Decomposizioni ai Valori Singolari

Esercitazione facoltativa in matlab

Relazione breve sullo strumento di decomposizione a valori singolari di matlab e delle sue applicazioni

Decomposizioni ai Valori Singolari

Esercitazione facoltativa in matlab

Non sono bello ma patcho

La relazione e il codice dell'esercitazione sono state portate a termine da:

Andrea Storace:

4186140

Andrea Straforini:

4338710

Elisa Zazzera:

4380663

Esercizio 1

Il primo esercizio chiedeva, dopo aver inizializzato una matrice di dimensioni *3×m* con *m=73* e definita come segue:

$$A = \begin{pmatrix} 1 & x_1 & x_1^2 \\ \vdots & \vdots & \vdots \\ 1 & x_m & x_m^2 \end{pmatrix} \; dove\; x_i = {}^{i}/_{m}, i \in \underline{m} - \{0\}$$

di calcolarne la decomposizione ai valori singolari tramite il comando `svd(A)`, quindi quelli della sua trasposta.

I valori singolari corrispondono in entrambe le matrici, come si può vedere in **figura 1**.

svd(A)			svd(At)		
4,551663	0	0	4,551663	0	0
0	1,323168	0	0	1,323168	0
0	0	0,184281	0	0	0,184281

Figura 1

Quindi era richiesto di confrontare gli autovalori delle matrici AA^t e A^tA con la decomposizione ai valori singolari di A. Prima di discutere della decomposizione, vorrei soffermarmi sugli autovalori delle matrici AA^t e A^tA: gli autovalori significativi di AA^t coincidono con quelli di A^tA, questo porta a pensare che le decomposizioni di A e A^t saranno perfettamente identiche: come descritto a pagina **17** del Capitolo **6** delle dispense, i valori singolari sono calcolati come radici degli autovalori della matrice data per la sua trasposta.

> **Algoritmo 6.1** Calcolo dei valori singolari e dei vettori singolari
> Calcolo λ_j di A^TA
> $\sigma_j := \sqrt{\lambda_j}$
> Calcolo autovettori di A^TA (vettori singolari destri)
> Calcolo λ_j di AA^T
> $\sigma_j := \sqrt{\lambda_j}$
> Calcolo autovettori di AA^T (vettori singolari sinistri)

Quindi i **valori singolari di A e A^T** saranno, prima di tutto, identici ed esattamente le **radici** degli **autovalori** delle matrici **AA^t e A^tA**, come riportato in **figura 2**. Guardando la pagina di manuale di `orth()` si può notare che l'immagine ortonormale della matrice passata come argomento viene presa dalle prime *r* colonne dei valori sinistri della decomposizione

svdA	eigAAt*	eigAtA*
4,551663	0,03396	0,03396
1,323168	1,750774	1,750774
[illegible]	[illegible]	[illegible]

Figura 2

ai valori singolari di A, dove *r* è il rango della matrice; ci si aspetterà quindi che le prime 3 colonne di U (ovvero le colonne associate ai valori singolari non nulli di A) coincidano con l'immagine di A, come si può vedere in **figura 3**, stesso discorso per A^T e i suoi valori sinistri come riportato in figura 4.

Immagine di A			Colonne 1:3 di U		
-0,18474	0,405719	0,539026	-0,18474	0,405719	0,539026
-0,19344	0,372071	0,309595	-0,19344	0,372071	0,309595
-0,20292	0,333319	0,116866	-0,20292	0,333319	0,116866
-0,21318	0,289463	-0,03916	-0,21318	0,289463	-0,03916
-0,22423	0,240503	-0,15849	-0,22423	0,240503	-0,15849
-0,23607	0,18644	-0,24112	-0,23607	0,18644	-0,24112
-0,24869	0,127273	-0,28704	-0,24869	0,127273	-0,28704
-0,2621	0,063002	-0,29627	-0,2621	0,063002	-0,29627
-0,27629	-0,00637	-0,26879	-0,27629	-0,00637	-0,26879
-0,29127	-0,08085	-0,20461	-0,29127	-0,08085	-0,20461
-0,30703	-0,16043	-0,10374	-0,30703	-0,16043	-0,10374
-0,32358	-0,24512	0,033844	-0,32358	-0,24512	0,033844
-0,34091	-0,33491	0,208124	-0,34091	-0,33491	0,208124

Figura 3

Immagine di At			Ut		
-0,80487	0,574604	0,148375	-0,80487	0,574604	0,148375
-0,47905	-0,4815	-0,73395	-0,47905	-0,4815	-0,73395
-0,35029	-0,66181	0,662804	-0,35029	-0,66181	0,662804

Figura 2

In fine, dal momento che il nucleo di una matrice può essere calcolato come le colonne associate ai valori singolari nulli, ci si aspetta un ulteriore corrispondenza tra le colonne di ker(A) e quelle di V e rispettivamente ker(A^T) e V^T. Come si può vedere di seguito però il nucleo di A è il vettore nullo, poiché nessun valore singolare di A risulta nullo (**figura 1**). I valori di ker(A^T) e V^T non tornano, calcolando infatti il nucleo di A^T come colonne di V^T associate ai valori singolari nulli, il risultato restituito è nullo, come per il nucleo di A, al contrario, se calcolato con la funzione `null()`, questa restituisce le colonne di V^T ottenute rimuovendo le colonne associate ai valori singolari non nulli (dalla quarta alla quattordicesima colonna).

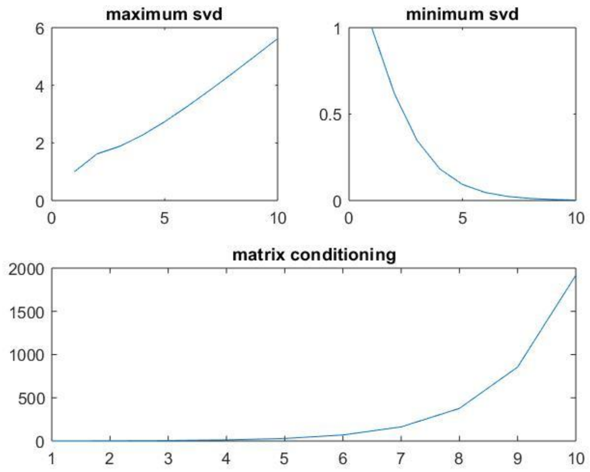
Esercizio 2

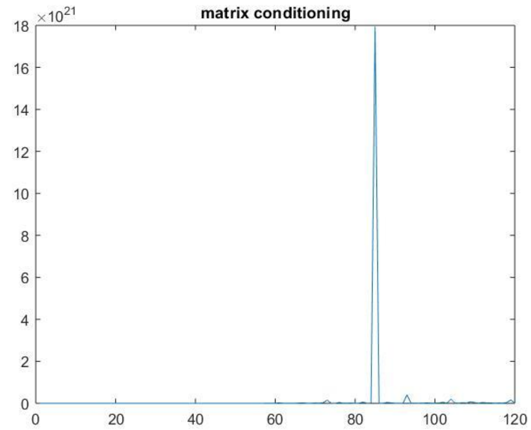
Il secondo esercizio richiedeva di studiare l'andamento del valore singolare massimo e minimo di una matrice definita:

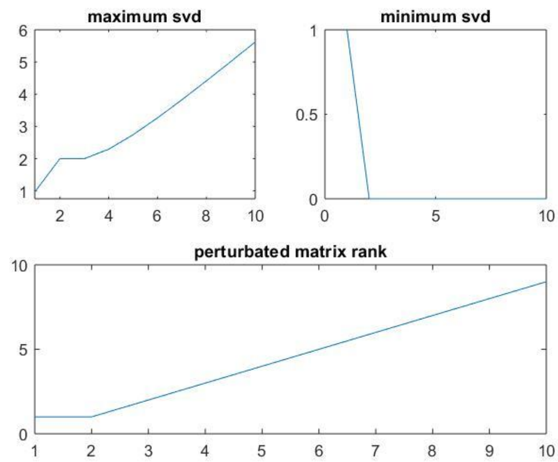
$$b_{i,j} = \begin{cases} 1 & se\ i = j \\ -1 & se\ i < j \\ 0 & se\ i > j \end{cases}$$

Quindi di fare un confronto sugli autovalori perturbando la matrice ottenuta.

Dal primo confronto sui valori singolari **massimi** e **minimi** della decomposizione, si può vedere come **la crescita dei primi sia inversamente proporzionale a quella dei secondi**. L'andamento della curva descritta dai valori di massima della decomposizione ha un andamento lineare per matrici di ordine minore uguale a 2 mentre, per matrici di ordine superiore, la crescita del valore dominante cresce più lentamente. Discorso differente per il valore minimo che decresce uniformemente. L'unica variabile con un andamento interessante è il **condizionamento della matrice** per ordini superiori a 58: strettamente crescente fino a tale ordine, il suo andamento cambia repentinamente con un picco massimo per matrici di ordine 85. L'andamento degli autovalori della matrice perturbata, mostrano lo stesso andamento: crescente per il massimo ma decrescente per il minimo; l'autovalore massimo mantiene sempre il comportamento lineare per ordini inferiori a 2 ma diviene costante per gli ordini 2 e 3, per poi riprendere lo stesso andamento delle matrici non perturbate. Al contrario l'autovalore minimo si abbatte per valori maggiori uguali a 2 al valore 0. Infine, il rango della matrice A è costante per matrici di ordine minore uguale a 2, per poi crescere linearmente.







Esercizio 3

Il terzo esercizio chiedeva di determinare la soluzione ai minimi quadrati del sistema:

$$Ac = y \qquad dove\ A = \begin{pmatrix} 1 & x_1 & x_1^2 \\ \vdots & \vdots & \vdots \\ 1 & x_m & x_m^2 \end{pmatrix}, y = \begin{pmatrix} \sin x_1 \\ \vdots \\ \sin x_m \end{pmatrix}$$

Con x_m definito come nell'esercizio 1.

Ogni metodo di risoluzione restituisce lo stesso vettore di soluzioni:

```
c0 =                 c1 =                 c2 =                 c3 =

   -0.0061              -0.0061              -0.0061              -0.0061
    1.0796               1.0796               1.0796               1.0796
   -0.2314              -0.2314              -0.2314              -0.2314
```

Il primo metodo utilizza la decomposizione ai valori singolari della matrice A, ed implementa l'algoritmo descritto a pagina 21 delle dispense (6.5 Pseudoinversa):

```
function b = bySVD(A, v)
    b=0;
    [U, S, V] = svd(A);
    dS=diag(S);
    for i=1:rank(A)
        b=b+((U(:,i)'*v)/dS(i))*V(:,i);
    end
end
```

$$\boxed{x = \sum_{i=1}^{r} \frac{u_i^T b}{\sigma_i} v_i}$$

Equazione 1

Il secondo metodo utilizza la decomposizione QR:

```
function [b, r] = byQR(A, v)
    [row, col]=size(A);
    [Q,R]=qr(A);
    h=Q' * v;
    h1 = h(1:col)';
    h2 = h(col:row)';
    b=R\h;
    r=norm(h2);
end
```

Il terzo metodo prevede l'uso delle equazioni normali $A^TAc=A^Ty$:

```
function b = byNormEq(A, y)
    b= (A'*A)\(A'*y);
end
```

Il quarto metodo utilizza il comando built-in di matlab:

```
A\y
```