Calcolo Numerico. Ingegneria Industriale. Canali 2 e 5.

Esercitazione 5: Soluzione iterativa di sistemi lineari.

Implementazione del metodo di Jacobi

Il file `jacobi.m` dovrà contenere una function simile a quella riportata sotto:

```
function [xnew,iter,vscarti]=jacobi(A,b,x0,itmax,toll)
%Metodo iterativo di JACOBI [x,iter,vscarti]=jacobi(A,b,x0,itmax,toll)
[n,m] = size(A);
% vettore iniziale x = x_0
xold = x0;
% partizionamento della matrice
D = diag(diag(A));
L = tril(A,-1);
U = triu(A,1);
M = D;
N = -(L +U);
% norme
vscarti= [];
scarto = 1;
iter = 0;
% ciclo iterativo M x_k+1 = N x_k + b;
while scarto > toll & iter < itmax
    tnoto = N*xold + b;
    xnew = M\tnoto;
    scarto = norm(xnew-xold);
    vscarti = [vscarti;scarto];
    iter = iter + 1;
    xold = xnew;
end
```

Commenti. La function riceve in input la matrice, il termine noto, il vettore iniziale, il numero massimo di iterazioni, e la tolleranza. In output restituisce: il vettore soluzione, il numero di iterazioni impiegate e il vettore delle norme degli scarti ad ogni iterazione.

Il test di arresto è basato sulla norma euclidea dello scarto. Per questo motivo si rende necessario l'utilizzo di due vettori, `xnew` e `xold` che rappresentano, rispettivamente, $x^{(k+1)}$ e $x^{(k)}$. Riflettere a questo proposito sul significato e sull'utilità dell' ultima istruzione del ciclo: `xold = xnew`.

Esercizio 1.

Si ricordano di seguito le formule per l'implementazione dei tre metodi iterativi dato lo *splitting* della matrice A in $A = L + D + U$.

$$\begin{array}{rll} \text{Jacobi} & Mx^{(k+1)} = Nx^{(k)} + b & M = D,\ N = -(L+U) \\ \text{Gauss-Seidel} & Mx^{(k+1)} = Nx^{(k)} + b & M = L + D,\ N = -U \\ \text{SOR} & Mx^{(k+1)} = Nx^{(k)} + \omega b & M = \omega L + D,\ N = (1-\omega)D - \omega U \end{array}$$

Si implementino i metodi di Gauss-Seidel e SOR producendo altrettante function `seidel.m` e `sor.m`. Il metodo di Gauss-Seidel deve avere **un parametro aggiuntivo in output rispetto alla funzione** `jacobi.m` ovvero il raggio spettrale della sua matrice di iterazione $\rho(E_S)$. Il raggio spettrale della matrice d'iterazione si calcoli facendo uso del comando `eig`. La function che implementa SOR dovrà avere invece un parametro aggiuntivo di input ovvero il parametro di rilassamento, ω.

Si produca un **unico** script che, chiamando le tre function, risolva il sistema lineare $Ax = b$ dove

$$A = \begin{pmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{pmatrix}$$

e b si ottiene imponendo che la soluzione vera sia $x =$ `ones(5,1)`. Si utilizzino i metodi di Jacobi Gauss-Seidel e SOR con $\omega = \omega_{opt}$ ricavato a partire da $\rho(E_S)$. Si dica perché è possibile determinare tale valore di ω. Usare `tol` $= 10^{-10}$. Si riportino in un grafico semilogaritmico i profili di convergenza ottenuti con i tre metodi (iterazioni – norma euclidea dello scarto).

Calcolare per i tre metodi (Jacobi, Gauss-Seidel e SOR con ω_{opt}) la velocità di convergenza teorica e metterla in relazione con il numero di iterazioni impiegate dai tre metodi.

Soluzione di equazioni differenziali alle derivate parziali

In molti campi dell'ingegneria si pone il problema di determinare la temperatura in corpi solidi. In problemi stazionari e per solidi omogenei ed isotropi la temperatura T è governata dall'equazione di Laplace:

$$T''_{xx} + T''_{yy} = 0$$

Le derivate seconde possono approssimarsi con schemi alle differenze finite (si ricordi l'Esercitazione 1, esercizio 3). Su una griglia regolare come per esempio quella della Figura,
l'approssimazione alle differenze finite delle derivate seconde sul nodo i, j è :

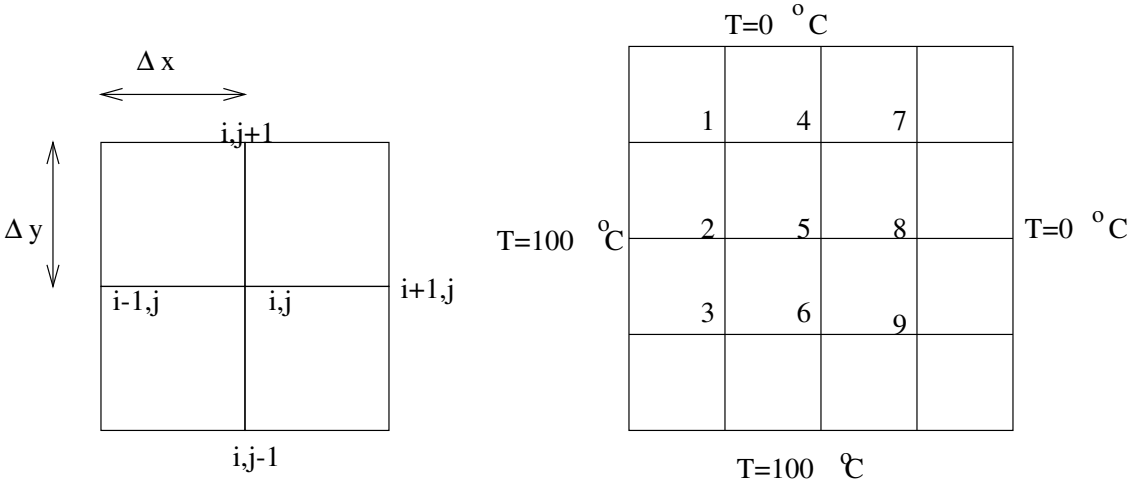
$$T''_{xx}(x_i, y_j) = \frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{h^2}$$

$$T''_{yy}(x_i, y_j) = \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{h^2}$$

con $h = \Delta x = \Delta y$. L'equazione di Laplace sul nodo i, j è dunque approssimata da:

$$T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1} - 4T_{i,j} = 0$$

Se i nodi del reticolo sono n, ne risulterà un sistema lineare di n equazioni nelle temperature nodali.



Esercizio 2.

Si consideri ancora la piastra riportata in Figura. I lati della piastra sono mantenuti alle temperature ivi indicate. Le temperature sui 9 nodi numerati si ottengono risolvendo il sistema:

$$\begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 4 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \\ T_7 \\ T_8 \\ T_9 \end{bmatrix} = \begin{bmatrix} 100 \\ 100 \\ 200 \\ 0 \\ 0 \\ 100 \\ 0 \\ 0 \\ 100 \end{bmatrix}$$

La matrice di tale sistema si può dimostrare essere biciclica e coerentemente ordinata. Ad essa si può applicare il teorema di Young-Varga. E' facile generare tale matrice in Matlab mediante il comando `A = delsq(numgrid('S',5))` (si veda l'esercizio 3. più avanti). In Octave il comando sarà `poisson(3)`, dove `poisson.m` è una function da scaricare alla pagina `dmsa.dispense.unipd.it` e cliccando sul nome del docente.

Costruire un programma Matlab/Octave che risolva il sistema assegnato con lo schema di sovra-rilassamento:

$$(\omega L + D)x^{(k+1)} = [(1-\omega)D - \omega U]x^{(k)} + \omega b$$

Si giustifichi perché è certamente possibile determinare un valore ottimo del parametro ω.

Partendo da $\omega = 1$ e finendo con $\omega = 1.5$ con passo $\Delta\omega = 0.025$ si risolva il sistema per i diversi valori di ω. Si fissi la tolleranza in uscita ε come segue:

$$\|x^{(k+1)} - x^{(k)}\| < 10^{-10}.$$

Si riporti in un grafico il numero di iterazioni al variare di ω e quindi si stimi il valore di ω_{opt}.

Si calcoli poi analiticamente ω_{opt} con la formula:

$$\omega_{opt} = \frac{2}{1 + \sqrt{1 - \rho(E_S)}},$$

dove $\rho(E_S)$ come nell'esercizio precedente è un parametro in uscita della function `seidel.m`. Si confrontino i due valori di ω_{opt} calcolati sperimentalmente ed analiticamente.

Esercizio 3.

Si consideri l'istruzione MatLAB:

```
nx = 22;
A = delsq(numgrid('S',nx));
```

Al variare del numero di nodi nx sull'asse x e y restituisce la matrice che discretizza l'equazione di Laplace ($nx = 22$ nell'esempio) su un quadrato unitario (parametro 'S' = square). Ponendo $nx = 5$ si ottiene come caso particolare la matrice dell'esercizio precedente. La matrice ha n righe e colonne con $n = (nx-2)^2$. Ogni matrice generata da questo comando è biciclica e coerentemente ordinata.
NB. In Octave la stessa matrice si genera con la sequenza di istruzioni:

```
nx = 22;
size = nx-2;
A = poisson(size);
```

Si prenda ancora $nx = 22$ e dunque $n = 400$. Si produca la matrice A ed il termine noto b è costruito in modo che la soluzione vera sia il vettore di tutti "uni": $x =$ `ones(n,1)`. Si usino per i sotto definiti metodi iterativi `tol` $= 10^{-8}$ e `itmax` $= 5000$.

1. Si risolva il sistema $Ax = b$ con il metodo di Gauss-Seidel (usare una variante della precedente function **senza il calcolo di** $\rho(E_S)$). Verificare che la soluzione finale sia corretta calcolando la norma dell'errore.

2. Si risolva ora lo stesso problema con il metodo SOR con $\omega = 1.5$. Verificare che la soluzione finale sia corretta calcolando la norma dell'errore.

3. Risolvere ancora il sistema con il metodo SOR ma stimando il valore ottimale di ω. Per le matrici che discretizzano l'equazione di Laplace, è teoricamente noto il valore del raggio spettrale della matrice di iterazione di Gauss-Seidel che si può approssimare come

$$\rho(E_S) \approx \left(1 - \frac{\pi^2 h^2}{2}\right)^2, \qquad \text{dove} \qquad h = \frac{1}{nx-2}.$$

   Si calcoli dunque l'ω_{opt} mediante la formula valida per matrici bicicliche e coerentemente ordinate. Dopo aver risolto il sistema con SOR e ω_{opt}, verificare che la soluzione finale sia corretta calcolando la norma dell'errore.

Si produca infine un profilo semilogaritmico della norma euclidea dello scarto in funzione del numero di iterazioni per i tre metodi: Seidel, SOR(1.5) e SOR(ω_{opt}).

Esercizio 4. Facoltativo

Si confronti il tempo di calcolo di un sistema $Ax = b$, soluzione vera $x =$ `ones(n,1)`, del metodo SOR con ω_{opt} e la fattorizzazione LU della matrice A. Si costruisca la matrice A con il comando

```
A = delsq(numgrid('S',nx))
```

ponendo $nx = [22\ 32\ 42\ 52\ 102\ 152\ 202]$; così da ottenere matrici di dimensioni rispettivamente $400, 900, \ldots, 40000$. Per il calcolo dei tempi si usino le istruzioni `tic` e `toc` come nel seguente esempio:

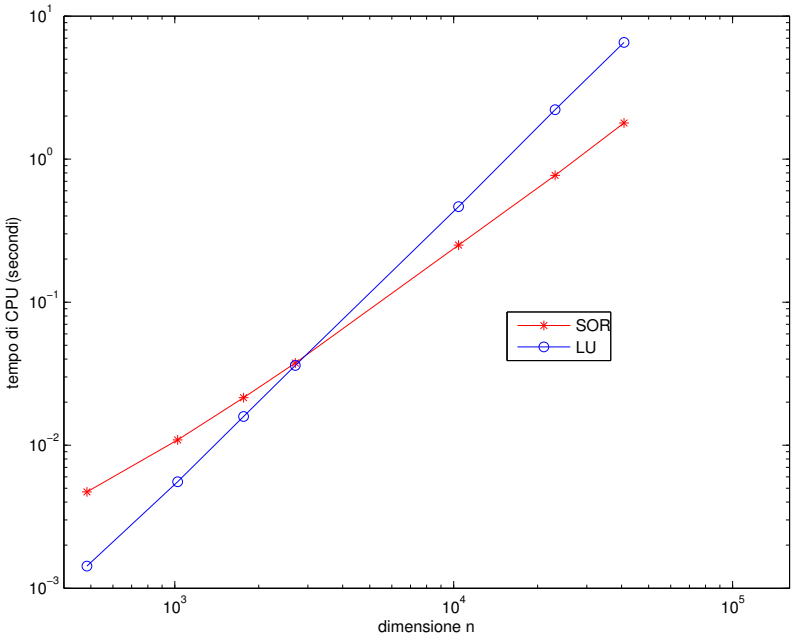
```
>> tic;
>> nx = 20;
>> A =delsq(numgrid('S',nx));
>> [L,U,P]=lu(A);
>> tempo = toc

tempo =
    0.017365000000000
```

ovvero si dà il comando `tic` prima della prima istruzione e il comando `toc` subito dopo l'ultima istruzione. La variabile a cui è assegnato `toc` darà il tempo (in secondi) trascorso per eseguire le istruzioni comprese tra i due comandi.

1. Si risolvano i 7 sistemi con i due metodi: (SOR con ω_{opt} calcolato come descritto nell'esericzio 3) e la fattorizzazione LU di MatLAB e successiva soluzione del sistema con la risoluzione dei due sistemi lineari triangolari. Per ciascuna matrice e metodo si verifichi l'accuratezza della soluzione mediante la norma euclidea dell'errore.

2. Per ogni metodo si produca un vettore dei tempi di calcolo ottenuti con i comandi tic e toc. Si produca infine un grafico logaritmico `loglog` con le due curve (una per metodo) in cui si grafica il tempo di calcolo in funzione della dimensione del sistema n.

   Il grafico avrà un andamento simile a quello mostrato in figura:



3. Per ciascuno dei due metodi si può stimare la **complessità** ovvero il numero di operazioni da eseguire (e quindi il tempo di CPU impiegato) in funzione della dimensione della matrice n. Possiamo supporre che:

   - Tempo CPU del metodo LU = $c_1 n^{p_1}$
   - Tempo CPU del metodo SOR = $c_2 n^{p_2}$

   Si stimino c_i e p_i approssimando gli ultimi 5 dati sperimentali (n, T_n) con un modello potenza.

NOTA BENE: Si deve consegnare entro il 15 Giugno 2013 un unico file (archivio) contenente: una relazione scritta (possibilmente in formato pdf) con la soluzione di questi esercizi, i grafici/output ottenuti e gli m-files Matlab/Octave **utilizzati.**