Lezioni di Calcolo Numerico
Lezione 13: Autovalori e valori singolari di matrici

Alberto Tibaldi

11 giugno 2018

Indice

3.3 Applicazioni della decomposizione ai valori singolari

Dopo aver introdotto la decomposizione ai valori singolari, in modo spero non troppo traumatico, è finalmente ora di vedere in che contesti essa possa essere applicata.

3.3.1 Approssimazione di matrici

La prima, interessantissima applicazione della decomposizione ai valori singolari è la **approssimazione di matrici**. Infatti, sfruttando il fatto che, data una matrice di rango r, i primi[1] r vettori singolari sinistri $\{\underline{u}_i\}$ e destri $\{\underline{v}_i\}$ costituiscono rispettivamente una base dell'immagine e dello spazio delle righe, è possibile costruire una matrice che in qualche senso *approssimi* quella di partenza. Sostanzialmente, invece di prendere le basi intere, ne prendiamo solo alcuni elementi. Per portare a termine questo compito possiamo appoggiarci al seguente teorema: si considerino i valori singolari **ordinati decrescentemente**:

$$s_1 \geq s_2 \geq ... \geq s_r \geq s_{r+1} \geq ...$$

e consideriamo al più i primi r: **quelli diversi da zero**. Se definiamo la matrice $\underline{\underline{A}}_k$ come

$$\underline{\underline{A}}_k = \sum_{i=1}^{k} s_i \underline{u}_i \underline{v}_i^{\mathrm{T}} \tag{1}$$

dove k è un numero intero minore del rangos, allora, dato

$$\mathcal{B}_k = \left\{ \underline{\underline{B}} \ : \ \text{rango di } \underline{\underline{B}} \text{ uguale a } k \right\}$$

che, a parole, è l'insieme di tutte le possibili matrici $\underline{\underline{B}}$ aventi rango pari a k, si ha che $\underline{\underline{A}}_k$ è la matrice tale per cui

$$\min_{\underline{\underline{B}} \in \mathcal{B}_k} \left\| \underline{\underline{A}} - \underline{\underline{B}} \right\|_2 = \left\| \underline{\underline{A}} - \underline{\underline{A}}_k \right\|_2 = s_{k+1}.$$

[1] ovvero associati ai valori singolari più grandi

Traduciamo dal matematichese: se definiamo la *distanza di due matrici* come la norma della loro differenza[2], la matrice $\underline{\underline{A}}_k$ è, di tutte le matrici $\underline{\underline{B}} \in \mathcal{B}_k$, quindi aventi rango k, quella meno distante da $\underline{\underline{A}}$, ossia quella che la approssima meglio. Inoltre, è possibile stimare la *distanza* tra $\underline{\underline{A}}$ e $\underline{\underline{A}}_k$, come il valore singolare $(k+1)$-esimo della matrice $\underline{\underline{A}}$. Di conseguenza, la $\underline{\underline{A}}_k$ è la miglior approssimazione in norma 2 di $\underline{\underline{A}}$ con una matrice di rango k.

Operativamente, implementare la definizione (1) è semplicissimo: è sufficiente *ritagliare* k colonne dalle matrici $\underline{\underline{U}}$, $\underline{\underline{S}}$, $\underline{\underline{V}}$, e quindi moltiplicare queste matrici; questo si può per esempio effettuare con il seguente codice MATLAB®:

```
[U,S,V] = svd(A); % calcolo decomposizione valori singolari
Uk = U(:,1:k); % ritaglio k colonne matrice dei vettori singolari sinistri
Vk = V(:,1:k); % ritaglio k colonne matrice dei vettori singolari destri
Sk = S(1:k,1:k); % ritaglio il blocco k x k della matrice dei valori singolari
Ak = Uk*Sk*Vk'; % ri-moltiplico le matrici
distanza = S(k+1,k+1) % calcolo distanza come valore singolare (k+1)-esimo
```

Questa idea è alla base di strepitose applicazioni, tra cui per esempio la compressione di immagini, come mostrato nel programmino presentato in Appendice A.

3.3.2 Calcolo del rango di una matrice

Il rango di una matrice $\underline{\underline{A}}$ coincide con il numero dei suoi valori singolari non nulli. Questo dovrebbe essere chiaro dalle precedenti sezioni, in cui abbiamo *costruito* la decomposizione ai valori singolari in modo tale che $s_i = 0$ per $i > r$. Tuttavia, potrebbe sorgere la seguente domanda:

> «Ma cosa vuol dire *nulli*? Cioè, proprio zero zero, o qualcosa di diverso?»

In effetti, la cosa più sana è definire il cosiddetto *rango numerico*: una volta fissata una certa tolleranza `toll`, è possibile definire il rango numerico come il numero di valori singolari più grandi di essa. Per esempio, se per qualche motivo fissassimo una tolleranza pari a 10^{-10}, avremmo

$$\{s_i\} = \{\underbrace{1, 0.5, 0.1, 10^{-6}}_{>10^{-10}}, \underbrace{10^{-11}, 0, 0}_{<10^{-10}}\},$$

ossia 4 valori singolari maggiori di 10^{-10}. Quindi, il rango numerico della matrice sarebbe 4. Si può calcolare il rango di una matrice `A` mediante il comando

```
r = rank(A);
```

o, volendo il rango numerico, data tolleranza `toll`,

```
r = rank(A,toll);
```

3.3.3 Calcolo del condizionamento spettrale di una matrice

Data $\underline{\underline{A}}$ una matrice di ordine n e rango n, quindi massimo, ricordiamo che il numero di condizionamento spettrale, ovvero il numero di condizionamento in norma 2, è

$$\kappa_2(\underline{\underline{A}}) = \left\|\underline{\underline{A}}\right\|_2 \left\|\underline{\underline{A}}^{-1}\right\|_2.$$

Ragionando sul primo termine, abbiamo visto che la norma spettrale della matrice è imparentata al raggio spettrale:

$$\left\|\underline{\underline{A}}\right\|_2 = \sqrt{\rho(\underline{\underline{A}}^{\mathrm{T}}\underline{\underline{A}})}.$$

Possiamo a questo punto sostituire in questa espressione la decomposizione ai valori singolari della matrice $\underline{\underline{A}}$, ovvero

[2] in questo caso ci interessiamo della norma spettrale

$$\underline{\underline{A}} = \underline{\underline{U}}\,\underline{\underline{S}}\,\underline{\underline{V}}^{\mathrm{T}},$$

ottenendo

$$\left\|\underline{\underline{A}}\right\|_2 = \sqrt{\rho((\underline{\underline{U}}\,\underline{\underline{S}}\,\underline{\underline{V}}^{\mathrm{T}})^{\mathrm{T}}(\underline{\underline{U}}\,\underline{\underline{S}}\,\underline{\underline{V}}^{\mathrm{T}}))} = \sqrt{\rho(\underline{\underline{V}}\,\underline{\underline{S}}^{\mathrm{T}}\underline{\underline{U}}^{\mathrm{T}}\underline{\underline{U}}\,\underline{\underline{S}}\,\underline{\underline{V}}^{\mathrm{T}})} = \sqrt{\rho(\underline{\underline{V}}\,\underline{\underline{S}}^2\underline{\underline{V}}^{\mathrm{T}})}.$$

Tuttavia, $\rho(\underline{\underline{V}}\,\underline{\underline{S}}^2\underline{\underline{V}}^{\mathrm{T}})$ è semplicemente il massimo autovalore della matrice $\underline{\underline{V}}\,\underline{\underline{S}}^2\underline{\underline{V}}^{\mathrm{T}}$ che, quindi, interpretando questo prodotto come una decomposizione agli autovalori, è

$$\left\|\underline{\underline{A}}\right\|_2 = \sqrt{\rho(\underline{\underline{S}}^2)} = \rho(\underline{\underline{S}}) = s_1,$$

ossia il tutto si riduce a essere s_1: il più grande dei valori singolari!

Con una dimostrazione molto simile si può arrivare a dimostrare che

$$\left\|\underline{\underline{A}}^{-1}\right\|_2 = \frac{1}{s_n},$$

dove s_n è il n-esimo valore singolare, dato n il numero di righe e colonne della matrice in questione. Quindi, riassumendo,

$$\kappa_2(\underline{\underline{A}}) = \frac{s_1}{s_n}.$$

3.3.4 Risoluzione di un sistema lineare determinato

Come quasi tutte le fattorizzazioni che abbiamo studiato finora, anche la SVD permette di risolvere un sistema lineare determinato nella forma

$$\underline{\underline{A}}\,\underline{x} = \underline{b},$$

dove dunque $\underline{\underline{A}}$ ha dimensione $n \times n$ e non è singolare. Applicando la decomposizione ai valori singolari, l'espressione appena riportata diventa

$$\underline{\underline{U}}\,\underline{\underline{S}}\,\underline{\underline{V}}^{\mathrm{T}}\underline{x} = \underline{b}.$$

Ricordando che $\underline{\underline{U}}$ è ortogonale, si ha che $\underline{\underline{U}}^{-1} = \underline{\underline{U}}^{\mathrm{T}}$, e quindi il sistema diventa

$$\underline{\underline{S}}\,\underline{\underline{V}}^{\mathrm{T}}\underline{x} = \underline{\underline{U}}^{\mathrm{T}}\underline{b}.$$

Definiamo a questo punto un vettore $\underline{y}$ come

$$\underline{y} = \underline{\underline{V}}^{\mathrm{T}}\underline{x},$$

che ci permette di riscrivere il sistema come

$$\underline{\underline{S}}\,\underline{y} = \underline{\underline{U}}^{\mathrm{T}}\underline{b}.$$

Notare che il sistema appena scritto è **diagonale**, e quindi la sua soluzione è assolutamente elementare! Risolto quindi il sistema, e trovato $\underline{y}$ dividendo ciascuna componente del vettore $\underline{\underline{U}}^{\mathrm{T}}\underline{b}$ per il corrispettivo elemento diagonale di $\underline{\underline{S}}$, si deve risolvere il sistema

$$\underline{\underline{V}}^{\mathrm{T}}\underline{x} = \underline{y}$$

dove, sfruttando l'ortogonalità di $\underline{\underline{V}}$, è sufficiente portarla a secondo membro trasponendola: $\underline{\underline{V}}^{\mathrm{T}} = \underline{\underline{V}}^{-1}$:

$$\underline{x} = \underline{\underline{V}}\,\underline{y}.$$

Questo metodo quindi non richiede la soluzione di alcun sistema triangolare, e da questo punto di vista sembrerebbe conveniente. Tuttavia, il costo necessario per produrre la SVD è talmente alto che rende tutto questo assolutamente sconsigliabile: si arriva a $32n^3/3$ operazioni richieste!

Per completare il discorso, bisognerebbe però spezzare una lancia a favore dell'uso della SVD per la soluzione di alcuni sistemi lineari determinati, perché esiste una situazione in cui è decisamente consigliata: quando il sistema è mal condizionato e prossimo a essere singolare, la SVD è nettamente più efficace al fine di risolverlo e, quindi, può valere la pena di pagare con l'elevatissimo numero di operazioni una maggiore stabilità numerica. Infatti, è possibile usare la SVD per identificare i valori singolari più piccoli di una prefissata tolleranza, approssimare la matrice $\underline{\underline{A}}$ con una di rango inferiore e, infine, risolvere il risultante problema nel senso dei minimi quadrati seguendo le indicazioni che verranno spiegate nella prossima sezione.

3.3.5 Risoluzione di un sistema lineare sovradeterminato

Non è la prima volta che trattiamo sistemi lineari sovradeterminati, ovvero aventi più equazioni che incognite. Tuttavia, quando lo abbiamo fatto usando la fattorizzazione QR, avevamo risolto il problema per matrici $\underline{\underline{A}}$ aventi rango massimo.

Senza voler ripassare troppo, vorrei ricordare che la soluzione $\underline{x}^*$ di un sistema lineare sovradimensionato non va intesa *in senso classico*, *esatto*, ma nel senso dei minimi quadrati, ovvero

$$\left\| \underline{\underline{A}}\,\underline{x}^* - \underline{b} \right\|_2 = \min_{\underline{y} \in \mathbb{R}^n} \left\| \underline{\underline{A}}\,\underline{y} - \underline{b} \right\|_2 ,$$

dove $\underline{x}^*$ è la soluzione che otteniamo dopo aver provato tutti i $\underline{y}$ possibili e avendo trovato quello che minimizza la norma del residuo. Similmente a quanto fatto in precedenza, cerchiamo di riscrivere la norma in modo conveniente. Ricordiamoci in particolare la proprietà

$$\|\underline{x}\|_2^2 = \left\| \underline{\underline{U}}^{\mathrm{T}} \underline{x} \right\|_2^2 ,$$

ovvero il fatto che, in una norma 2, è possibile far apparire o sparire liberamente una matrice ortogonale $\underline{\underline{U}}^{\mathrm{T}}$. Allora, facciamolo:

$$\left\| \underline{\underline{A}}\,\underline{y} - \underline{b} \right\|_2^2 = \left\| \underline{\underline{U}}^{\mathrm{T}} (\underline{\underline{A}}\,\underline{y} - \underline{b}) \right\|_2^2 = \left\| \underline{\underline{U}}^{\mathrm{T}} \underline{\underline{A}}\,\underline{y} - \underline{\underline{U}}^{\mathrm{T}} \underline{b} \right\|_2^2 ,$$

dove è ovvio che questa $\underline{\underline{U}}^{\mathrm{T}}$ è proprio la trasposta della matrice $\underline{\underline{U}}$ della fattorizzazione SVD di $\underline{\underline{A}}$. A questo punto usiamo l'altro dei nostri trucchi preferiti, ovvero introdurre dove ci fa comodo una matrice identità $\underline{\underline{I}}$, definita come $\underline{\underline{I}} = \underline{\underline{V}}\,\underline{\underline{V}}^{\mathrm{T}}$; questo permette all'espressione di diventare

$$\left\| \underline{\underline{U}}^{\mathrm{T}} \underline{\underline{A}}\,\underline{y} - \underline{\underline{U}}^{\mathrm{T}} \underline{b} \right\|_2^2 = \left\| \underline{\underline{U}}^{\mathrm{T}} \underline{\underline{A}}\,\underline{\underline{V}}\,\underline{\underline{V}}^{\mathrm{T}}\,\underline{y} - \underline{\underline{U}}^{\mathrm{T}} \underline{b} \right\|_2^2 ,$$

dove guarda caso anche $\underline{\underline{V}}$ è proprio quella della SVD ;-). Dalla ormai ben nota

$$\underline{\underline{A}} = \underline{\underline{U}}\,\underline{\underline{S}}\,\underline{\underline{V}}^{\mathrm{T}},$$

è possibile moltiplicare da destra per $\underline{\underline{V}}$, da sinistra per $\underline{\underline{U}}^{\mathrm{T}}$, sfruttare l'ortogonalità e ottenere

$$\underline{\underline{S}} = \underline{\underline{U}}^{\mathrm{T}} \underline{\underline{A}}\,\underline{\underline{V}},$$

che, guarda di nuovo caso, è proprio quello che è apparso nella norma! Quindi, possiamo riscrivere l'ultimo termine come

$$\left\| \underline{\underline{U}}^{\mathrm{T}} \underline{\underline{A}}\,\underline{\underline{V}}\,\underline{\underline{V}}^{\mathrm{T}}\,\underline{y} - \underline{\underline{U}}^{\mathrm{T}} \underline{b} \right\|_2^2 = \left\| \underline{\underline{S}}\,\underline{\underline{V}}^{\mathrm{T}}\,\underline{y} - \underline{\underline{U}}^{\mathrm{T}} \underline{b} \right\|_2^2 .$$

Se a questo punto definiamo un vettore

$$\underline{z} = \underline{\underline{V}}^{\mathrm{T}} \underline{y},$$

e

$$\underline{c} = \underline{\underline{U}}^{\mathrm{T}} \underline{b}, \tag{2}$$

arriviamo a ottenere

$$\left\|\underline{\underline{A}}\,\underline{y} - \underline{b}\right\|_2^2 = \left\|\underline{\underline{S}}\,\underline{z} - \underline{c}\right\|_2^2,$$

un po' come eravamo riusciti a fare con la fattorizzazione QR.

Continuando a imitare le mosse fatte allora, definiamo a partire dal decomporre i vettori $\underline{z}$ e $\underline{c}$ in due parti:

$$\underline{z} = \begin{bmatrix} \underline{z}_1 \\ \underline{z}_2 \end{bmatrix}, \qquad \underline{c} = \begin{bmatrix} \underline{c}_1 \\ \underline{c}_2 \end{bmatrix},$$

dove $\underline{c}_1 \in \mathbb{R}^r$ è il vettore contenente le prime r componenti, dato r il rango della matrice $\underline{\underline{A}}$, del vettore $\underline{c}$, mentre $\underline{c}_2 \in \mathbb{R}^{n-r}$ contiene le restanti, ultime, $n-r$ componenti. Lo stesso discorso si può fare per il vettore $\underline{z}$: $\underline{z}_1 \in \mathbb{R}^r$ è il vettore contenente le prime r componenti del vettore $\underline{z}$, mentre $\underline{z}_2 \in \mathbb{R}^{n-r}$ contiene le rimanenti $n-r$. Infine, se definiamo la matrice $\widetilde{\underline{\underline{S}}} \in \mathbb{R}^{r,r}$ come il *ritaglio* delle prime r righe e r colonne della matrice $\underline{\underline{S}}$, possiamo arrivare a scrivere

$$\left\|\underline{\underline{A}}\,\underline{y} - \underline{b}\right\|_2^2 = \left\|\underline{\underline{S}}\,\underline{z} - \underline{c}\right\|_2^2 = \left\|\begin{bmatrix} \widetilde{\underline{\underline{S}}}\,\underline{z}_1 - \underline{c}_1 \\ \underline{c}_2 \end{bmatrix}\right\|_2^2,$$

che ha senso poiché la matrice $\underline{\underline{S}}$ in totale ha una forma del tipo

$$\underline{\underline{S}} = \begin{bmatrix} \widetilde{\underline{\underline{S}}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} \end{bmatrix},$$

dove tutti gli elementi all'infuori di quelli in $\widetilde{\underline{\underline{S}}}$ sono nulli, dal momento che i valori singolari in posizioni superiori a quella del rango sono nulli per costruzione della SVD.

Proprio come con la fattorizzazione QR, siamo riusciti a decomporre il problema in due sottoproblemi: le prime r equazioni sono raggiungibili dai gradi di libertà a nostra disposizione, e quindi questo ci permette di risolvere il problema dei minimi quadrati mediante la soluzione del sistema lineare

$$\widetilde{\underline{\underline{S}}}\,\underline{z}_1 = \underline{c}_1,$$

ma $\underline{c}_2$ non è *raggiungibile* dai nostri gradi di libertà e, quindi, proprio come prima, non possiamo farci nulla. Il sistema lineare appena scritto è ancora più semplice del solito: la matrice $\widetilde{\underline{\underline{S}}}$ infatti è diagonale e, quindi, è possibile scrivere la soluzione come

$$z_i = \frac{c_i}{s_i}, \qquad i = 1, 2, \ldots, r.$$

dove z_i e c_i sono le i-esime componenti dei vettori $\underline{z}$ e $\underline{c}_1$, rispettivamente, e s_i è l'i-esimo valore singolare della matrice $\underline{\underline{A}}$.

Si noti che però quanto appena scritto vale solo per le prime r componenti del vettore $\underline{z}$ ma non ci dice nulla sulle restanti $n-r$ componenti, che sono state di fatto escluse da tutti questi conti in quanto ininfluenti. Poiché tuttavia ciò che ci piacerebbe è non avere una soluzione a caso, ma **la soluzione di minima norma**, dal momento che qualsiasi cosa noi mettiamo per le ultime $n-r$ componenti del vettore $\underline{z}$, queste non cambieranno in alcun modo la norma che stiamo cercando di minimizzare[3], possiamo definire la soluzione di minima norma $\underline{z}^*$ come avente le componenti

$$z_i^* = \begin{cases} \dfrac{c_i}{s_i}, & i = 1, \ldots, r \\ 0, & i = r+1, \ldots, n. \end{cases} \tag{3}$$

Dalla $\underline{z}^*$ di minima norma è possibile trovare la $\underline{x}^*$ di minima norma; infatti, dal momento che

$$\underline{x}^* = \underline{\underline{V}}\,\underline{z}^*, \tag{4}$$

essendo $\underline{\underline{V}}$ una matrice ortogonale, abbiamo che

[3] dal momento che corrispondono a elementi nulli della matrice $\underline{\underline{S}}$ in quanto in posizioni maggiori del rango della matrice

$$\|\underline{x}^*\|_2^2 = \left\|\underline{\underline{V}}\,\underline{z}^*\right\|_2^2 = \|\underline{z}^*\|_2^2,$$

e quindi la norma è preservata: essendo certamente $\underline{z}^*$ la soluzione di minima norma, ed essendo $\underline{z}^*$ e $\underline{x}^*$ legate da una matrice ortonormale, se $\underline{z}^*$ è quella avente minima norma, allora lo sarà anche $\underline{x}^*$.

3.3.6 Matrice pseudoinversa di Moore-Penrose

Se in qualche modo siamo riusciti a estendere il concetto di autovalore, e di conseguenza quello di determinante[4] a matrici rettangolari, non abbiamo ancora avuto modo di estendere il concetto di matrice inversa. Al fine di andare in questa direzione, ricordiamoci che, a partire da un sistema lineare

$$\underline{\underline{A}}\,\underline{x} = \underline{b},$$

in linea di principio[5], è possibile ottenere la soluzione $\underline{x}$ moltiplicando ambo i membri per la matrice inversa di $\underline{\underline{A}}$:

$$\underline{x} = \underline{\underline{A}}^{-1}\underline{b}. \tag{5}$$

Questa visione ci permetterà, per analogia, di estendere il concetto di matrice inversa. A partire dalla (3), possiamo definire una matrice diagonale $\underline{\underline{S}}^+$ avente componenti

$$(\underline{\underline{S}}^+)_{ii} = \begin{cases} \dfrac{1}{s_i}, & i = 1, \dots, r \\ 0, & i = r+1, \dots, n. \end{cases}$$

Questa definizione ci permette di scrivere compattamente l'espressione (3) come segue:

$$\underline{z}^* = \underline{\underline{S}}^+\underline{c}.$$

Questa espressione si può elaborare sostituendo (4) a membro sinistro e (2) a membro destro, arrivando a ottenere

$$\underline{\underline{V}}^{\mathrm{T}}\underline{x}^* = \underline{\underline{S}}^+\underline{\underline{U}}^{\mathrm{T}}\underline{b},$$

e, sfruttando l'ortogonalità della matrice $\underline{\underline{V}}$, portarla a membro destro è molto semplice, permettendoci di scrivere

$$\underline{x}^* = \underline{\underline{V}}\,\underline{\underline{S}}^+\underline{\underline{U}}^{\mathrm{T}}\underline{b}. \tag{6}$$

È evidente che questa espressione ricordi molto (5): si tratta di una relazione diretta tra il termine noto $\underline{b}$ e la soluzione $\underline{x}^*$; per questo motivo la matrice $\underline{\underline{A}}^+$, definita come

$$\underline{\underline{A}}^+ = \underline{\underline{V}}\,\underline{\underline{S}}^+\underline{\underline{U}}^{\mathrm{T}} \tag{7}$$

viene detta **matrice pseudoinversa di Moore-Penrose**, e, poiché permette di ottenere una relazione del tipo

$$\underline{x}^* = \underline{\underline{A}}^+\underline{b},$$

essa rappresenta una generalizzazione del concetto di matrice inversa. Non solo: è possibile dimostrare che, se $\underline{\underline{A}}$ è una matrice quadrata e a rango pieno,

$$\underline{\underline{A}}^+ = \underline{\underline{A}}^{-1},$$

a ulteriore testimonianza di quanto appena detto. MATLAB® permette di calcolare questa matrice, mediante il comando `pinv`.

[4] interpretato come prodotto degli autovalori

[5] ma la cosa è totalmente sconsigliata, come abbiamo discusso in passato

Attenti alle scorciatoie!

Abbiamo appena completato l'argomento *sistemi lineari sovradeterminati*; tuttavia, vale la pena di proporre una piccola precisazione. Abbiamo visto, dopo aver introdotto la fattorizzazione QR, che l'implementazione può essere semplificata utilizzando il comando \, che riconosce automaticamente il fatto che il sistema è sovradeterminato e *internamente* effettua tutti i passaggi descritti. Tuttavia, **questo è vero solamente quando la matrice associata al sistema ha rango massimo**.

In verità, il comando \ permette di trovare **una** soluzione anche nel caso in cui il rango non sia massimo; tuttavia, **questa non sarà la soluzione avente norma minima**. Al fine di dimostrarlo, si propone il seguente script.

```
clear
close all
clc

A = [1   2  3;
     4   5  6;
     7   8  9;
     10 11 12]; % esempio di sistema sovradeterminato la cui matrice non ha
                % rango massimo

 b = [1:4]'; % esempio di termine noto

 r = rank(A) % dimostriamo, mediante il comando rank, che il rango
             % effettivamente e` inferiore a 3 (numero di colonne di A) !

 xBS = A\b % calcolo soluzione mediante il comando \
 xPINV = pinv(A)*b % calcolo soluzione mediante psuedo-inversa
```

Possiamo vedere che questo script visualizzerà:

```
xBS =
   -0.0000
         0
    0.3333

xPINV =
   -0.0556
    0.1111
    0.2778

>>
```

che sono soluzioni chiaramente diverse! Prima di tutto: è vero che entrambe sono davvero soluzioni del sistema lineare? Per farlo, proviamo a eseguire i seguenti comandi:

```
>> norm(A*xBS-b)
ans =
   9.1551e-16

>> norm(A*xPINV-b)
ans =
   3.0445e-15

>>
```

Entrambe le soluzioni sono valide: la norma 2 del residuo è sostanzialmente pari a zero. Tuttavia proviamo a guardare non solo la norma del residuo, ma anche la norma delle soluzioni; otteniamo:

```
>> norm(xBS)
ans =
    0.3333

>> norm(xPINV)
```

```
ans =
    0.3043

>>
```

Questo significa, come abbiamo già anticipato, che il comando \ trova sì una soluzione, **ma non quella di norma minima**! Si sia dunque consapevoli di questo, al momento di cercare delle *scorciatoie* per la risoluzione di problemi e/o esercizi!

3.3.7 Risoluzione di un sistema lineare sottodeterminato

La decomposizione ai valori singolari permette non solo di risolvere un sistema sovradeterminato, ma anche uno sottodeterminato, ammesso che la sua matrice abbia rango massimo! Considerando quindi un sistema nella forma

$$\underline{\underline{A}}\,\underline{x} = \underline{b},$$

dove $\underline{\underline{A}} \in \mathbb{R}^{m,n}$ ma, in questo caso, $m < n$, è possibile applicare la seguente procedura. Sostituendo $\underline{\underline{A}}$ con la sua SVD, si ha

$$\underline{\underline{U}}\,\underline{\underline{S}}\,\underline{\underline{V}}^{\mathrm{T}}\underline{x} = \underline{b},$$

che diventa, come già mostrato,

$$\underline{\underline{S}}\,\underline{\underline{V}}^{\mathrm{T}}\underline{x} = \underline{\underline{U}}^{\mathrm{T}}\underline{b}.$$

Come nel caso dei sistemi lineari determinati, si può quindi definire $\underline{y} = \underline{\underline{V}}^{\mathrm{T}}\underline{x}$, $\underline{c} = \underline{\underline{U}}^{\mathrm{T}}\underline{b}$, e risolvere i due sistemi lineari

$$\begin{cases} \underline{\underline{S}}\,\underline{y} = \underline{c} \\ \underline{\underline{V}}^{\mathrm{T}}\underline{x} = \underline{y}. \end{cases}$$

Proprio come fatto per quanto riguarda i sistemi sovradeterminati, è possibile accorgersi che la matrice $\underline{\underline{S}}$ è decomponibile nei blocchi

$$\underline{\underline{S}} = \left[\underline{\underline{\widetilde{S}}} \quad \underline{\underline{0}}\right],$$

dove $\underline{\underline{\widetilde{S}}}$ è il ritaglio $r \times r$ di $\underline{\underline{S}}$, dunque è diagonale con elementi non nulli. Quindi, si può trovare la soluzione $\underline{y}^*$ avente minima norma del sistema

$$\underline{\underline{\widetilde{S}}}\,\underline{y}^* = \underline{c}, \tag{8}$$

usando la relazione

$$y_i^* = \begin{cases} \dfrac{c_i}{s_i}, & i = 1, \ldots, r \\ 0, & i = r+1, \ldots, n. \end{cases}$$

Infine, trovato $\underline{y}^*$ in questo modo, si ottiene

$$\underline{x}^* = \underline{\underline{V}}\,\underline{y}^*.$$

L'ipotesi sulla matrice $\underline{\underline{A}}$ di essere di rango massimo è molto importante, nonostante per ora apparentemente ingiustificata. In effetti, se la matrice fosse di rango non massimo, la matrice $\underline{\underline{S}}$ non sarebbe più decomponibile come appena mostrato, ma avrebbe una forma simile a quella dei sistemi sovradeterminati:

$$\underline{\underline{S}} = \begin{bmatrix} \underline{\underline{\widetilde{S}}} & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} \end{bmatrix}.$$

Questa cosa sarebbe molto problematica, dal momento che non ci permetterebbe più di scrivere il sistema sottodeterminato nella forma (8), per via dei valori singolari nulli, generati dalle dipendenze lineari tra le righe di $\underline{\underline{A}}$. In particolare, eliminando l'ipotesi di rango massimo, perderemmo la garanzia di avere un sistema lineare dotato di soluzione. Infatti, volendo scrivere esplicitamente il sistema, avremmo più di r equazioni e, quindi, una forma del tipo[6]

$$\begin{cases} s_1 y_1^* = c_1 \\ s_2 y_2^* = c_2 \\ \vdots \\ s_r y_r = c_r \\ s_{r+1} y_{r+1} = c_{r+1} \\ \vdots \\ s_m y_m = c_m. \end{cases}$$

Fino alla r-esima equazione, dove r come al solito indica il rango della matrice, nessun problema! Tuttavia, $s_{r+1} = s_{r+2} = ... = s_m = 0$ e, quindi, le ultime $m - r$ equazioni sarebbero

$$\begin{cases} 0 \times y_{r+1} = c_{r+1} \\ \vdots \\ 0 \times y_m = c_m, \end{cases}$$

dove, a meno che per qualche fortuito caso i c_i siano uguali a zero, saremmo dinnanzi a un assurdo: il sistema non avrebbe soluzione! Per questo, l'ipotesi di rango massimo, ovvero $r = m$, è molto importante!

A Applicazione della SVD alla compressione di immagini

Una delle proprietà della decomposizione ai valori singolari riguarda la possibilità di trovare la migliore approssimante $\underline{\underline{A}}_k$ avente rango massimo k di una matrice $\underline{\underline{A}}$. Un'applicazione di questa fantastica idea è la compressione di immagini. Infatti, dal punto di vista di un computer, un'immagine è descrivibile mediante il linguaggio matriciale. In particolare, negli schermi, ogni colore viene ottenuto mescolando additivamente il rosso, il verde e il blu, portandoci al sistema RGB (red, green, blue)[7]. Ciò che possiamo fare è dunque associare una matrice alle componenti di rosso, una ai verdi e una ai blu, e calcolarne la SVD; quindi, utilizzando il teorema (1) è possibile ricostruirne una versione di rango inferiore, arrivando a comprimere l'immagine.

Come possiamo quantificare l'efficacia della compressione? Compressione significa rapporto e, quindi, consideriamo come *dimensione su disco* della figura la dimensione della matrice, memorizzata in doppia precisione, su MATLAB®. Questo non è davvero il *peso* del file `.jpg`, in quanto i JPEG sono già un formato compresso, e non è detto che i numeri siano memorizzati in doppia precisione: per applicazioni di questo tipo probabilmente gli interi a 8 bit, ottenibili a partire dai `double` mediante il comando `uint8`, sono più che sufficienti. Tuttavia, visto che ci interessiamo solo al rapporto tra il *peso* dell'immagine di partenza e di quella compressa, il tipo di memorizzazione non è troppo rilevante, a patto di essere coerenti tra numeratore e denominatore.

Volendo prendere questa strada, ricordiamo che un numero in doppia precisione richiede 64 bit di memoria[8], ovvero 8 byte. Memorizzare una figura in una matrice di numeri a doppia precisione $\underline{\underline{A}}$ richiede $24 P_x P_y$ byte, dove P_x e P_y sono i numeri di pixel lungo ascisse e ordinate, e che quindi saranno il numero di colonne e il numero di righe della nostra matrice, rispettivamente: un numero per ogni pixel; il 24 nasce come 8×3: 8 byte, per 3 colori.

Al momento di calcolare la decomposizione ai valori singolari di questa matrice $\underline{\underline{A}}$, otteniamo:

$$\underline{\underline{A}} = \underline{\underline{U}}\,\underline{\underline{S}}\,\underline{\underline{V}}^{\mathrm{T}},$$

[6] dove le equazioni sono così semplici grazie al fatto che sono costruite a partire dalla matrice $\underline{\underline{S}}$, che è diagonale -o quasi-!

[7] in cartellonistica per esempio si utilizza il sistema sottrattivo ciano-magenta-giallo CMY (cyan, magenta, yellow)

[8] non per nulla negli ultimi anni si utilizzano architetture a 64 bit!

dove $\underline{\underline{U}}$ sarà una matrice quadrata di dimensioni $P_y \times P_y$, $\underline{\underline{V}}$ sarà ancora quadrata e di dimensioni $P_x \times P_x$, e $\underline{\underline{S}}$ sarà rettangolare ma avente elementi non nulli solo sulla diagonale principale. Sfruttando il teorema (1), invece di considerare tutte le colonne, possiamo decidere di prenderne solo k, e ricostruire quindi la matrice di rango k più simile rispetto alla norma spettrale. Dover salvare solo le matrici $\underline{\underline{U}}^{(k)}$, $\underline{\underline{S}}^{(k)}$ e $\underline{\underline{V}}^{(k)}$ al posto di $\underline{\underline{A}}$ richiede, dal punto di vista della memoria,

$$24(kP_x + kP_y + k),$$

sfruttando il fatto che ritagliamo solo k colonne di $\underline{\underline{U}}$ (ciascuna delle quali ha P_y componenti), k colonne di $\underline{\underline{V}}$ (ciascuna delle quali ha P_x componenti), più k valori singolari; questo, per 8 byte e 3 colori.

Segue un codice che effettua tutti questi passaggi e permette di visualizzare le immagini compresse.

```
clear
close all
clc

vr=[1 2 5 10 20 30 150]; % vettore con ranghi per approssimazione

A = imread('demo1.jpg'); % carico la figura nella matrice A

% A e` una matrice di matrici, ovvero una matrice a 3 indici; da questa si
% possono ricavare tre matrici "a due indici", corrispondenti alle
% componenti R, G, B dei colori.
% Il comando "squeeze" serve a ottenere una matrice a 2 indici (vedi help)
A_R=squeeze(A(:,:,1)); % blackness matrix color R
A_G=squeeze(A(:,:,2)); % blackness matrix color G
A_B=squeeze(A(:,:,3)); % blackness matrix color B

% La SVD va applicata a numeri "reali", ovvero "double", quindi bisogna
% fare un cast a double prima di calcolarla
A_R=double(A_R);
A_G=double(A_G);
A_B=double(A_B);

% Calcolo la SVD per ciascuna delle matrici di colori
[U_R,S_R,V_R] = svd(A_R);
[U_G,S_G,V_G] = svd(A_G);
[U_B,S_B,V_B] = svd(A_B);

% Calcolo la dimensione totale della figura non compressa
[Py,Px] = size(A_R); % le tre matrici hanno lo stesso numero di pixel e
                     % sono tutte memorizzate in doppia precisione quindi
                     % posso calcolare Px, Py solo da una di esse!

DimTotale = 8*3*Px*Py; % numero totale di pixel per 24 byte

% Applichiamo a questo punto l'algoritmo di approssimazione della matrice;
% proviamo per matrici approssimanti avente diverso rango a verificare la
% qualita` della ricostruzione
for indn=1:length(vr)

    r=vr(indn);
    A_R_r=U_R(:,1:r)*S_R(1:r,1:r)*V_R(:,1:r)';
    A_G_r=U_G(:,1:r)*S_G(1:r,1:r)*V_G(:,1:r)';
    A_B_r=U_B(:,1:r)*S_B(1:r,1:r)*V_B(:,1:r)';

    % Per la visualizzazione, bisogna riconvertire "all'indietro" in numeri
    % interi, e salvare in quella specie di matrice di matrici!
    Ar(:,:,1)=uint8(A_R_r);
    Ar(:,:,2)=uint8(A_G_r);
    Ar(:,:,3)=uint8(A_B_r);

    DimApp = (Px*r + Py*r + r)*8*3;

    figure(1)
    clf % cancello contenuto figura precedente
    imshow(Ar) % mostro la figura approssimata in Ar
    title(['Approssimazione a rango ',num2str(r), ', compressione al ',num2str(DimApp/DimTotale*100)↩
        ,' %'])
    pause

end
```

Qualche commento conclusivo. Il comando

```
A = imread('nomefile.jpg');
```

ordina a MATLAB®di leggere la figura e salvarla nella matrice `A`; questa è una specie di *matrice di matrici*, cioè una *matrice a tre indici*. In verità non è nulla di troppo complicato: si tratta di tre matrici, ciascuna delle quali contiene le informazioni su un colore. Volendo, il programma si può semplificare e portare in bianco e nero, usando il comando

```
B = rgb2gray(A);
```

che converte le tre matrici in un'unica matrice, che verrà interpretata come scala di grigi. Le matrici generate con queste funzioni sono in un formato particolare, quindi è necessario effettuare dei cast a `double` al momento di elaborarle, o a `uint8` al momento di visualizzarle, per esempio con il comando `imshow`.