Applicazione della tsvd all'elaborazione di immagini

A cura di:
Mauro Franceschelli
Simone Secchi

Indice pag

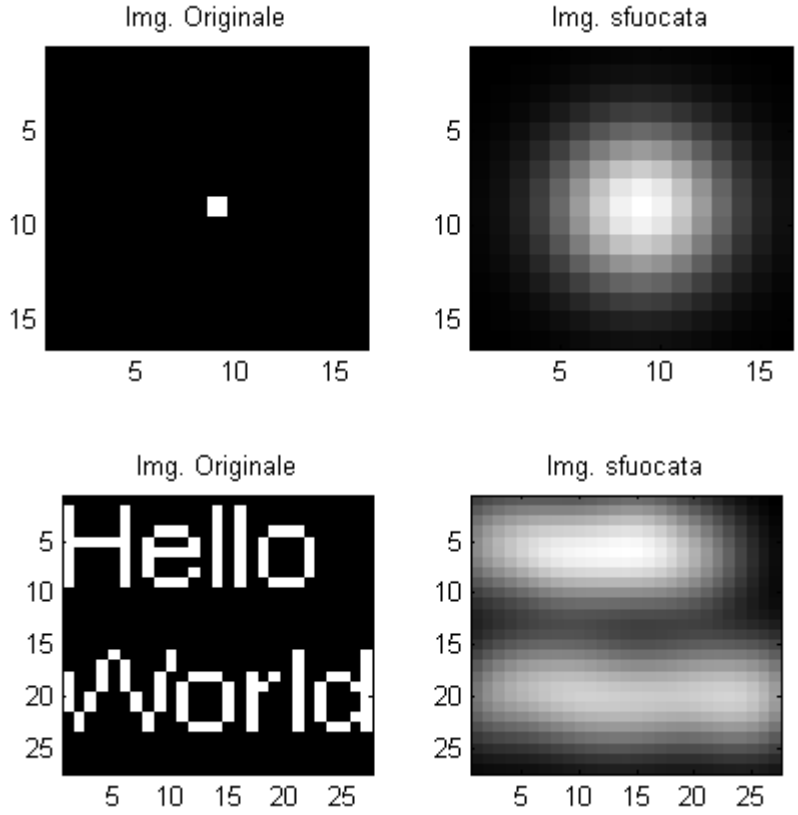
Introduzione

Scopo di questo lavoro è la sperimentazione dell' uso della fattorizzazione SVD nella risoluzione di sistemi lineari fortemente malcondizionati utilizzando il criterio della curva-L. In particolare si è fatto riferimento alla risoluzione di problemi inversi nell'elaborazione digitale delle immagini mostrando diversi risultati ottenuti con immagini campione di piccole dimensioni e il loro paragone con i risultati ottenibili utilizzando l'algoritmo di Gauss. Le immagini campione scelte sono: il punto su sfondo nero e una scritta bianco su nero .

Problema diretto

Per la risoluzione del problema inverso tramite la TSVD occorre prima disporre di un immagine contenente sfuocamento e rumore. Non avendo a disposizione una tale immagine ne è stata creata una ad hoc risolvendo il problema diretto. Come legge di sfuocamento è stata creata una maschera di convoluzione contenente una gaussiana con media nulla e varianza determinata in ingresso come parametro simulativo. Chiaramente per utilizzare il metodo della TSVD è stato necessario trasformare la relazione integrale in un sistema lineare mediante discretizzazione. Il sistema lineare risultante consiste in una matrice di toeplitz a blocchi di Toeplitz, è stato quindi creato il primo blocchetto utilizzando la funzione Matlab "toeplitz.m" ed ottenuta la matrice desiderata utilizzando il prodotto di Kronecker della matrice con se stessa e normalizzando all' altezza della gaussiana. L' immagine è stata trasformata in un vettore colonna seguendo l'ordinamento lessicografico e si è dunque ottenuta l'immagine sfuocata mostrata a fine paragrafo. A questo punto è possibile verificare come il sistema lineare sia estremamente mal condizionato al crescere della varianza della gaussiana, ad esempio con una varianza *sigma* di 3 si è misurato un condizionamento in norma 2 di 10^19. Ad ogni modo l' errore sui dati è molto piccolo essendo circa l' epsilon di macchina pari a 2.2204e-016. Si è dunque aggiunto un rumore gaussiano bianco con varianza pari a 1 e ampiezza massima N, in modo che il sistema restituisca un pessimo risultato se risolto con le normali tecniche, ad esempio con l'algoritmo di Gauss (un esempio dell' immagine risolta con Gauss è mostrato a fine testo).

Sotto sono mostrate le immagini originali e quelle dopo lo sfuocamento:



Problema inverso

E' stato introdotto in precedenza come il problema inverso di determinazione dell'immagine originale (riordinata in un vettore) a partire dall'immagine sfuocata e corrotta da rumore (di tipo gaussiano bianco con varianza assegnata) e dalla conoscenza del kernel di sfuocamento sia un problema fortemente malcondizionato. L'inversione non può quindi essere effettuata con un semplice algoritmo di risoluzione del sistema lineare $A \cdot x = b$ (per esempio Gauss) per via del forte condizionamento della matrice A, che porterebbe ad una soluzione inaffidabile per la presenza del rumore: vale infatti

$$\frac{\|\Delta x\|}{\|x\|} = \|A^{-1}\| \cdot \|A\| \cdot \frac{\|\Delta b\|}{\|b\|};$$

si noti inoltre che nel nostro caso, per determinati valori della varianza della curva gaussiana di sfuocamento, il numero di condizionamento della matrice A raggiunge valori di 10^19, motivo per cui la sola presenza del noise di memorizzazione sul calcolatore (che vale circa 2.2 10^-16) è sufficiente per rendere elevato l'errore relativo e di conseguenza del tutto inattendibile la soluzione.
L'algoritmo di inversione basato sulla SVD (Singular Values Decomposition) che è stato implementato costituisce una possibile soluzione del problema: nello script che è stato realizzato si è scelto di operare prima una decomposizione SVD completa (considerando quindi tutti i valori singolari della matrice A) e si è calcolata la relativa soluzione (vettoriale) xx:

```
[U,S,V]=svd(full(A));

C=U'*b;
S=diag(S);
y=C./S;
xx=V*y;
```

In seguito è stata calcolata la soluzione del sistema lineare con algoritmo di Gauss:

```
y2=A\b;
```

Si noti che, per come viene calcolata la matrice caratteristica del problema in esame (sfuocamento con Point Spread Function di tipo gaussiano) il sistema lineare è quadrato. Conoscendo la soluzione vera del problema `x` (l'immagine originale vettorializzata) sono stati confrontati gli errori delle due soluzioni così calcolate rispetto alla soluzione originale; la metrica scelta è la norma2 vettoriale, che quindi coincide con la norma di Frobenius per le matrici-immagine che di fatto compongono i vettori-soluzione. Sebbene si sia scelto di utilizzare la norma2, è bene osservare che questa non è la metrica "migliore" rispetto alla visualizzazione umana, infatti è possibile che una immagine soluzione con errore in norma2 maggiore di un'altra immagine soluzione sia "visivamente" più simile alla soluzione vera di quanto non sia la seconda immagine; a tal proposito esistono ricerche che cercano di individuare quale sia la metrica più adatta per la rappresentazione del funzionamento dell'HVS (Human Visual System).

```
disp('norma2 errore svd completa= ')
norm(xx-x)
disp('epsilon di macchina = ')
eps
disp('norma2 errore guass= ')
norm(y2-x)
```

Nel seguito dello script viene poi calcolato e mostrato a schermo il condizionamento della matrice A, che (per proprietà della scomposizione SVD) concide con il rapporto tra il primo valore singolare della matrice (il più grande) e l'ultimo (il più piccolo):

```
disp('condizionamento matrice A completa ')
```

```
S(1)/S(m^2)
```

Per via della presenza inevitabile dell'errore sui termini noti (vettore C nel codice) il calcolo della soluzione del problema con la SVD risulta instabile se si considerano tutti i valori singolari della matrice: infatti dividendo (punto-punto) i valori del vettore C dei termini noti per i valori singolari si ottiene un errore molto più sensibile sugli ultimi elementi, che vengono divisi per i valori singolari più piccoli. Per cercare di ridurre questo contributo di errore si ricorre alla TSVD (Truncated SVD): questo algoritmo prevede di considerare non tutti i valori singolari, ma solo i primi k (cioè quelli più grandi), mentre i restanti verranno posti a zero. Per stimare il valore di k ottimale sono state utilizzate due diverse tecniche:

- La prima fa uso della "curva-L", che ottiene il valore ottimale di k calcolando tutte le possibili TSVD (al variare di k) e considerando, per ognuna, la norma2 del vettore soluzione e la norma2 del vettore residuo (b-Ax); al variare di k si rappresenteranno in un grafico in scala logaritmica questi valori e si otterrà una curva con un andamento "ad L", il valore di k che individua l'angolo sarà quello ottimale, in quanto è quello per cui si ottiene residuo minimo e contemporaneamente soluzione non divergente.
- La seconda tecnica calcola tutte le possibili TSVD (al variare di k) e, per ognuna, calcola l'errore in norma2 della soluzione rispetto alla soluzione vera del problema (immagine originale): il valore ottimale di k sarà quello per cui si ottiene errore minimo.

Si noti che la prima tecnica è più generale della seconda perchè non necessità della conoscenza della soluzione esatta del problema; è comunque utile, per fini di sperimentazione dell'algoritmo, applicare anche la seconda e verificare il discostamento dei valori ottimali di k trovati. Il codice è:

```
X=tsvd(U,S,V,b,[1:m^2]); % tutte le possibili serie troncate in una matrice

    l=0;
for i=1:1:m^2-1
    l=l+1;
    err(i)=norm(x-X(:,i));
    condA2(l)=S(1)/S(i);
    rho(i)=norm(A*X(:,i)-b);
    eta(i)=norm(X(:,i));
end

[kcorner,info] = corner(rho, eta, 1);
close all
kcorner
kopt=find(err==min(err))
```

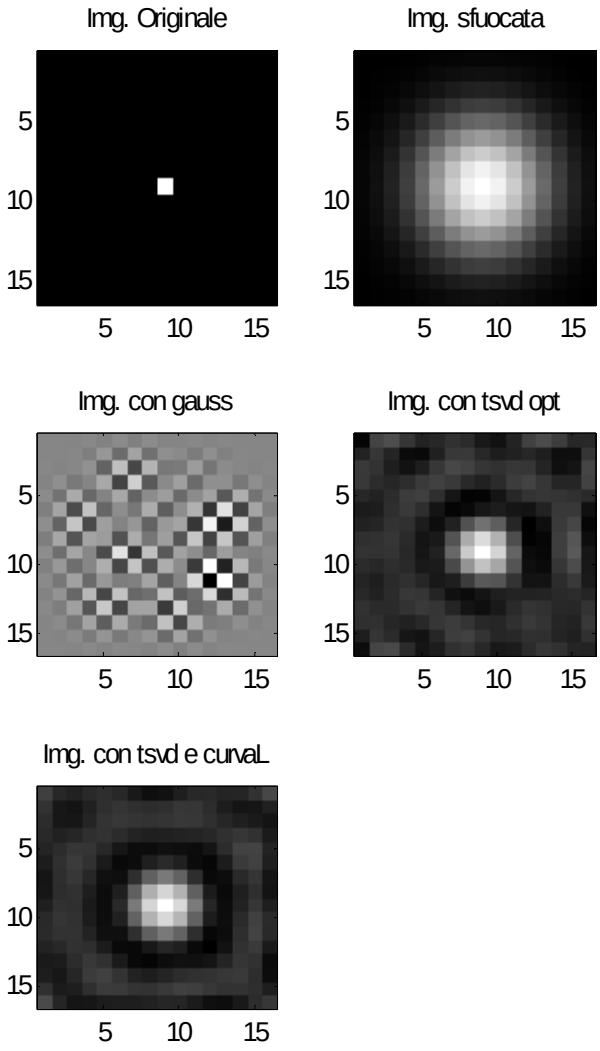
In conclusione dello script si ha un plot con 5 immagini:

1. immagine originale
2. immagine sfuocata
3. immagine ricostruita con Gauss
4. immagine ricostruita con TSVD ottimale (2° tecnica)
5. immagine ricostruita con TSVD e curva ad L (1° tecnica)

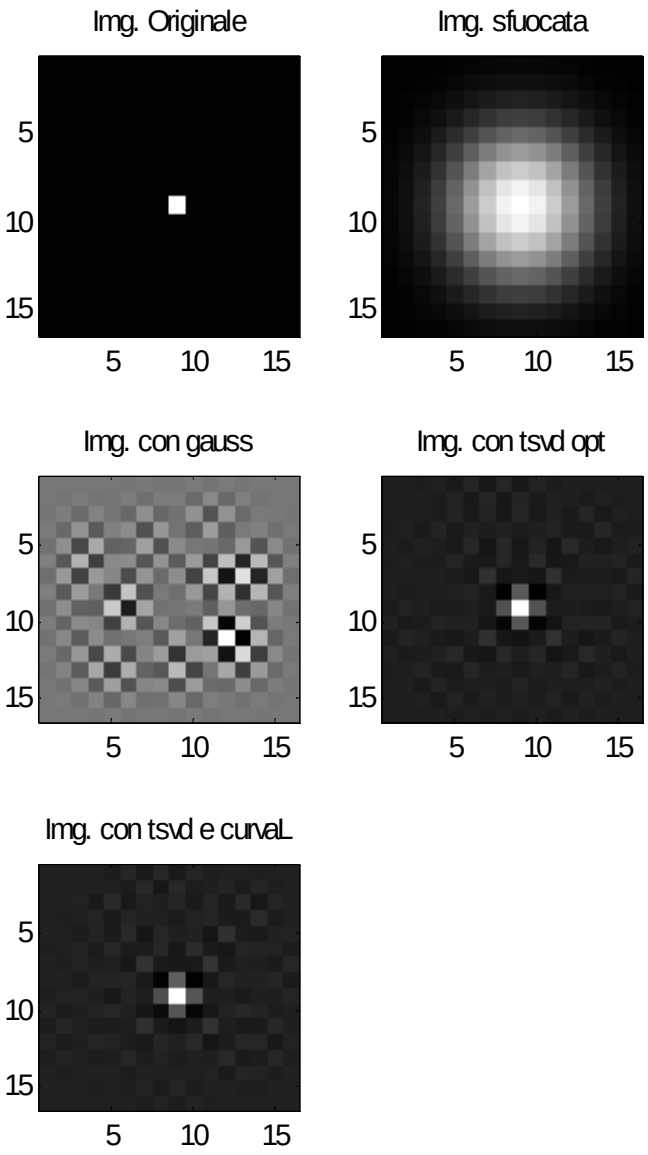
Simulazioni

Per verificare il funzionamento dell'algoritmo si è utilizzata come prima immagine un unico pixel bianco su sfondo completamente nero: questo problema può essere visto come il problema della determinazione della risposta impulsiva del sistema di sfuocamento in esame, infatti un'immagine come quella usata è paragonabile ad un segnale impulsivo in ingresso al sistema, l'immagine sfuocata è la risposta del sistema sotto esame e l'obiettivo è ricostruire l'immagine impulsiva a partire da quella sfuocata e corrotta dal noise. Le dimensioni delle immagini usate nei seguenti esempi sono ridotte (intorno ai 30*30 pixel) al fine di non dilatare troppo la durata della simulazione. I due parametri modificabili dello script sono la varianza `sigma` della curva gaussiana di sfuocamento e la ampiezza N del noise che si aggiunge ai dati misurati, ossia al vettore b dei termini noti. Si noti che il rumore scelto è gaussiano a media nulla ed è stato generato con la funzione Matlab `randn( ).` La varianza `sigma` è il parametro che principalmente influisce sul condizionamento del problema, ed è stato scelto un valore tale da portare il numero di condizionamento ad un valore di circa 10^19, quindi in grado di rendere del tutto inaffidabile la soluzione a causa dell'errore di memorizzazione del vettore `b` dei termini noti sul calcolatore (epsilon di macchina = 2.2 *10^-16). I risultati in questo caso sono, al variare dell'ampiezza N del noise:

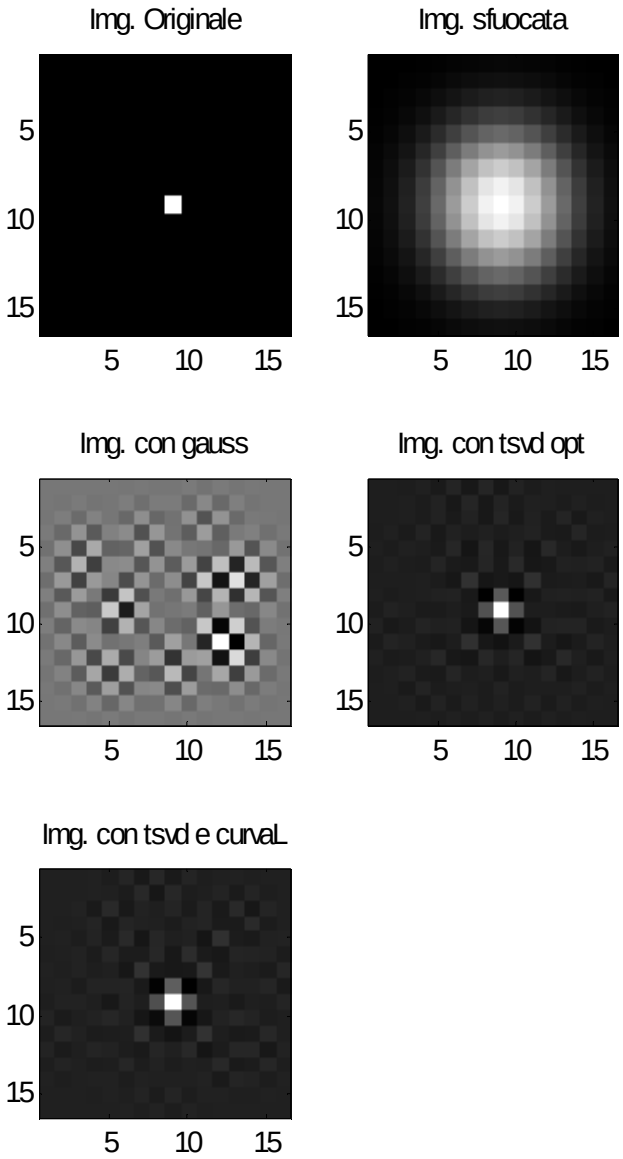
`N=10^-005`



N=10^-010

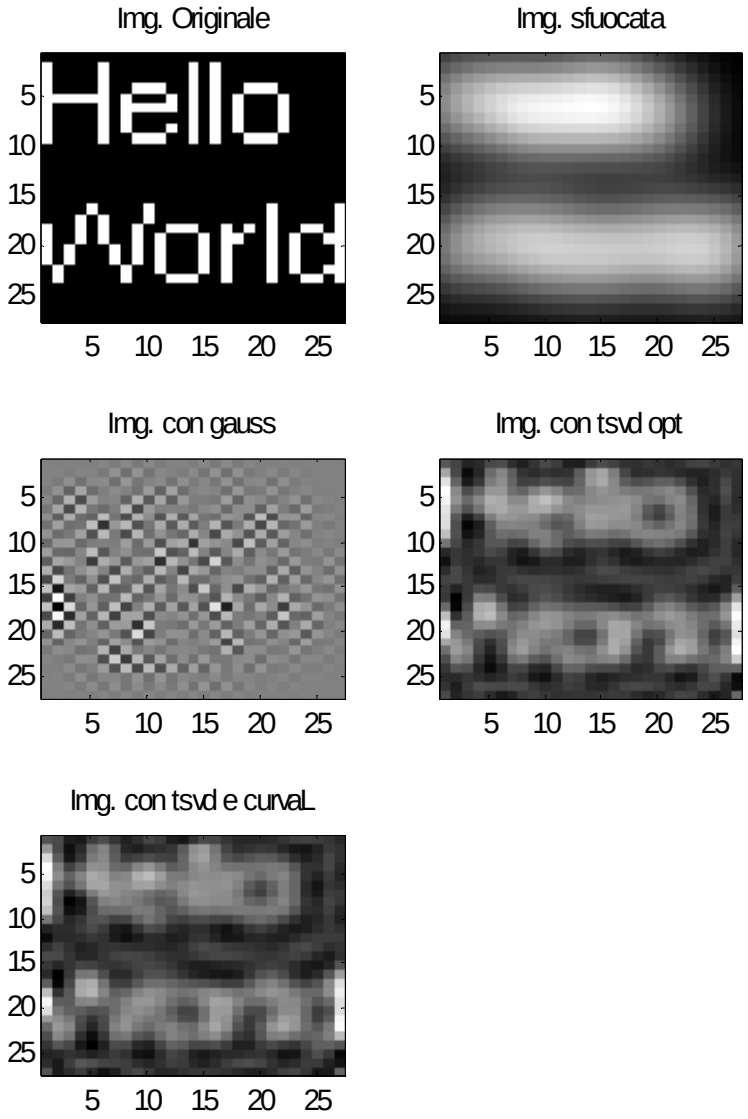

Img. Originale
Img. sfuocata
Img. con gauss
Img. con tsvd opt
Img. con tsvd e curvaL
5
10
15

N=10^-015

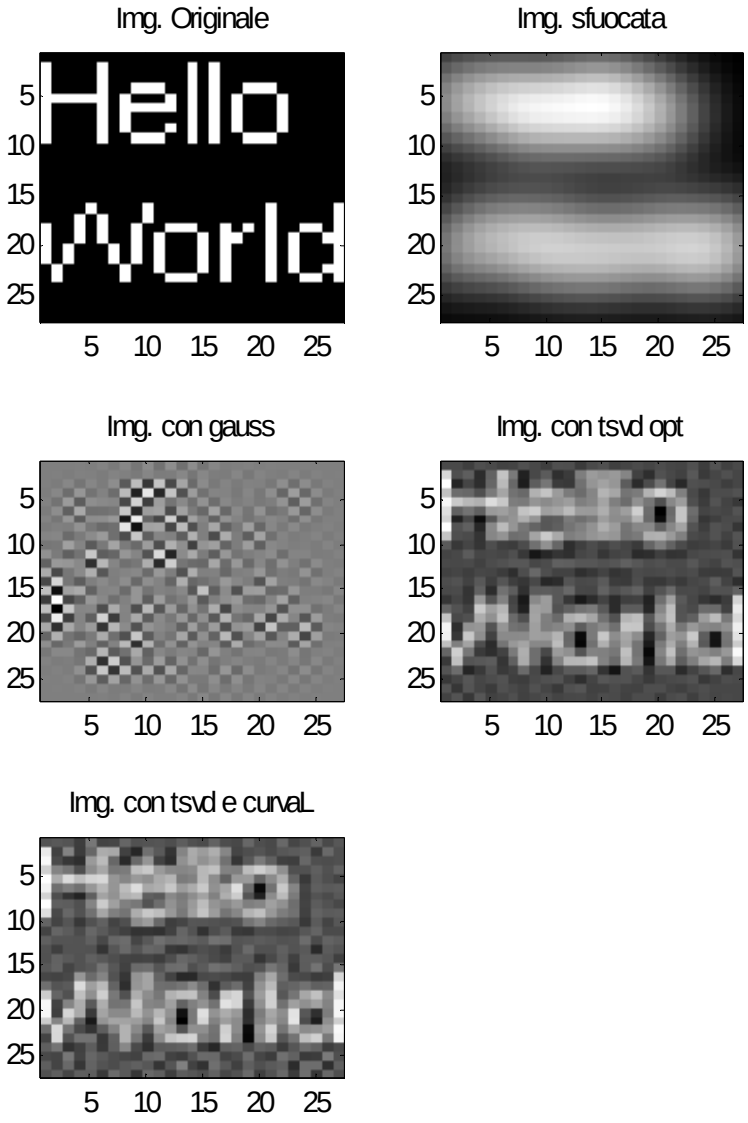

Img. Originale
Img. sfuocata
Img. con gauss
Img. con tsvd opt
Img. con tsvd e curvaL
5
10
15

Come seconda immagine test si è scelta una immagine creata ad hoc con testo bianco su sfondo nero: le variazioni rapide da pixel neri a pixel bianchi su una scala di grigi corrispondono ad immagini con contenuti ad alta frequenza non trascurabili. I risultati in questo caso sono i seguenti, ancora al variare dell'ampiezza N del rumore:

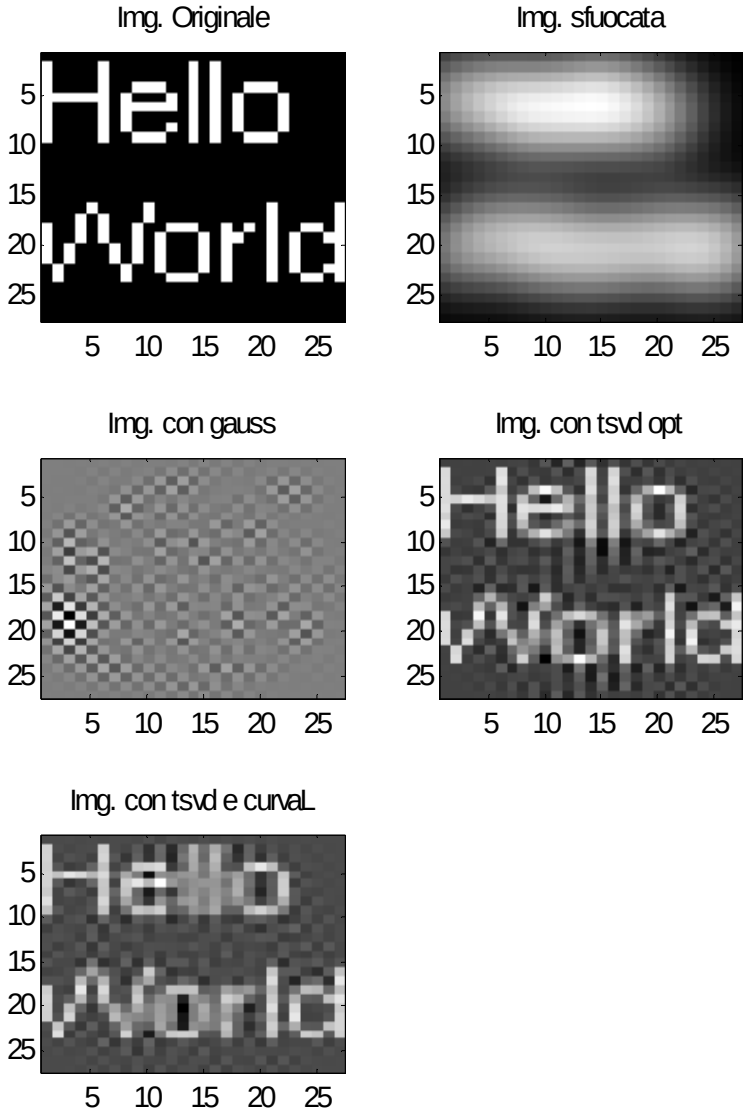
N=10^-005



N=10^-010


Img. Originale
Img. sfuocata
Img. con gauss
Img. con tsvd opt
Img. con tsvd e curvaL
5
10
15
20
25

N=10^-015



Conclusione

I risultati simulativi con le due immagini campione hanno mostrato come il metodo della curva-L associato alla fattorizzazione TSVD sia un valido metodo per la risoluzione di sistemi lineari fortemente malcondizionati anche in presenza di elevato rumore sui termini noti. Purtroppo l'eccessivo onere computazionale limita l'utilizzo di questa tecnica nell'image processing a causa dell'elevatissimo ordine dei sistemi lineari generati per la soluzione. Questo metodo risulta molto utile anche nella risoluzione di sistemi lineari sovradeterminati e sottodeterminati (rettangolari) in svariate applicazioni.