che coinvolgono elementi di x);
    ...
endfor
x=x_new;
```

Testare la funzione sulle quattro matrici di test A_1, \dots, A_4 e un termine noto b a piacere (per esempio il vettore di tutti uni): dovrebbe venire qualcosa simile all'output qui sotto. I numeri precisi calcolati da voi possono cambiare (dipende da un paio di scelte che potete fare nell'algoritmo), però il comportamento della successione (va a zero? Quanto velocemente?) dovrebbe corrispondere.

```
octave:140> jacobi(A1,ones(3,1),10)
ans = 9.5000
ans = 33.083
ans = 52.250
ans = 45.750
ans = 177.34
ans = 107.49
ans = 438.86
ans = 491.67
ans = 746.34
ans = 1966.4
```

```

ans =

    149.5063
    218.0709
     9.7088

octave:141> jacobi(A2,ones(3,1),10)
ans = 7.5556
ans = 5.4921
ans = 4.1623
ans = 0.73637
ans = 0.63473
ans = 1.3142
ans = 0.39427
ans = 0.60966
ans = 0.47145
ans = 0.31303
ans =

    0.0049121
   -0.1289996
   -0.2012020

octave:142> jacobi(A3,ones(3,1),10)
ans = 7.1111
ans = 2.2222
ans = 1.5062
ans = 0.40329
ans = 0.24829
ans = 0.13397
ans = 0.029064
ans = 0.025834
ans = 0.010817
ans = 0.0028290
ans =

    0.287301
   -0.046952
   -0.104023

octave:143> jacobi(A4,ones(3,1),10)
ans = 23.400
ans = 12.514
ans = 4.3015
ans = 3.6876

```

```

ans = 2.8242
ans = 1.0743
ans = 0.75329
ans = 0.68841
ans = 0.29360
ans = 0.25730
ans =
-0.261799
0.430087
0.056597

```

I risultati non sono molto significativi perché 10 iterazioni sono molto poche, provare per esempio con $k = 50$. Quando c'è convergenza? Quante iterazioni servono perché il residuo scenda sotto la soglia $\varepsilon = 10^{-8}$?

1.3 Gauss–Seidel

La formula che definisce il metodo di Gauss–Seidel è invece

$$x_i^{(new)} = \frac{1}{A_{ii}} \left(b_i - \sum_{j=1}^{i-1} A_{ij} x_j^{(new)} - \sum_{j=i+1}^n A_{ij} x_j^{(old)} \right), \quad i = 1, \dots, n.$$

Notate che l'unica cosa che cambia rispetto a Jacobi è il *(new)* in rosso.

Esercizio 2. Scrivere una funzione `function x=gaussseidel(A,b,k)` che esegua k passi del metodo di Gauss–Seidel. Dopo ogni passo, scrivere sullo schermo la norma-2 del residuo con l'istruzione `norm(A*x-b)` (senza punto e virgola finale).

Suggerimento: come prima, aggiornate gli elementi $1, 2, 3, \dots$ in quest'ordine. Ad ogni passo k , nell'equazione che corrisponde alla riga k -esima della matrice c'è solo un elemento incognito $x_k^{(new)}$, quindi potete calcolarlo risolvendo l'equazione. In questo modo evitate di dover risolvere il sistema triangolare superiore con `sup_solve...`

Testare il metodo sulle quattro matrici di test. Quante iterazioni servono per ognuna delle matrici perché il residuo scenda sotto $\varepsilon = 10^{-8}$?

Esercizio 3. Testare i due metodi sulla matrice

$$A5 = \begin{bmatrix} 3 & 0 & 4 \\ 7 & 4 & 3 \\ -1 & 1 & 0 \end{bmatrix}.$$

```

octave:147> jacobi(A5,ones(3,1),5)
warning: in /.automount/homeserver/root/RAID/5/home/p/poloni
/prova/jacobi.m near line 13, column 14:
warning: division by zero
ans = Inf

```

```
warning: division by zero
ans = NaN
warning: division by zero
ans = NaN
warning: division by zero
ans = NaN
warning: division by zero
ans = NaN
ans =
      NaN
      NaN
      NaN
```

Come mai i metodi non funzionano?

Esercizio 4. Quale dei due metodi è implementato nella funzione qui sotto? Perché? A seconda di come avete implementato i due metodi, forse ve ne siete già accorti.

```
function x=mystery(A,b,k)
%esegue k passi di uno tra Jacobi e Gauss-Seidel: quale?
s=size(A);
n=s(1);
x=b %o qualunque altro valore iniziale
for it=1:k
  for i=1:n
    t=b(i); %accumulatore
    for j=1:n
      if(i~=j) %ricordate che ~= e' il simbolo di "diverso"
        t=t-A(i,j)*x(j);
      endif
    endfor
    x(i,1)=t/A(i,i);
  endfor
  norm(A*x-b) %stampa la norma del residuo --- senza punto e virgola
endfor
endfunction
```

1.4 Esercizi facoltativi

Esercizio 5 (facoltativo). Scrivere due funzioni `x=jacobi_opt(A,b)` e `x=gaussseidel_opt(A,b)` che eseguono un numero variabile di iterazioni dei due metodi, fermandosi quando il residuo `norm(A*x-b)` scende sotto $\varepsilon = 10^{-8}$. Ricordarsi che è buona norma inserire anche un numero massimo di iterazioni, ad esempio `max_iter=100` in modo che la procedura si arresti nel caso non ci sia convergenza o la convergenza sia troppo lenta.

Esercizio 6 (facoltativo). Modificare le funzioni viste introducendo tra i parametri di input il vettore iniziale x_0 . Cosa succede se applichiamo il metodo di Jacobi alla matrice

$$A = \begin{bmatrix} 3 & 1 & 1 \\ -2 & 3 & 1 \\ 2 & 2 & -1 \end{bmatrix},$$

con vettore $b = (5, 2, 3)$, a partire dal vettore

$$x_0 = [3.338379432298626, -1.590892516099053, 4.580399083988357]'$$

Cosa succede invece se prendiamo un vettore di partenza casuale? Sapete spiegare questo comportamento?

Esercizio 7 (facoltativo). Cosa succede se applichiamo i metodi di Jacobi o Gauss-Seidel ad una matrice singolare?

Esercizio 8 (facoltativo). Come si può rimediare all'inconveniente che si presenta nell'esercizio 3? Diverse risposte sono possibili. . .

Esercizio 9 (facoltativo). Consideriamo la matrice L_n di dimensione $n \times n$ che ha 2 sulla diagonale, -1 sulla sopradiagonale e sottodiagonale, e zeri altrove (già vista nelle lezioni scorse):

$$L_5 = \begin{bmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix}.$$

Riuscite a utilizzare la struttura di questa matrice per implementare i metodi di Jacobi e Gauss-Seidel facendo meno calcoli rispetto al caso generale? Scrivete due funzioni `function x=jacobi_lap(n,b)` e `function x=gausseidel_lap(n,b)` che calcolino la soluzione del sistema $L_n x = b$ con i due metodi. Quale è più veloce?