Singular Value Decomposition

Marco Casamassima 730346

Questo documento si divide in tre parti: una parte riguardante gli aspetti teorici della Decomposizione a Valori Singolari, una sulle sue applicazioni e l'ultima sulla sua implementazione in R.

1 Aspetti teorici

La *Decomposizione a Valori Singolari* (SVD) è una tecnica di algebra lineare che permette di scomporre una qualsiasi matrice nel prodotto di tre matrici (Figura 1). In particolare, sia $\mathbf{A} \in \mathbb{R}^{n \times m}$ (o $\mathbf{A} \in \mathbb{C}^{n \times m}$) con $rank(\mathbf{A}) = k <= min(n, m)$, è possibile scomporla in

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{T(H)}$$

(In seguito si considereranno solo matrici reali).

Dove:

- $\mathbf{U} = \left[\mathbf{u}_1|\mathbf{u}_2|...|\mathbf{u}_n\right] \in \mathbb{R}^{n \times n}$ ortogonale, ossia $\mathbf{u}_i \in \mathbb{R}^n$ *t.c.*

  $$\mathbf{u}_i^T * \mathbf{u}_j = \begin{cases} 1 & se\ i = j \\ 0 & se\ i \neq j \end{cases}$$

  I $\mathbf{u}_i$ vengono chiamati *vettori singolari sinistri* di $\mathbf{A}$;
- $\mathbf{\Sigma} = \left[\begin{array}{c|c} \mathbf{\Sigma}_k & \mathbf{0}_{k \times (m-k)} \\ \hline \mathbf{0}_{(n-k) \times k} & \mathbf{0}_{(n-k) \times (m-k)} \end{array}\right] \in \mathbb{R}^{n \times m}$

  con $\mathbf{\Sigma}_k = \begin{bmatrix} \sigma_1 & \dots & 0 \\ \dots & \dots & \dots \\ 0 & \dots & \sigma_k \end{bmatrix} \in \mathbb{R}^{k \times k}$ diagonale con i k elementi $\sigma_1 \geq ... \geq \sigma_k > 0$.

  I σ_i vengono chiamati *valori singolari* di $\mathbf{A}$. Inoltre, $\mathbf{\Sigma}$ è unicamente determinata;
- $\mathbf{V} = \left[\mathbf{v}_1|\mathbf{v}_2|...|\mathbf{v}_m\right] \in \mathbb{R}^{m \times m}$ ortogonale, ossia $\mathbf{v}_i \in \mathbb{R}^m$ *t.c.*

  $$\mathbf{v}_i^T * \mathbf{v}_j = \begin{cases} 1 & se\ i = j \\ 0 & se\ i \neq j \end{cases}$$

  i $\mathbf{v}_i$ vengono chiamati **vettori singolari destri** di $\mathbf{A}$.

Questa versione della SVD in cui si considerano tutte le colonne e le righe di $\mathbf{U}$, $\mathbf{\Sigma}$ e $\mathbf{V}$ viene chiamata *Full SVD*. In seguito si vedranno altre versioni.

Osservazione:
I vettori singolari sinistri e destri hanno un comportamente simile a quello degli autovettori, con i valori singolari che si comportano da autovalori. Infatti:

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$

$$\mathbf{AV} = \mathbf{U\Sigma V}^T\mathbf{V} = \mathbf{U\Sigma}$$
$$\Rightarrow \mathbf{Av}_i = \sigma_i \mathbf{u}_i \quad \forall i = 1...k$$

$$\mathbf{A}^T = \mathbf{V\Sigma U}^T$$
$$\mathbf{A}^T\mathbf{U} = \mathbf{V\Sigma U}^T\mathbf{U} = \mathbf{V\Sigma}$$
$$\Rightarrow \mathbf{A}^T\mathbf{u}_i = \sigma_i \mathbf{v}_i \quad \forall i = 1...k$$

Come visto in precedenza, la matrice $\mathbf{\Sigma}$ è formata da $\sigma_1, \ldots, \sigma_k \neq 0$ e $\sigma_i = 0$ per $i > k$. Di conseguenza, considerando solo le prime k colonne di $\mathbf{U}$ e le prime k righe di $\mathbf{V}^T$, $\mathbf{A}$ può essere decomposta come

$$\mathbf{A} = \mathbf{U}_{n\times k}\mathbf{\Sigma}_k\mathbf{V}^T_{k\times m} =$$

(Per $\mathbf{V}^T_{k\times m}$ si intendono le prime k righe e m colonne di $\mathbf{V}^T$)

$$= \left[\mathbf{u}_1|\mathbf{u}_2|...|\mathbf{u}_k\right] \begin{bmatrix} \sigma_1 & \ldots & 0 \\ \ldots & \ldots & \ldots \\ 0 & \ldots & \sigma_k \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^T \\ ... \\ \mathbf{v}_k^T \end{bmatrix} =$$

$$= \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

Questa variante della SVD è chiamata *Compact SVD* (figura 2).

Osservazione:
Nel caso in cui $k = n = m$, la Compact SVD è uguale alla Full SVD.

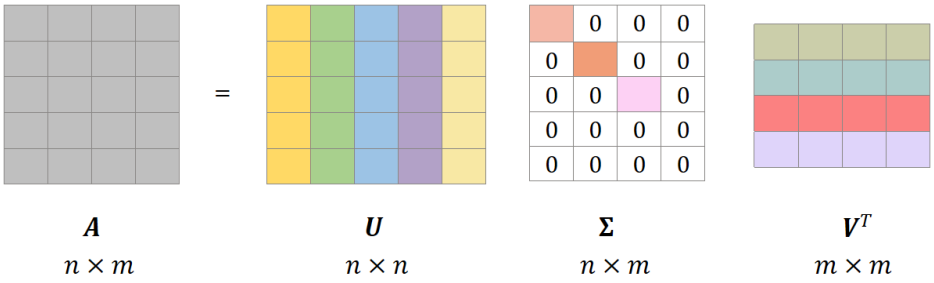

Figure 1: Visualizzazione della SVD di una matrice di rango k<min(n,m).

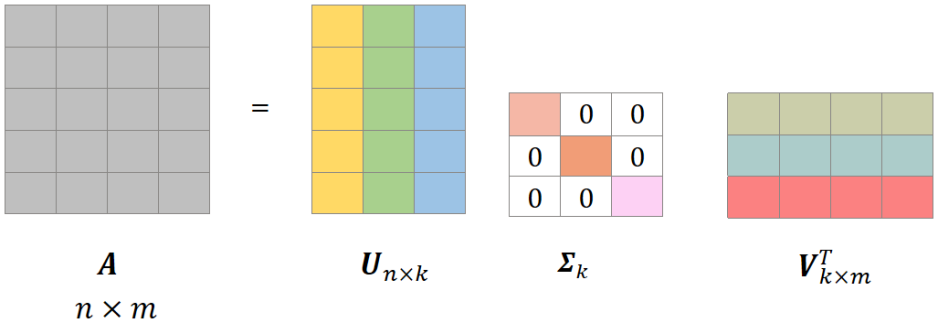

Figure 2: Visualizzazione della Compact SVD di una matrice di rango k<min(n,m).

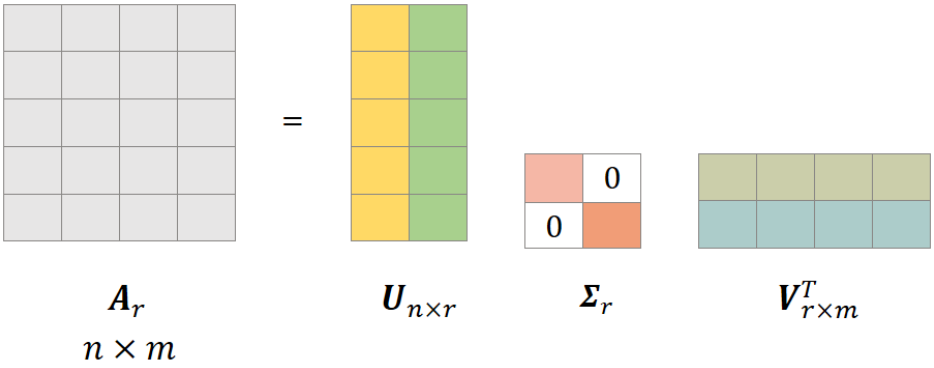

Figure 3: Visualizzazione della Truncated SVD di una matrice di rango k<min(n,m) utilizzando r<k valori singolari.

1.1 Calcolo di $\mathbf{U}, \mathbf{\Sigma}$ e $\mathbf{V}$

Sia $\mathbf{A} \in \mathbb{R}^{n \times m}$ con $rank(\mathbf{A}) = k \leq min(n, m)$. Le tre matrici $\mathbf{U}, \mathbf{\Sigma}$ e $\mathbf{V}$ utilizzate nella Decomposizione a Valori Singolari possono essere calcolate considerando le matrici $\mathbf{A}^T\mathbf{A}$ e $\mathbf{A}\mathbf{A}^T$ e utilizzando la definizione di matrice diagonalizzabile.

Definizione: Due matrici $\mathbf{A}$, $\mathbf{B}$ di ordine n si dicono *simili* se esiste una matrice invertibile $\mathbf{P}$ con la proprietà che $\mathbf{P}^{-1}\mathbf{A}\mathbf{P} = \mathbf{B}$. In particolare, se $\mathbf{B}$ è una matrice diagonale, $\mathbf{A}$ si dice *diagonalizzabile*. Quest'ultima definizione risulta molto utile perché, se $\mathbf{A}$ è diagonalizzabile, i suoi autovalori si trovano sulla diagonale di $\mathbf{B}$ e i suoi autovettori sono le colonne di $\mathbf{P}$.

Il calcolo di $\mathbf{U}, \mathbf{\Sigma}$ e $\mathbf{V}$ può essere effettuato nel seguente modo:

- Le colonne di $\mathbf{U}$ sono gli autovettori della matrice $\mathbf{A}\mathbf{A}^T$.

  Dimostrazione
  Si consideri $\mathbf{A}\mathbf{A}^T$ e si applichi la SVD su $\mathbf{A}$

  $$\mathbf{A}\mathbf{A}^T = (\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T)(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T)^T =$$

  $$= \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\mathbf{V}\mathbf{\Sigma}^T\mathbf{U}^T =$$

  Poiché $\mathbf{V}$ è ortogonale, $\mathbf{V}\mathbf{V}^T = \mathbf{V}^T\mathbf{V} = \mathbf{I}_m$

  $$= \mathbf{U}\mathbf{\Sigma}\mathbf{\Sigma}^T\mathbf{U}^T =$$

  $$= \mathbf{U} \left[\begin{array}{c|c} \mathbf{\Sigma}_k & \mathbf{0}_{k\times(m-k)} \\ \hline \mathbf{0}_{(n-k)\times k} & \mathbf{0}_{(n-k)\times(m-k)} \end{array}\right] \left[\begin{array}{c|c} \mathbf{\Sigma}_k & \mathbf{0}_{k\times(n-k)} \\ \hline \mathbf{0}_{(m-k)\times k} & \mathbf{0}_{(m-k)\times(n-k)} \end{array}\right] \mathbf{U}^T =$$

  $$= \mathbf{U} \left[\begin{array}{c|c} \mathbf{\Sigma}_k\mathbf{\Sigma}_k & \mathbf{0}_{k\times(n-k)} \\ \hline \mathbf{0}_{(n-k)\times k} & \mathbf{0}_{(n-k)\times(n-k)} \end{array}\right] \mathbf{U}^T$$

  $\mathbf{A}\mathbf{A}^T$ è quindi simile ad una matrice diagonale, ossia $\left[\begin{array}{c|c} \mathbf{\Sigma}_k\mathbf{\Sigma}_k & \mathbf{0}_{k\times(n-k)} \\ \hline \mathbf{0}_{(n-k)\times k} & \mathbf{0}_{(n-k)\times(n-k)} \end{array}\right]$, di conseguenza i suoi autovettori saranno le colonne di $\mathbf{U}$.

- Le colonne di $\mathbf{V}$ sono gli autovettori della matrice $\mathbf{A}^T\mathbf{A}$.

  Dimostrazione
  Si consideri $\mathbf{A}^T\mathbf{A}$ e si applichi la SVD su $\mathbf{A}$

  $$\mathbf{A}^T\mathbf{A} = (\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T)^T(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T) =$$

  $$= \mathbf{V}\mathbf{\Sigma}^T\mathbf{U}^T\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T =$$

  $\mathbf{U}$ è ortogonale, $\mathbf{U}\mathbf{U}^T = \mathbf{U}^T\mathbf{U} = \mathbf{I}_n$

  $$= \mathbf{V}\mathbf{\Sigma}^T\mathbf{\Sigma}\mathbf{V}^T =$$

  $$= \mathbf{V} \left[\begin{array}{c|c} \mathbf{\Sigma}_k & \mathbf{0}_{k\times(n-k)} \\ \hline \mathbf{0}_{(m-k)\times k} & \mathbf{0}_{(m-k)\times(n-k)} \end{array}\right] \left[\begin{array}{c|c} \mathbf{\Sigma}_k & \mathbf{0}_{k\times(m-k)} \\ \hline \mathbf{0}_{(n-k)\times k} & \mathbf{0}_{(n-k)\times(m-k)} \end{array}\right] \mathbf{U}^T =$$

$$= \mathbf{V} \left[\begin{array}{c|c} \mathbf{\Sigma}_k \mathbf{\Sigma}_k & \mathbf{0}_{k\times(m-k)} \\ \hline \mathbf{0}_{(m-k)\times k} & \mathbf{0}_{(m-k)\times(m-k)} \end{array} \right] \mathbf{V}^T$$

$\mathbf{A}^T\mathbf{A}$ è quindi simile ad una matrice diagonale, ossia $\left[\begin{array}{c|c} \mathbf{\Sigma}_k \mathbf{\Sigma}_k & \mathbf{0}_{k\times(m-k)} \\ \hline \mathbf{0}_{(m-k)\times k} & \mathbf{0}_{(m-k)\times(m-k)} \end{array} \right]$, di conseguenza i suoi autovettori saranno le colonne di $\mathbf{V}$.

- Sulla diagonale di $\mathbf{\Sigma}$ sono presenti le radici quadrate degli autovalori di $\mathbf{A}^T\mathbf{A}$ (e $\mathbf{A}^T\mathbf{A}$).

  Dimostrazione
  Come dimostrato nei precedenti due punti, $\mathbf{A}^T\mathbf{A}$ e $\mathbf{A}\mathbf{A}^T$ sono rispettivamente simili alle matrici $\left[\begin{array}{c|c} \mathbf{\Sigma}_k \mathbf{\Sigma}_k & \mathbf{0}_{k\times(n-k)} \\ \hline \mathbf{0}_{(n-k)\times k} & \mathbf{0}_{(n-k)\times(n-k)} \end{array} \right]$ e $\left[\begin{array}{c|c} \mathbf{\Sigma}_k \mathbf{\Sigma}_k & \mathbf{0}_{k\times(m-k)} \\ \hline \mathbf{0}_{(m-k)\times k} & \mathbf{0}_{(m-k)\times(m-k)} \end{array} \right]$, e pertanto ne condividono gli autovalori, ossia gli elementi sulla diagonale principale.

Nel paragrafo precedente, è stato mostrato che esiste una relazione tra i vettori singolari sinistri, i vettori singolari destri e i valori singolari, data da $\mathbf{A}\mathbf{v}_i = \sigma_i \mathbf{u}_i$. Da ciò risulta che, dopo aver calcolato le matrici $\mathbf{V}_{m\times k}$ e $\mathbf{\Sigma}_k$, è possibile calcolare le colonne di $\mathbf{U}_{n\times k}$ attraverso $\mathbf{u}_i = \frac{\mathbf{A}\mathbf{v}_i}{\sigma_i}$ per $i = 1...k$.

1.2 Sottospazi fondamentali

La Decomposizione a Valori Singolari permette di rappresentare i quattro spazi fondamentali associati alle righe e alle colonne di $\mathbf{A}$ ($Null(\mathbf{A})$, $Null(\mathbf{A}^T)$, $Range(\mathbf{A})$ e $Range(\mathbf{A}^T)$) attraverso nuove basi ortonormali. Sia $\mathbf{A} \in \mathbb{R}^{n\times m}$ con $rank(\mathbf{A}) = k <= min(n, m)$:

- $Null(\mathbf{A}) = \{\mathbf{x} \in \mathbb{R}^m | \mathbf{A}\mathbf{x} = \mathbf{0}_n\} = span\{\mathbf{v}_{k+1}, \mathbf{v}_{k+2}, ..., \mathbf{v}_m\}$.
  Dimostrazione
  Sia $\mathbf{x} \in \mathbb{R}^m$ *t.c.* $\mathbf{x} \in Null(\mathbf{A})$. Quindi,

$$\mathbf{A}\mathbf{x} = \mathbf{0}_n$$

$$\mathbf{U}_{n\times k}\mathbf{\Sigma}_k\mathbf{V}^T_{k\times m}\mathbf{x} = \mathbf{0}_n$$

  Poiché $\mathbf{U}_{n\times k}$ è ortonormale, $\exists \mathbf{U}^+ = \mathbf{U}^T$

$$\mathbf{U}^T_{k\times n}\mathbf{U}_{n\times k}\mathbf{\Sigma}_k\mathbf{V}^T_{k\times m}\mathbf{x} = \mathbf{0}_k$$

$$\mathbf{\Sigma}_k\mathbf{V}^T_{k\times m}\mathbf{x} = \mathbf{0}_k$$

  $\mathbf{\Sigma}_k$ è una matrice diagonale con gli elementi $\sigma_1, \sigma_2, \ldots, \sigma_k \neq 0$, di conseguenza $det(\mathbf{\Sigma}_k) \neq 0 \iff \exists \mathbf{\Sigma}_k^{-1}$

$$\mathbf{\Sigma}_k^{-1}\mathbf{\Sigma}_k\mathbf{V}^T_{k\times m}\mathbf{x} = \mathbf{0}_k$$

$$\mathbf{V}^T_{k\times m}\mathbf{x} = \mathbf{0}_k$$

$$\begin{bmatrix}\mathbf{v}_1^T\\ ...\\ \mathbf{v}_k^T\end{bmatrix}\mathbf{x}=\mathbf{0}_k \iff \forall i=1...k\ \ \mathbf{v}_i^T\mathbf{x}=0 \implies \mathbf{x}\perp\mathbf{v}_i^T$$

Di conseguenza, poiché $\forall i=1\dots m$ i $\mathbf{v}_i$ sono ortonormali tra loro e $dim(Null(\mathbf{A}))=m-k$, necessariamente $\mathbf{x}\in span\{\mathbf{v}_{k+1},\mathbf{v}_{k+2},...,\mathbf{v}_m\}$.

- $Null(\mathbf{A}^T)=\{\mathbf{x}\in\mathbb{R}^n|\mathbf{A}^T\mathbf{x}=\mathbf{0}_m\}=span\{\mathbf{u}_{k+1},\mathbf{u}_{k+2},...,\mathbf{u}_n\}$.

  Dimostrazione
  Sia $\mathbf{x}\in\mathbb{R}^n$ *t.c.* $\mathbf{x}\in Null(\mathbf{A}^T)$. Quindi,

$$\mathbf{A}^T\mathbf{x}=\mathbf{0}_m$$

$$\mathbf{V}_{m\times k}\mathbf{\Sigma}_k\mathbf{U}_{k\times n}^T\mathbf{x}=\mathbf{0}_m$$

Poiché $\mathbf{V}_{m\times k}$ è ortonormale, $\exists\mathbf{V}^+=\mathbf{V}^T$

$$\mathbf{V}_{k\times m}^T\mathbf{V}_{m\times k}\mathbf{\Sigma}_k\mathbf{U}_{k\times n}^T\mathbf{x}=\mathbf{0}_k$$

$$\mathbf{\Sigma}_k\mathbf{U}_{k\times n}^T\mathbf{x}=\mathbf{0}_k$$

$\mathbf{\Sigma}_k$ è una matrice diagonale con gli elementi $\sigma_1,\sigma_2,\dots,\sigma_k\neq 0$, di conseguenza $det(\mathbf{\Sigma}_k)\neq 0 \iff \exists\mathbf{\Sigma}_k^{-1}$

$$\mathbf{\Sigma}_k^{-1}\mathbf{\Sigma}_k\mathbf{U}_{k\times n}^T\mathbf{x}=\mathbf{0}_k$$

$$\mathbf{U}_{k\times n}^T\mathbf{x}=\mathbf{0}_k$$

$$\begin{bmatrix}\mathbf{u}_1^T\\ ...\\ \mathbf{u}_k^T\end{bmatrix}\mathbf{x}=\mathbf{0} \iff \forall i=1...k\ \ \mathbf{u}_i^T\mathbf{x}=0 \implies \mathbf{x}\perp\mathbf{u}_i^T$$

Di conseguenza, poiché $\forall i=1\dots n$ i $\mathbf{u}_i$ sono ortonormali tra loro e $dim(Null(\mathbf{A}^T))=n-k$, necessariamente $\mathbf{x}\in span\{\mathbf{u}_{k+1},\mathbf{u}_{k+2},...,\mathbf{u}_n\}$.

- $Range(\mathbf{A})=\{\mathbf{y}\in\mathbb{R}^n|\exists\mathbf{x}\in\mathbb{R}^m\ t.c.\ \mathbf{A}\mathbf{x}=\mathbf{y}\}=span\{\mathbf{u}_1,\mathbf{u}_2,...,\mathbf{u}_k\}$.

  Dimostrazione
  Sia $\mathbf{y}\in\mathbb{R}^n$ *t.c.*

$$\mathbf{y}=\mathbf{A}\mathbf{x}=$$

$$=\mathbf{U}_{n\times k}\mathbf{\Sigma}_k\mathbf{V}_{k\times m}^T\mathbf{x}=$$

Sia $\mathbf{z}:=\mathbf{\Sigma}_k\mathbf{V}_{k\times m}^T\mathbf{x}$ con $\mathbf{z}\in\mathbb{R}^k$

$$=\mathbf{U}_{n\times k}\mathbf{z}==\mathbf{u}_1z_1+...+\mathbf{u}_kz_k$$

Quindi $\mathbf{y} \in span\{\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_k\}$.

- $Range(\mathbf{A}^T) = \{\mathbf{y} \in \mathbb{R}^m | \exists \mathbf{x} \in \mathbb{R}^n \ t.c. \ \mathbf{A}^T\mathbf{x} = \mathbf{y}\} = span\{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k\}$.
  Dimostrazione
  Sia $\mathbf{y} \in \mathbb{R}^m$ *t.c.*

$$\mathbf{y} = \mathbf{A}^T\mathbf{x} =$$

$$= \mathbf{V}_{m\times k}\mathbf{\Sigma}_k\mathbf{U}^T_{k\times n}\mathbf{x} =$$

  Sia $\mathbf{z} := \mathbf{\Sigma}_k\mathbf{U}^T_{k\times n}\mathbf{x}$ con $\mathbf{z} \in \mathbb{R}^k$

$$= \mathbf{V}_{m\times k}\mathbf{z} = \ = \mathbf{v}_1 z_1 + ... + \mathbf{v}_k z_k$$

  Quindi $\mathbf{y} \in span\{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k\}$.

1.3 Teorema di Eckart-Young

Le principali applicazioni della Decomposizione a Valori Singolari provengono dall'utilizzo del seguente teorema.
Teorema di Eckart-Young: la Decomposizione a Valori Singolari ottenuta scartando tutti i valori singolari più piccoli di σ_r e i corrispondenti vettori singolari destri e sinistri, è la migliore approssimazione di rango r della matrice originale, con r minore del rango della matrice originale. Quindi, siano $\mathbf{A} \in \mathbb{R}^{n\times m}$ con $rank(\mathbf{A}) = k \leq min(n, m)$, $r \in \mathbb{N}$ *t.c* $r < k$ e $\mathbf{B} \in \mathbb{R}^{n\times m}$ con $rank(\mathbf{B}) = r$, il teorema garantisce che, utilizzando la norma 2,

$$min\|\mathbf{A} - \mathbf{B}\|_2^2 \equiv \|\mathbf{A} - \mathbf{A}_r\|_2^2$$

o, utilizzando la norma di Frobenius,

$$min\|\mathbf{A} - \mathbf{B}\|_F^2 \equiv \|\mathbf{A} - \mathbf{A}_r\|_F^2$$

con $\mathbf{A}_r = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$.
L'errore che si commette considerando $\mathbf{A}_r$ invece di $\mathbf{A}$ è dato da:

$$\epsilon = \|\mathbf{A} - \mathbf{A}_r\|_2^2 = \sigma_{r+1}^2$$

$$\epsilon = \|\mathbf{A} - \mathbf{A}_r\|_F^2 = \sum_{j=r+1}^{k} \sigma_{r+1}^2$$

La versione della SVD in cui si calcola $\mathbf{A}_r = \mathbf{U}_{n\times r}\mathbf{\Sigma}_r\mathbf{V}^T_{r\times m}$ da $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ è chiamata *Truncated SVD* (figura 3).

1.4 Geometria della Decomposizione a Valori Singolari in $\mathbb{R}^2$

Qualsiasi matrice $\mathbf{A} \in \mathbb{R}^{n \times m}$ è associata ad una trasformazione lineare da $\mathbb{R}^m$ a $\mathbb{R}^n$. La Decomposizione a Valori Singolari permette, quindi, di scomporre questa trasformazione in una composizione di tre trasformazioni geometriche: una rotazione, una scalatura ed un'altra rotazione. Per comprendere come agiscono geometricamente le matrici $\mathbf{U}, \mathbf{\Sigma}$ e $\mathbf{V}^T$, si consideri $\mathbf{A} \in \mathbb{R}^{2 \times 2}$, con $rank(\mathbf{A}) = 2$ e $\mathbf{x} \in \mathbb{R}^2$ *t.c.* $\|\mathbf{x}\| = 1$.
Applicando la SVD su $\mathbf{A}$ si avranno le seguenti matrici:

$$\mathbf{U} = [u_1|u_2]$$

$$\mathbf{V} = [v_1|v_2]$$

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix}$$

Si consideri $\mathbf{Ax} = \mathbf{U\Sigma V}^T\mathbf{x}$:

1. $\mathbf{z} := \mathbf{V}^T\mathbf{x} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$: $\mathbf{V}$ è una matrice ortogonale, questo vuol dire che, moltiplicandola per $\mathbf{x}$, non cambia modulo di quest'ultimo ($\|\mathbf{Vx}\|_2 = \|\mathbf{x}\|_2$), ma ne effettua solo una rotazione.
2. $\mathbf{y} := \mathbf{\Sigma z} = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} \sigma_1 z_1 \\ \sigma_2 z_2 \end{bmatrix}$: $\mathbf{\Sigma}$ allunga o accorcia le componenti di $\mathbf{z}$ di una quantità che dipende dai valori singolari (σ_1 avrà un contributo maggiore o uguale di σ_2, poiché $\sigma_1 \geq \sigma_2$).
3. $\mathbf{b} := \mathbf{Uy}$: $\mathbf{U}$ è una matrice ortogonale e quindi, come per $\mathbf{V}$, effettua solo una rotazione di $\mathbf{y}$.

I precedenti tre passi sono visualizzati nella figura 4.

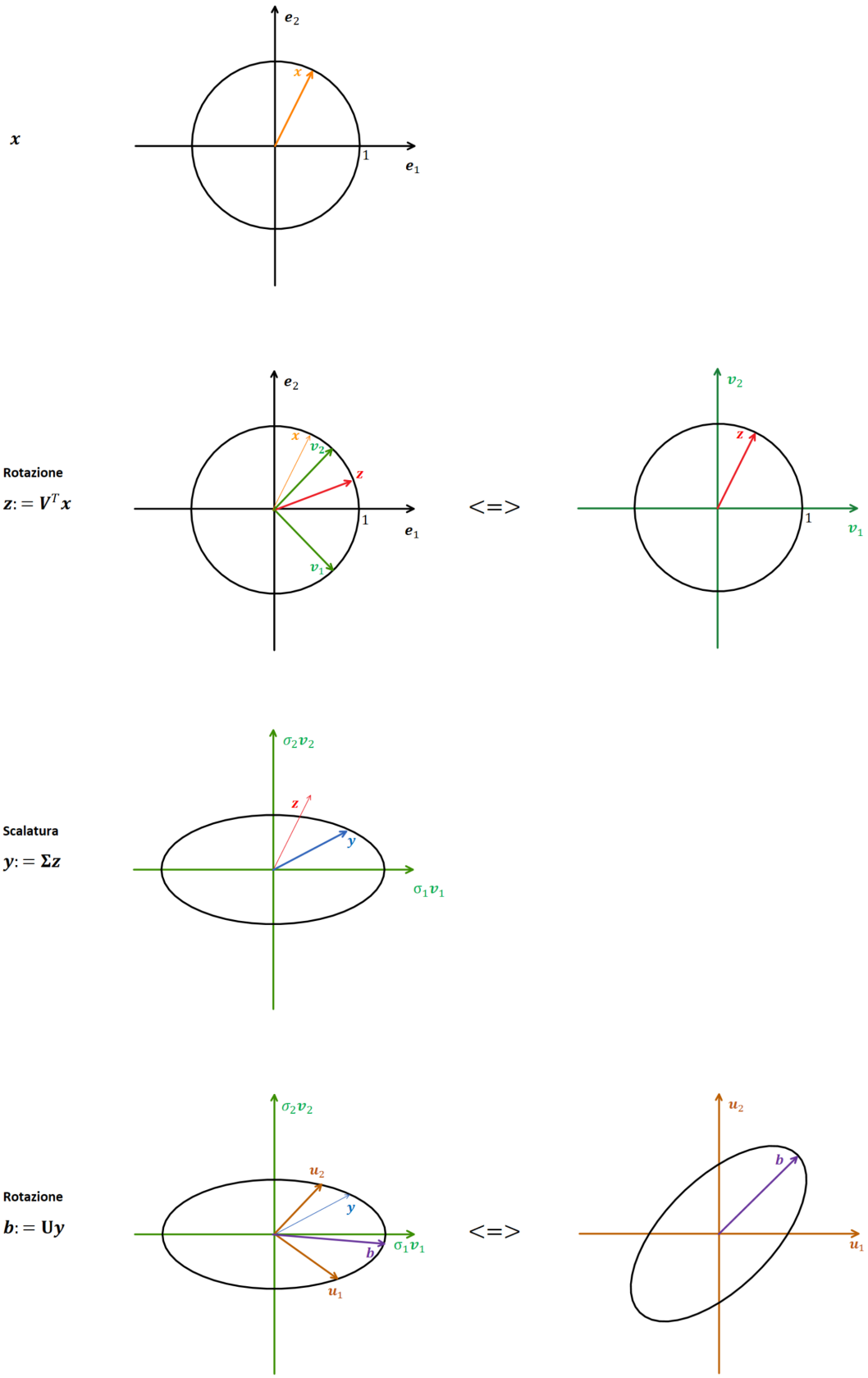

Figure 4: Visualizzazione geometrica dell'applicazione delle tre matrici $\mathbf{U}$, $\mathbf{\Sigma}$ e $\mathbf{V}$.

2 Applicazioni della Decomposizione a Valori Singolari

2.1 Compressione di immagini

Un'immagine di dimensioni $n \times m$ può essere rappresentata da una o più matrici di dimensioni $n \times m$ in cui ogni cella (i, j) memorizza l'intensità di colore del pixel (i, j) nell'immagine. Il valore che può assumere ogni cella dipende dal modello di colore che si utilizza, il quale è una rappresentazione dei colori in forma numerica con tipicamente 3 o 4 valori o componenti cromatiche. Il modello di colore più diffuso è il *modello RGB* che tiene in considerazione l'intensità del rosso, del verde e del blu dei pixel utilizzando un numero intero compreso tra 0 (assenza del colore) e 255 (intensità massima del colore) o, normalizzando, valori tra 0 e 1. Questi tre colori, miscelati tra loro, coprono quasi tutto lo spazio dei colori visibile.
Quindi, mentre un'immagine in bianco e nero può essere rappresentata da una sola matrice, un'immagine a colori, utilizzando il modello RGB, può essere rappresentata da tre matrici, una per il colore rosso, una per il colore verde ed una per il colore blu, e ogni cella (i, j) delle matrici memorizza l'intensità, rispettivamente, di rosso, di verde e di blu nel pixel (i, j).

Le immagini, a volte, richiedono un grande spazio di archiviazione. Una delle possibili soluzioni a questo problema è la compressione dei dati, il cui obiettivo principale è ridurre la quantità di dati utilizzati per rappresentare l'immagine. Il processo di compressione delle immagini può essere classificato in due categorie:

- *Compressione senza perdita di qualità*: vengono rimossi i metadati e le informazioni non essenziali che non influiscono sull'aspetto o sulla qualità dell'immagine.

- *Compressione con perdita di qualità*: questi metodi sfruttano l'incapacità del sistema visivo umano di rilevare piccoli dettagli e variazioni nelle immagini. L'eliminazione di questi dettagli riduce la quantità di spazio necessaria per memorizzare l'immagine.

La Decomposizione a Valori Singolari può essere utilizzata come metodo di compressione con perdita di qualità.

Si consideri un'immagine a colori grande $n \times m$. Come detto in precedenza, questa può essere rappresentata da tre matrici $\mathbf{R} \in \mathbb{R}^{n \times m}$, $\mathbf{G} \in \mathbb{R}^{n \times m}$ e $\mathbf{B} \in \mathbb{R}^{n \times m}$. La compressione può essere effettuata applicando la truncated SVD su queste tre matrici e considerando solo i primi r valori singolari, dove r è minore del rango delle tre matrici. La composizione di $\mathbf{R}$, $\mathbf{G}$ e $\mathbf{B}$ corrispenderà all'immagine compressa. Nel capitolo successivo ne viene mostrata un'implementazione in R.

2.2 Testo

I meccanismi di indicizzazione tradizionali per la ricerca di documenti sono costruiti a partire da informazioni quali titoli, elenchi di autori, abstract, elenchi di parole chiave, ecc. Prima dell'avvento dei moderni sistemi informatici, la ricerca delle informazioni poteva essere effettuata solo manualmente. Questi metodi manuali di indicizzazione stanno soccombendo di fronte a problemi di scala e coerenza, che si aggravano maggiormente nel dominio digitale, rendendo quindi necessario l'utilizzo di tecniche automatizzate. Tuttavia, nonostante i sistemi automatizzati siano la risposta ad alcune preoccupazioni della gestione delle informazioni, hanno i loro problemi provocati dalla polisemia (parole che hanno molteplici significati) e dalla sinonimia (più parole che hanno lo stesso significato).
Uno dei modelli più utilizzati in algebra lineare in questo dominio, è il *Vector Space Model*, nel quale ogni documento di una raccolta viene rappresentato con un vettore. Ogni componente del vettore si riferisce ad un particolare concetto, parola chiave o termine associato al documento in questione. Il valore assegnato a tale componente riflette l'importanza del termine nel documento. Dato un insieme di termini, chiamato Dizionario, una raccolta di documenti contenente un totale di d documenti descritti da t termini, viene rappresentata dalla *matrice termine-documento* $\mathbf{A} \in \mathbb{R}^{t \times d}$. Quindi, i vettori che rappresentano i documenti costituiscono le colonne della matrice. Pertanto, l'elemento a_{ij} della matrice $\mathbf{A}$ è l'importanza del termine i

nel documento j. Quindi, poiché un documento utilizza generalmente solo un piccolo sottoinsieme dell'intero dizionario di termini generato per un determinato database, la maggior parte degli elementi di una matrice termine-documento è pari a zero.
Nello schema del vector space model, l'utente interroga il database con una *query* per trovare i documenti rilevanti. La query è un insieme di termini rappresentato proprio come un documento. Anche in questo caso, è probabile che molti dei termini presenti nel dizionario non compaiano nella query, il che significa che molte delle componenti del vettore della query sono nulle. Il *Query matching* consiste nel trovare i documenti più simili alla query data. In altre parole, i documenti selezionati saranno quelli geometricamente più vicini alla query secondo una certa misura, come la similarità di coseno. Se la matrice termine-documento $\mathbf{A}$ ha colonne $\mathbf{a}_j$ per $j = 1, ..., d$, data una query $\mathbf{q}$, si calcolano d similarità di coseno (una per ogni documento) utilizzando la formula

$$cos\theta_j = \frac{\mathbf{a}_j^T \mathbf{q}}{\|\mathbf{a}_j\|_2 \|\mathbf{q}\|_2}$$

Vengono quindi recuperati i primi k (scelto dall'utente) documenti con la similarità più alta, oppure i documenti con una similarità maggiore di una certa soglia (definita dall'utente).
Per risolvere i problemi relativi alla sinonimia, alla polisemia e alle grandi dimensioni della matrice termine-documento, si può utilizzare la *Latent Semantic Indexing* (LSI), ossia una tecnica che proietta le query e i documenti in uno spazio con dimensioni semantiche "latenti". In questo nuovo spazio, una query e un documento possono avere un'elevata similarità di coseno anche se non condividono alcun termine, purché i loro termini siano semanticamente simili. Lo spazio semantico latente, in cui vengono proiettate le query e i documenti, ha meno dimensioni dello spazio originale. In particolare, la LSI è un metodo di riduzione della dimensionalità che applica la truncated SVD alla matrice termine-documento. Utilizzando quindi un'approssimazione low-rank $\mathbf{A}_r$ di $\mathbf{A}$ si ottiene:

$$cos\theta_j = \frac{(\mathbf{A}_r \mathbf{e}_j)^T \mathbf{q}}{\|\mathbf{A}_r \mathbf{e}_j\|_2 \|\mathbf{q}\|_2} =$$

dove $\mathbf{e}_j$ è il j-esimo vettore della base canonica

$$= \frac{(\mathbf{U}_{t\times r} \mathbf{\Sigma}_r \mathbf{V}_{r\times d}^T \mathbf{e}_j)^T \mathbf{q}}{\|\mathbf{U}_{t\times r} \mathbf{\Sigma}_r \mathbf{V}_{r\times d}^T \mathbf{e}_j\|_2 \|\mathbf{q}\|_2} =$$

$$= \frac{\mathbf{e}_j^T \mathbf{V}_{d\times r} \mathbf{\Sigma}_r (\mathbf{U}_{r\times t}^T \mathbf{q})}{\|\mathbf{\Sigma}_r \mathbf{V}_{r\times d}^T \mathbf{e}_j\|_2 \|\mathbf{q}\|_2} =$$

semplificando, sia $\mathbf{s}_j := \mathbf{\Sigma}_r \mathbf{V}_{r\times d}^T \mathbf{e}_j$

$$= \frac{\mathbf{s}_j^T (\mathbf{U}_{r\times t}^T \mathbf{q})}{\|\mathbf{s}_j\|_2 \|\mathbf{q}\|_2}$$

2.3 Pseudoinversa

La Decomposizione a Valori Singolari consente di trovare facilmente la pseudoinversa di una matrice, sfruttando le caratteristiche delle matrici $\mathbf{U}$, $\mathbf{\Sigma}$ e $\mathbf{V}$. Infatti, per una matrice ortogonale, l'inversa coincide con la trasposta, mentre per una matrice diagonale, la sua pseudoinversa sarà la matrice diagonale con i reciproci degli elementi diagonali.
Quindi, sia $\mathbf{A} \in \mathbb{R}^{n\times m}$, con $k = rank(\mathbf{A})$, la sua pseudoinversa $\mathbf{A}^+$ può essere calcolata con

$$\mathbf{A}^+ = \mathbf{V}\mathbf{\Sigma}^+\mathbf{U}^T$$

con $\mathbf{\Sigma}^+ = diag(\frac{1}{\sigma_1}, ..., \frac{1}{\sigma_k}, 0, ..., 0)$.

2.4 Problema dei Minimi Quadrati

Siano $\mathbf{A} \in \mathbb{R}^{n \times m}$ e $\mathbf{b} \in \mathbb{R}^n$ con $n \geq m$, e $k = rank(\mathbf{A}) = m$, ossia con rango massimo. Il Problema dei Minimi Quadrati consiste nel determinare $\mathbf{x}^* \in \mathbb{R}^m$, *t.c.*:

$$\|\mathbf{b} - \mathbf{A}\mathbf{x}^*\|_2^2 = \min_{\mathbf{x} \in \mathbb{R}^m} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2$$

La Decomposizione a Valori Singolari è utile per risolvere questo tipo di problema. Infatti, applicando la SVD su $\mathbf{A}$:

$$\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2 = \|\mathbf{b} - \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\mathbf{x}\|_2^2 =$$

$$= \|\mathbf{U}(\mathbf{U}^T\mathbf{b} - \mathbf{\Sigma}\mathbf{V}^T\mathbf{x})\|_2^2 =$$

$\mathbf{U}$ è ortogonale, quindi $\|\mathbf{U}\bullet\| = \|\bullet\|$

$$= \|\mathbf{U}^T\mathbf{b} - \mathbf{\Sigma}\mathbf{V}^T\mathbf{x}\|_2^2 =$$

Sia $\mathbf{y} := \mathbf{V}^T\mathbf{x}$

$$= \left\| \begin{bmatrix} \mathbf{U}_{k \times n}^T \mathbf{b} \\ \mathbf{U}_{(n-k) \times n}^T \mathbf{b} \end{bmatrix} - \begin{bmatrix} \mathbf{\Sigma}_k \\ 0 \end{bmatrix} \mathbf{y} \right\|_2^2 =$$

$$= \|\mathbf{U}_{k \times n}^T \mathbf{b} - \mathbf{\Sigma}_k \mathbf{y}\|_2^2 + \|\mathbf{U}_{(n-k) \times n}^T \mathbf{b}\|_2^2$$

Poiché l'unica parte che dipende da $\mathbf{x}$ è $\|\mathbf{U}_{k \times n}^T \mathbf{b} - \mathbf{\Sigma}_k \mathbf{y}\|_2^2$, solo quest'ultima può essere minimizzata, ponendola $= 0$. Ossia:

$$\|\mathbf{U}_{k \times n}^T \mathbf{b} - \mathbf{\Sigma}_k \mathbf{y}\|_2^2 = 0$$

$$\iff$$

$$\mathbf{U}_{k \times n}^T \mathbf{b} - \mathbf{\Sigma}_k \mathbf{y} = 0$$

$$\iff$$

$$\mathbf{y} = \mathbf{\Sigma}_k^{-1} \mathbf{U}_{k \times n}^T \mathbf{b}$$

Siccome $\mathbf{y} = \mathbf{V}^T\mathbf{x}$, la soluzione al problema è $\mathbf{x}^* = \mathbf{V}\mathbf{y} = \mathbf{V}\mathbf{\Sigma}_k^{-1}\mathbf{U}_{k \times n}^T \mathbf{b}$.

2.5 PCA

L'*Analisi delle Componenti Principali*, nota anche come PCA, è una tecnica comunemente utilizzata per ridurre la dimensionalità dei dati preservando il più possibile le informazioni contenute nei dati originali. La PCA raggiunge questo obiettivo proiettando i dati su un sottospazio di dimensioni inferiori che conserva la maggior parte della varianza tra i dati. In particolare, la PCA crea delle nuove variabili, non correlate tra loro, che sono combinazioni lineari di quelle originali. La prima variabile, ossia la prima componente principale, spiegherà la massima varianza, la seconda variabile spiegherà meno varianza della prima, la terza spiegherà meno varianza della seconda e così via. In questo modo, è possibile ridurre la dimensionalità dei dati considerando solo le prime variabili.
I passi necessari per calcolare le componenti principali sono:

1. Standardizzare i dati: Le variabili che compongono il set di dati sono spesso su scale e medie diverse. Questo può causare problemi, come la produzione di numeri estremamente grandi durante il calcolo. Per rendere il processo più efficiente, è buona norma centrare i dati sulla media zero e renderli privi di unità di misura.

2. Calcolare la matrice di covarianza: la PCA cerca di catturare la maggior parte delle informazioni di un insieme di dati identificando le componenti principali che massimizzano la varianza tra le osservazioni. La matrice di covarianza è una matrice simmetrica con righe e colonne pari al numero di dimensioni dei dati avente, nella cella (i, j) la covarianza tra la variabile i e la variabile j.

3. Calcolare gli autovettori e gli autovalori della matrice di covarianza: gli autovettori della matrice di covarianza puntano nella direzione della varianza maggiore. Più grande è l'autovalore, più varianza è spiegata. In altre parole, l'autovettore con l'autovalore più grande corrisponde alla prima componente principale, che spiega la maggior parte della varianza, l'autovettore con il secondo autovalore più grande corrisponde alla seconda componente principale, ecc.

4. Proiettare i dati nel nuovo spazio: sia $\mathbf{A}$ la matrice con gli autovettori (mutuamente ortonormali) della matrice di covarianza sulle colonne e $\mathbf{X}$ la matrice dei dati originali. La matrice contenente i dati proiettati nel nuovo spazio sarà $\mathbf{Y} = \mathbf{XA}$.

C'è un legame tra la PCA e la SVD. Infatti, sia $\mathbf{X} \in \mathbb{R}^{n \times m}$ la matrice dei dati centrata, la sua matrice di covarianza è data da

$$\mathbf{C}_X = \frac{1}{n-1}\mathbf{X}^T\mathbf{X} =$$

Applicando la SVD su $\mathbf{X}$

$$= \frac{1}{n-1}\mathbf{V}\mathbf{\Sigma}\mathbf{U}^T\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T =$$

$$\mathbf{V}\frac{\mathbf{\Sigma}^2}{n-1}\mathbf{V}$$

Da questa relazione risulta che $\mathbf{C_X}$ è simile a $\frac{\mathbf{\Sigma}^2}{n-1}$ e quindi, i suoi autovettori, ossia le componenti principali, saranno le colonne di $\mathbf{V}$, mentre gli autovalori saranno uguali a $\lambda_i = \frac{\sigma_i^2}{n-1}$

3 SVD in R

La Decomposizione a Valori Singolari può essere facilmente calcolata in R utilizzando la funzione svd() o La.svd(). Entrambe le funzioni utilizzano LAPACK (Linear Algebra PACKage), ossia una libreria, scritta in Fortran 90, utile per la risoluzione di problemi di algebra lineare, come la risoluzione di sistemi di equazioni lineari, risoluzione ai minimi quadrati di sistemi di equazioni lineari e decomposizioni come SVD, LU, QR, Cholesky e Schur. La Full SVD, la Compact SVD e la Truncated SVD possono essere calcolate utilizzando svd() o La.svd() e scegliendo accuratamente i parametri.

3.1 Funzione svd()

La funzione svd() offerta da R si presenta con la seguente intestazione:
svd(x, nu = min(n, p), nv = min(n, p), LINPACK = FALSE)
Dove:

- x: una matrice numerica o complessa di cui si vuole calcolare la decomposizione SVD. Le matrici logiche vengono forzate a numeriche;
- nu: numero di vettori singolari sinistri da calcolare. Deve essere compreso tra 0 e il numero di righe di della matrice. Di default, il valore è uguale al minimo tra il numero delle righe e il numero delle colonne;
- nv: numero di vettori singolari destri da calcolare. Deve essere compreso tra 0 e il numero di colonne di della matrice. Di default, il valore è uguale al minimo tra il numero delle righe e il numero delle colonne.
- LINPACK: valore booleano utilizzato per risolvere alcuni problemi di compatibilità con vecchie versioni di R.

Questa funzione ritorna una lista contenente:

- $d: vettore contente i valori singolari della matrice x in ordine decrescente;
- $u: matrice le cui colonne sono i vettori singolari sinistri della matrice x;
- $v: matrice le cui colonne sono i vettori singolari destri della matrice x.

3.2 Funzione La.svd()

La funzione La.svd() si presenta con la seguente intestazione:
svd(x, nu = min(n, p), nv = min(n, p))

Dove i parametri x, nu e nv sono gli stessi di svd(). L'unica differenza tra le due funzioni è che La.svd() sostituisce $v con $vt, ossia la matrice le cui righe sono i vettori singolari destri.

3.3 Implementazione della SVD, Compact SVD e Truncated SVD

Di seguito sono riportate le implementazioni della Full SVD, della Compact SVD e della Truncated SVD.

La libreria *Matrix* è necessaria per il calcolo del rango della matrice **A**.

```
library(Matrix)
set.seed(5)
A <- matrix(sample(-100:100,
                   21,
                   replace=TRUE),
            ncol=7)
print(A)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
## [1,]  -35   20   46  -43  -25   27  -27
## [2,]    6  -60   30   74  -85  -74  -16
## [3,]   84  -30  -63  -48   53  100   21
```

```
full_svd <- function(A) {
  k = rankMatrix(A)
  #Inizializzazione di Sigma
  Sigma <- matrix(0, nrow(A), ncol(A))

  ATA <- t(A) %*% A
  AAT <- A %*% t(A)

  #Calcolo di V
  eigenATA <- eigen(ATA)
  #Ogni colonna della matrice degli autovettori corrisponde ad un autovettore
  V <- eigenATA$vectors

  #Calcolo di U
  eigenAAT <- eigen(AAT)
  U <- eigenAAT$vectors

  #Calcolo di Sigma
  #Gli autovalori ritornati sono in ordine decrescente.
  eigenValAAT <- eigenAAT$values[0:k]
  sigmas <- sqrt(eigenValAAT)
  SigmaK <- sigmas * diag(length(sigmas))
  Sigma[0:nrow(SigmaK), 0:ncol(SigmaK)] <- SigmaK

  return <- list(U=U , Sigma=Sigma, V=V)
}

compact_svd <- function(A) {
  k = rankMatrix(A)

  ATA <- t(A) %*% A

  #Calcolo di Vk
  eigenATA <- eigen(ATA)
  Vk <- eigenATA$vectors[, 0:k]

  #Calcolo di Sigmak
  eigenValATA <- eigenATA$values[0:k]
  sigmas <- sqrt(eigenValATA)
  Sigmak <- sigmas * diag(length(sigmas))
```

```
  #Calcolo Uk
  Uk <- A %*% Vk %*% solve(Sigmak)

  return <- list(U=Uk , Sigma=Sigmak, V=Vk)
}

truncated_svd <- function(A, r) {
  ATA <- t(A) %*% A

  #Calcolo di Vr
  eigenATA <- eigen(ATA)
  Vr <- eigenATA$vectors[, 0:r]


  #Calcolo di Sigma
  #Si considerano i primi r autovalori
  eigenValATA <- eigenATA$values[0:r]
  sigmas <- sqrt(eigenValATA)
  Sigmar <- sigmas * diag(length(sigmas))

  #Calcolo di Ur
  Ur <- A %*% Vr %*% solve(Sigmar)

  return <- list(U=Ur , Sigma=Sigmar, V=Vr)
}
```

Per la Truncated SVD è stato scelto un numero di valori singolari pari al $rank(\mathbf{A}) - 1$.

```
SVD <- full_svd(A)
A_SVD <- SVD$U %*% SVD$Sigma %*% t(SVD$V)

compactSVD <- compact_svd(A)
A_compactSVD <- compactSVD$U %*% compactSVD$Sigma %*% t(compactSVD$V)

truncatedSVD <- truncated_svd(A, rankMatrix(A)-1)
A_truncatedSVD <- truncatedSVD$U %*% truncatedSVD$Sigma %*% t(truncatedSVD$V)


all.equal(A, A_SVD)
```

```
## [1] TRUE
```

```
all.equal(A, A_compactSVD)
```

```
## [1] TRUE
```

L'errore in norma 2 che si commette considerando **A_truncatedSVD** al posto di **A**, può essere calcolato con:

```
err_norm2 = norm(A-A_truncatedSVD, type="2")**2
err_norm2
```

```
## [1] 4207.704
```

3.3.1 Applicazione della SVD sulle immagini in R

Di seguito è riportata l'applicazione della Decomposizione a Valori Singolari per la compressione di una foto in R. Viene utilizzato il package *jpeg*, il quale fornisce funzioni utili come la lettura e la scrittura di file .jpeg.

```
library(jpeg)
foto <- readJPEG('images/foto.jpg')
```

readJPEG legge un'immagine jpeg e la trasforma in un tensore $altezza \times larghezza \times canale$. Ogni cella rappresenterà un pixel e memorizzerà un valore compreso tra 0 e 1, il quale rappresenterà l'intensità di rosso, verde o blu nel pixel.

```
r <- foto[ , , 1]
g <- foto[ , , 2]
b <- foto[ , , 3]

r.rank <- rankMatrix(r)
g.rank <- rankMatrix(g)
b.rank <- rankMatrix(b)

sprintf("Rango della matrice relativa al canale rosso: %i", r.rank)
```

```
## [1] "Rango della matrice relativa al canale rosso: 2205"
```

```
sprintf("Rango della matrice relativa al canale verde: %i", g.rank)
```

```
## [1] "Rango della matrice relativa al canale verde: 2205"
```

```
sprintf("Rango della matrice relativa al canale blu: %i", b.rank)
```

```
## [1] "Rango della matrice relativa al canale blu: 2205"
```

```
foto.r.svd <- svd(r)
foto.g.svd <- svd(g)
foto.b.svd <- svd(b)

rgb.svds <- list(foto.r.svd, foto.g.svd, foto.b.svd)
```

Viene applicata la SVD sulle tre matrici che rappresentano i tre canali di colore della foto.

```
num_sigma <- c(3, 5, 10, 25, 35, 70, 100)
for (r in num_sigma) {
  foto_compressa <- sapply(rgb.svds, function(i) {
    foto_troncata <- i$u[,1:r] %*% diag(i$d[1:r]) %*% t(i$v[,1:r])
```

```
  }, simplify = 'array')
  writeJPEG(foto_compressa, paste('images/foto_compressa/foto_compressa_con_',
                                  r, '_sigma.jpg', sep=''))
}
```

La foto è stata compressa 7 volte, variando il numero di valori singolari che si tengono in considerazione nella Truncated SVD. Di seguito vengono riportate le foto compresse. Come si può notare dalla figura 12, utilizzando più o meno 100 valori singolari, la foto compressa è molto simile a quella originale e ha una dimensione molto inferiore, passando 1078KB a 279KB.

Figure 5: Foto Originale. Dimensione: 1077.65 KB

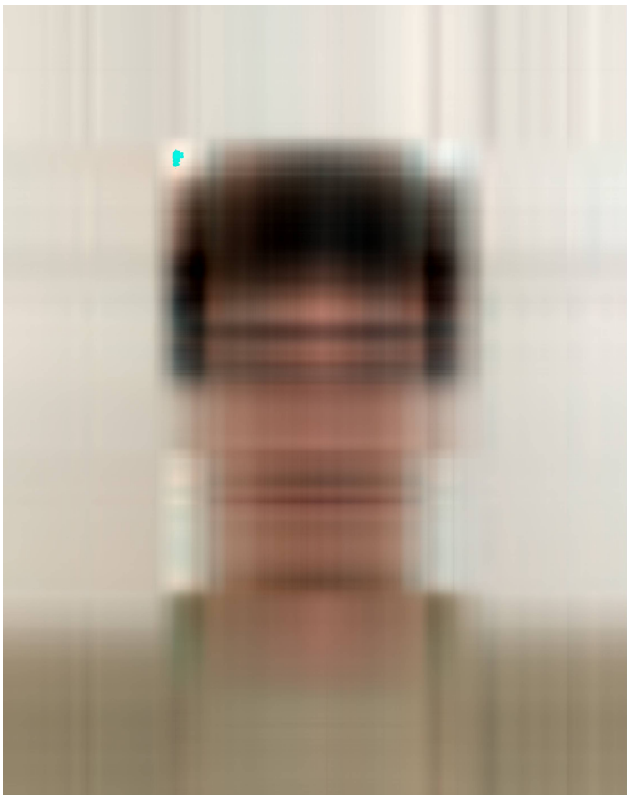

Figure 6: Foto compressa utilizzando 3 Valori Singolari. Dimensione: 143.86 KB

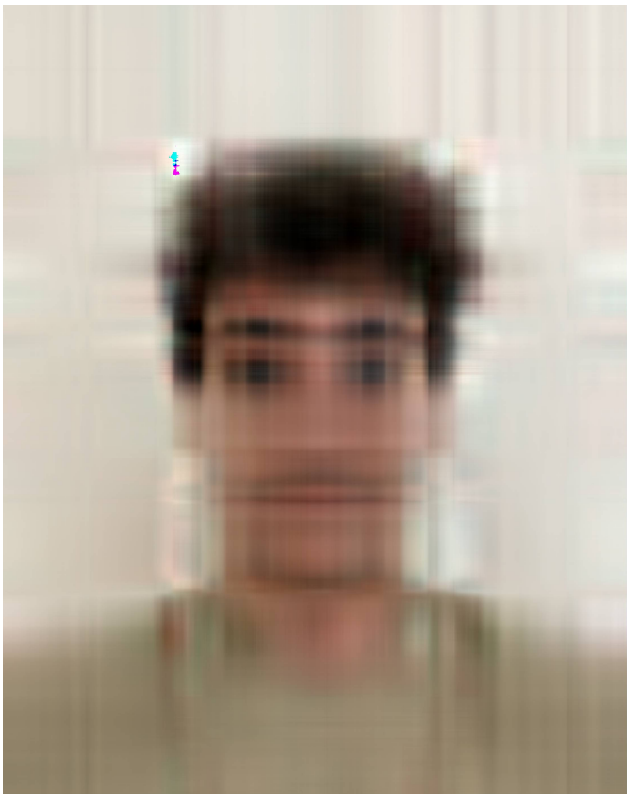

Figure 7: Foto compressa utilizzando 5 Valori Singolari. Dimensione: 154.99 KB

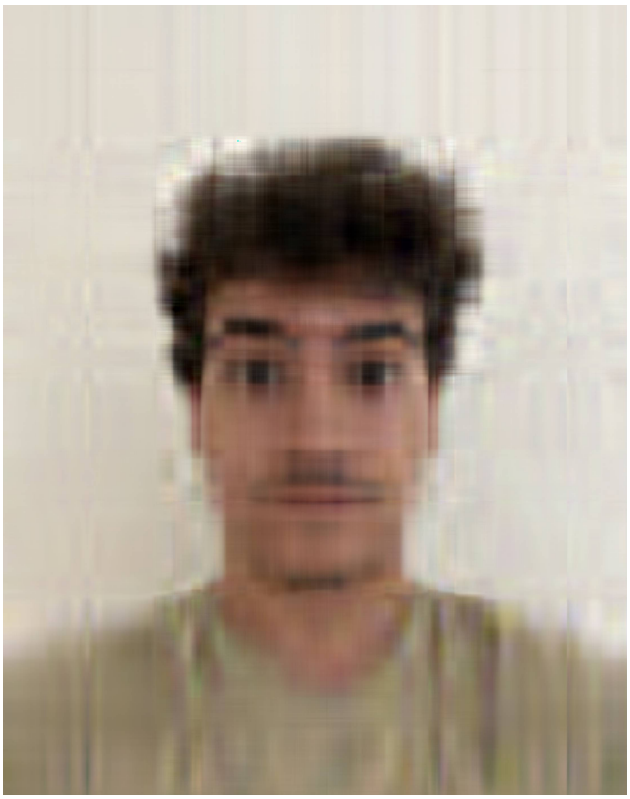

Figure 8: Foto compressa utilizzando 10 Valori Singolari. Dimensione: 171.47 KB

Figure 9: Foto compressa utilizzando 25 Valori Singolari. Dimensione: 204.10 KB

Figure 10: Foto compressa utilizzando 35 Valori Singolari. Dimensione: 218.04 KB

Figure 11: Foto compressa utilizzando 70 Valori Singolari. Dimensione: 255.56 KB

Figure 12: Foto compressa utilizzando 100 Valori Singolari. Dimensione: 278.50 KB