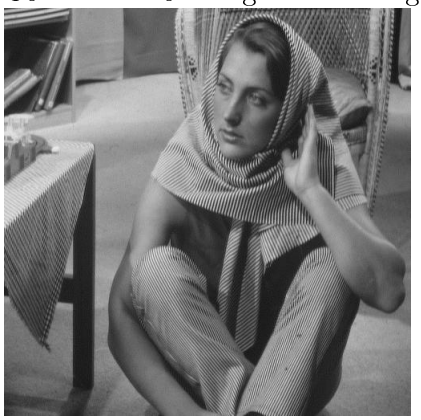
Ricostruzione Di Immagini

Hergert Gjoni

1.Introduzione

In questo articolo parleremo di ricostruzione di immagini.
Le immagini che tratteremo saranno tutte in bianco e nero e non a colori.
La questione principale peró non sarà tanto come ricostruire un'immagine sfocata, per cui esistono tecniche note. (Spiegheremo maggiori dettagli riguardo alla ricostruzione e sfocature di immagini nel Capitolo 3). In realtà ció che ci interessa é la questione seguente:
Sia A una matrice $m \times n$,che rappresenta un'immagine, k semiampiezza di una data psf [1], B $(m-2k) \times (n-2k)$ la matrice sfocata ottenuta mediante tale psf :
Come possiamo osservare dalle dimensioni, la matrice B é più piccola rispetto alla matrice A, dunque, sia A_1 l'immagine ricostruita da B, allora potrebbero esserci dei problemi nei pressi del bordo di A_1. Mostriamo questa cosa con il seguente esempio:
Consideriamo la seguente immagine:



Questa immagine ha dimensione 508x508

Questa é l'immagine sfocata di dimensioni 488x488

Questa invece é quella ricostruita.
Possiamo osservare che nei pressi del bordo l'immagine ricostruita é tutta nera, questo é dovuto alla mancanza di informazione che abbiamo nei pressi del bordo di B.

Come risolvere allora questo problema? Noi vorremmo avere una buona approssimazione nei pressi del bordo dell'immagine ricostruita, pur avendo mancanza di informazioni.
Allora procediamo così:

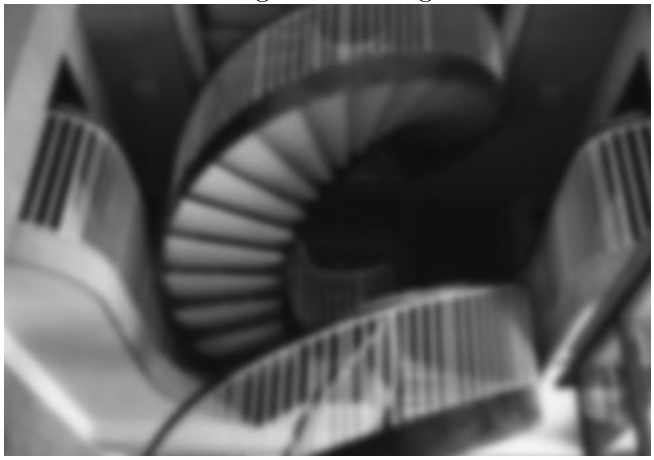
- La matrice B , che é di dimensioni $(m-2k)\times(n-2k)$ la estendiamo con una cornice larga k, dove k ricordiamo é la semiampiezza della psf, ottenendo una matrice B_1 di dimensioni $m \times n$.
- Applichiamo la tecnica di ricostruzione sulla matrice B_1 ottenendo una matrice A_2 che sarà, per coerenza di dimensioni, di dimensione (m+2k)×(n+2k).
- Togliamo ad A_2 la cornice di larghezza k, cioé togliamo ad A_2 le prime k sopra, le ultime k righe sotto, le prime k colonne a sinistra e le ultime k colonne a destra, in modo da renderla una matrice $m \times n$.

Facciamo un esempio:

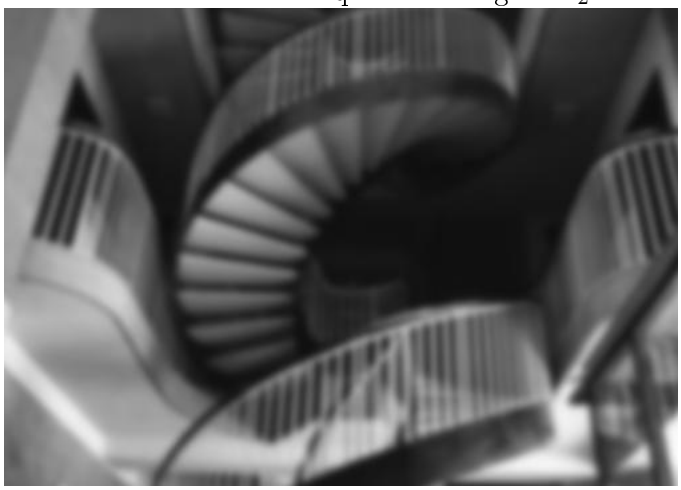
Consideriamo la seguente immagine A:

Ha dimensioni 358×505

Sia data allora la seguente immagine sfocata B di dimensioni 338×485



La estendiamo ottenendo questa immagine B_2 di dimensioni 358×505:



Otteniamo dunque la seguente immagine ricostruita A_2, che ha ancora dimensioni 358×505

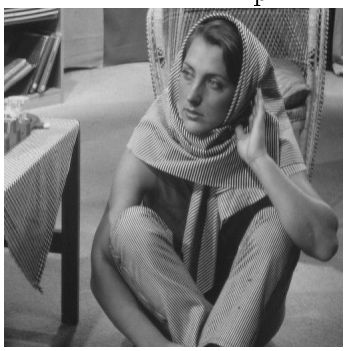
Come abbiamo potuto notare l'immagine viene ricostruita anche nei pressi del bordo.

In questo esempio abbiamo peró dato all'immagine B un'estensione che già conoscevamo. In un problema reale in realtà noi non conosceremo l'estensione, quindi il nostro obiettivo é proprio costruire un'estensione adatta di larghezza k.

Sarà importante dare una buona estensione all'immagine perché come vedremo successivamente la qualità dell'immagine ricostruita (specialmente sul bordo) dipenderà molto dall'estensione fatta sull'immagine sfocata.
Quindi la domanda é:
COME ESTENDERE B?
Esistono vari tipi di estensione già noti tra i quali:

- Bordo nero
- Bordo periodico
- Bordo riflettente
- Bordo anti-riflettente

Mostriamo un esempio sulla seguente immagine:



Ora le estensioni:

Nero

Periodico

Riflettente

Anti-Riflettente

Per maggiori dettagli vi rimandiamo all'articolo [3].

Le estensioni elencate sono semplici da realizzare. Esistono anche immagini in cui esse sono approssimazioni accettabili, per esempio queste che sono estese rispettivamente con il bordo nero e il bordo riflettente


Purtroppo esistono molte immagini in cui queste estensioni non vanno bene, e l'esempio di sopra con l'immagine della donna ne é una chiara conferma.
Quindi, sempre nell'articolo [3] , viene proposta una nuova tecnica di estensione:
La cosiddetta *Synthetic boundary conditions*.

Nel prossimo capitolo, il capitolo 2, entreremo nei dettagli riguardanti questa tecnica, mentre nel capitolo 3 faremo alcuni richiami di teoria sulla ricostruzione di immagini e in seguito effettueremo esperimenti per mettere a confronto le ricostruzioni partendo con le varie estensioni proposte, usando psf diverse, con diversi livelli di intensità, mentre nel capitolo 4 trarremo le nostre conclusioni

2.Synthetic BC

In questo capitolo mostreremo come effettuare un'estensione usando una Synthetic BC.
Per semplicità ció che mostreremo lo mostriamo su immagini rappresentate da matrici di dimensioni $m \times n$, dove m ed n saranno pari, ma é possibile tranquillamente anche estendere il tutto a immagini di dimensioni generiche. Prendendo spunto dalla tecnica proposta dall'articolo [3] ció che faremo é questo:

Sia A $m \times n$ come abbiamo detto sopra. Gli elementi di A saranno numeri reali appartenenti all'invervallo $[0, 1]$, ma va bene anche se A fosse uint8 o di altro tipi che permettono di rappresentare un'immagine, l'importante é che prima che si eseguano le operazioni A sia trasformata in double (per poi essere ritrasformata alla fine).

Come procediamo dunque per costruire il nostro bordo di larghezza fissata k, dove k sarà la semiampiezza della nostra psf?

Supponiamo che $k \geq 2$ e che k sia pari:
costruiremo allora delle colonne $q_1, q_2, ..., q_{\frac{k}{2}}$ di dimensioni $n \times 2$ in modo da ottenere una nuova matrice $[\,^{A\ q_1\ q_2\ \cdots\ q_{\frac{k}{2}}}\,]$ di dimensioni $n \times (n + k)$.
Come saranno queste colonne?

La colonna q_i sarà fatta così:

$$\begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{i\frac{n}{2}} \end{bmatrix}$$

Le x_{ij} sono matrici 2×2 che verranno costruite come mostriamo in seguito:

Proprio a tal proposito chiameremo A_i la matrice $[\, A\ q_1\ q_2\ \dots\ q_i\,]$ con la convenzione che A_0 sia proprio A.

Iniziamo dunque a costruire q_1:
Facendo uso della tecnica proposta nell'articolo [3] costruiamo le varie x_{1j} procedendo così:
-Individuiamo la riga $r = 2(j-1)+1$ della matrice A
-Denominiamo una sottomatrice di A $A_{c,d,s,t}$ fatta nel seguente modo:

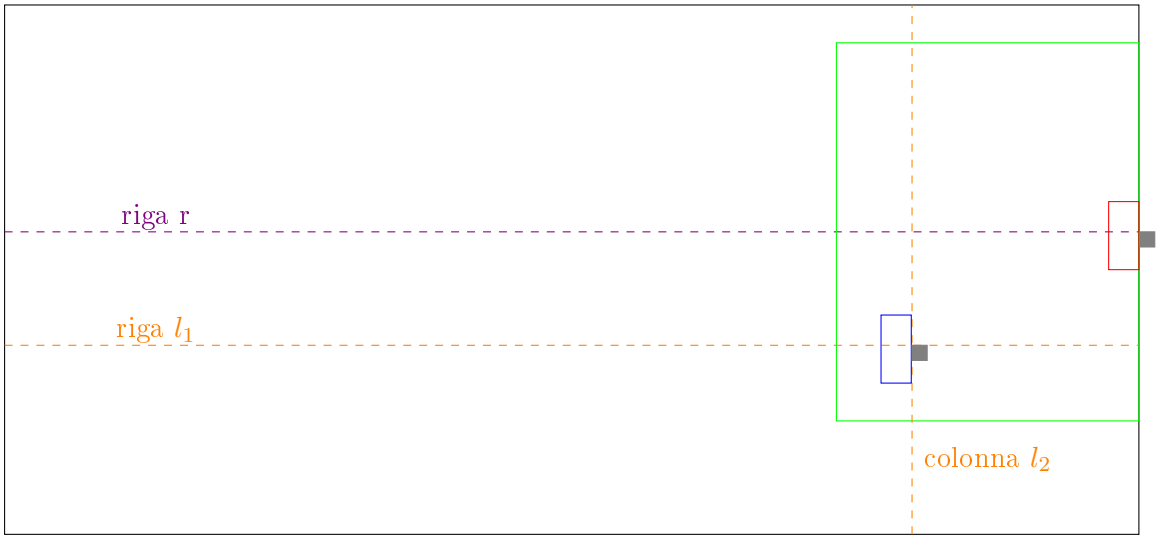
$$\begin{bmatrix} a_{c-s,d-t+1} & \dots & a_{c-s,d} \\ \vdots & \dots & \vdots \\ \vdots & \dots & a_{c,d} \\ \vdots & \dots & a_{c+1,d} \\ \vdots & \dots & \vdots \\ a_{c+1+s,d-t+1} & \dots & a_{c+1+s,d} \end{bmatrix}$$

che sarà una $(2+1)s \times t$.
-Costruiamo una sottomatrice A_{r,n,k_1,k_2} dove k_1, k_2 sono due interi da fissare,e diremo in seguito come.
-Costruiamo una sottomatrice A_{r,n,h_1,h_2} con h_1, h_2 che saranno più piccoli di k_1, k_2, in genere anche non di poco.
-$\forall l_1, l_2$ t.c. $l_1 = r-k_1, ..., r+k_1+1$,$l_2 = n-k_2+1, ..., n$ costruiamo la sottomatrice A_{l_1,l_2-1,h_1,h_2} e poi calcoliamo $\|A_{r,n,h_1,h_2} - A_{l_1,l_2-1,h_1,h_2}\|_F^2$ dove $\|\,\cdot\,\|_F$ é la norma di Frobenius di una matrice.
-Prendiamo la coppia di indici (l_1^0, l_2^0) che ci dia il minimo tra tutte queste norme e dunque la nostra x_{1j} sarà la matrice

$$\begin{bmatrix} a_{l_1^0,l_2^0} & a_{l_1^0,l_2^0+1} \\ a_{l_1^0+1,l_2^0} & a_{l_1^0+1,l_2^0+1} \end{bmatrix}$$

Vedere ([3] p.2251)

Per capire meglio vediamo la seguente figura:



Ció che si vuole dire é che:
-L'immagine A é delimitata dal riquadro nero.
-si individua la riga r individuata in figura dalla linea viola
-si costruisce l'area di ricerca A_{r,n,k_1,k_2} che in figura sarà tutta l'area delimitata dal rettangolo di bordo verde
-si costruisce un intornino adiacente A_{r,n,h_1,h_2} che in figura sarà l'area piccola (in genere piccola) delimitata dal rettangolino di bordo rosso
-si effettua una scansione sull'area delimitata in verde e si cerca ogni coppia di indici (l_1, l_2) all'interno di tale area, che in figura sono individuate dalle linee arancioni, e si confronta l'area delimitata dal rettangolo di bordo blu con quella rossa.

Con quale criterio farlo lo abbiamo già detto in precedenza. Una volta trovato (l_1^0, l_2^0) il quadratino 2×2 che abbiamo già definito in precedenza, e che in figura é rappresentato da un quadratino grigio accanto al rettangolino blusarà quello che metteremo nella nostra q_1 esattamente come si vede in figura a fianco del rettangolino rossa, e soprattutto come é stato già detto in precedenza.
Intuitivamente si cerca con una scansione sull'area verde il rettangolino blu che assomiglia di più a quello rosso, perché così é ragionevole pensare che il quadratino che sta di fianco al rettangolino blu sia un buon pezzettino da mettere sul bordo dell'immagine.

-Costruiremo dunque la nostra q_1 costruendo le varie x_{1j} come abbiamo proposto sopra.
Le altre q_i le costruiremo applicando la stessa tecnica sulle matrici A_{i-1}.

Abbiamo quindi costruito il bordo destro della nostra immagine. Per gli altri bordi possiamo procedere effettuando le seguenti osservazioni:
-Data una matrice P $m \times n$ definiamo $\tilde{P}$ la matrice t.c. $\tilde{P}e_k = Pe_{n+1-k}$ $\forall$ $k = 1, ..., n$ con n uguale al numero di colonne di P e di $\tilde{P}$ e $k = 1, ..., n$, e gli e_j sono i vettori della base canonica di $\mathbb{R}^m$. In pratica $\tilde{P}$ é la matrice con le colonne invertite di P.

-Dunque prendiamo la nostra matrice A $m \times n$

-Vogliamo costruire una cornice di larghezza k

-Costruiamo allora la matrice A_{dx} che é la matrice $[\,A \mid bordodx\,]$ dove $bordodx$ é il bordo $m \times k$ che abbiamo costruito usando la tecnica sopra proposta.

-Costruiamo la matrice $A_{dx,up}$ che sarà la matrice $\begin{bmatrix} bordoup \\ A_{dx} \end{bmatrix}$ dove $bordoup$ sarà il bordo $k \times (n+k)$ che costruiamo applicando la tecnica sopra proposta alla matrice $\tilde{A_{dx}^T}$

-Costruiamo allo stesso modo la matrice $A_{dx,up,sx}$ che sarà $[\,bordosx \mid A_{dx,up}\,]$ applicando la tecnica sulla matrice $\tilde{A}_{dx,up}$

-E analogamente la matrice $A_{dx,up,sx,down} = \begin{bmatrix} A_{dx,up,sx} \\ bordodown \end{bmatrix}$ applicando la tecnica sulla matrice $A_{dx,up,sx}^T$

Mostriamo un esempio

Questa é una 508×508

Questa é una 528×528, cioé estesa con una cornice larga 10 pixel.

Mostreremo alcuni esperimenti effettuati con la tecnica proposta.

Usiamo ancora la solita immagine.

Una cosa che abbiamo notato dagli esperimenti é che fissando h_1, h_2 per tutti e quattro i bordi non si ottengono i migliori risultati, perché possiamo estendere magari un bordo in maniera soddisfacente, ma un altro bordo potrebbe essere problematico.

Mostriamo un esempio:

Nella figura qui sotto possiamo vedere che scegliendo $h_1 = 5$ $h_2 = 5$, ed estendendo il bordo destro e quello sinistro dell'immagine di 10 pixel, il bordo sinistro viene esteso con un'approssimazione decisamente più accettabile rispetto a quella del bordo destro nel quale in alto a destra é ben visibile una variazione netta di colore. Di conseguenza sarebbe meglio agire bordo per bordo, con h_1 e h_2 che variano a seconda del bordo.

C'é un'altra cosa che possiamo notare dagli esperimenti. Ovvero che effettuare la scansione su una grossa area della matrice puó essere controproducente, non solo ai fini dell'efficienza, ma anche dell'accuratezza.

Mostriamo un esempio con le seguenti due immagini di cui la prima é l'originale invertita, la seconda é estesa lungo il bordo destro:

Questo esperimento é stato effettuato scegliendo i valori $k_1 = 100$, $k_2 = 100$, $h_1 = 5$ $h_2 = 5$
Si vede chiaramente che nei pressi del tavolo si nota una piccola area in cui i pixel sono tutti di inensità molto diversa tra loro,che non si adatta per niente al resto della zona che li circonda.
Si puó verificare che anche scegliendo k_1 e k_2 in modo che l'area di ricerca sia strettissima si ottengono allo stesso modo risultati poco soddisfacenti.

Da un po' di esperimenti abbiamo potuto verificare che scegliendo k_1,k_2 di valore 20 si ottengono dei buoni risultati.(Per questo motivo useremo sempre questi due valori di k_1 e k_2)

L'immagine mostrata prima nell'esempio é stata costruita in base ai seguenti parametri:

- bordo destro: $h_1 = 10$, $h_2 = 10$
- bordo sinistro: $h_1 = 5$, $h_2 = 5$
- bordo di sopra: $h_1 = 5$, $h_2 = 4$
- bordo di sotto: $h_1 = 5$, $h_2 = 5$

Pur avendo detto che fissando h_1,h_2 non si ottengono i risultati migliori é comunque possibile,fissandoli opportunamente per tutti e quattro i bordi, che si possa trovare una buona estensione.
Per esempio fissiamo $h_1 = 10$, $h_2 = 4$ (che per come abbiamo detto sopra, significa costruire un intorno adiacente quadrato di lato 10)

- Allargando la cornice l'estensione dei bordi tende a curvarsi e a deformarsi
- Nell'esempio sopra (anche in quelli precedenti),l'estensione della sedia diventa problematica. Questo é dovuto al fatto che proprio sul bordo dell'immagine originale si trova sottilissima striscia scura, di colore molto diverso e molto più scuro rispetto al resto della sedia, dunque complica molto i dettagli perché essendo la tecnica praticamente un "copia e incolla" di riquadri 2×2 dell'immagine da estendere é difficilissimo trovarne uno da incollare che renda l'estensione uniforme.

Proprio a tal proposito, per provare a migliorare questi dettagli, si é pensato di introdurre la derivata orizzontale e la derivata verticale di una matrice.

Definizione 1 *Data una matrice A $m \times n$ $A = [\, a_1 \mid \dots \mid a_n \,]$ definiamo la sua derivata orizzontale DO_A la matrice $A = [\, a_1 - a_2 \mid \dots \mid a_{n-1} - a_n \,]$ di dimensioni $m \times (n-1)$*

Definizione 2 *Data una matrice A $m \times n$ definiamo la sua derivata verticale DV_A la matrice $(DO_{A^T})^T$*

Date queste due definizioni ora procediamo nel seguente modo: fissiamo tre parametri α, β, γ in modo che la loro somma faccia 1, e che α abbia il valore maggiore.
Allora, come prima, invece di calcolare
$\|A_{r,n,h_1,h_2} - A_{l_1,l_2-1,h_1,h_2}\|_F^2$
calcoliamo
$\alpha\|A_{r,n,h_1,h_2} - A_{l_1,l_2-1,h_1,h_2}\|_F^2 + \beta\|DO_{A_{r,n,h_1,h_2}} - DO_{A_{l_1,l_2-1,h_1,h_2}}\|_F^2 + \gamma\|DV_{A_{r,n,h_1,h_2}} - DV_{A_{l_1,l_2-1,h_1,h_2}}\|_F^2$
e cerchiamo di minimizzarlo come abbiamo fatto in precedenza.
Questo che significa intuitivamente? Prima cercavamo di minimizzare la variazione tra i due intornini adiacenti, ora cerchiamo anche di minimizzare la variazione dei pixel consecutivi orizzontalmente e verticalmente tra i due intornini.

Proveremo a fare qualche esperimento scegliendo i seguenti parametri:
$\alpha = 0.6, \beta = 0.2, \gamma = 0.2$

Effettuiamo il test sulla seguente immagine:

Scegliendo come prima h_1=10, h_2=4 ,allargando il bordo destro, otteniamo

A sinistra l'estensione é senza derivata, a destra con la derivata

Abbiamo dunque visto che i dettagli vicino alla sedia sono migliorati.

Ci sono peró due fattori sfavorevoli introducendo le derivate:

-L'estensione é sicuramente più costosa, e non di poco
-Non é detto che ci siano necessariamente miglioramenti

Mostriamo ció con un esempio sempre applicato sulla stessa immagine ora usata peró ponendo $h_1 = 5$ e $h_2 = 4$

Come prima a sinistra é senza derivata, a destra con la derivata.

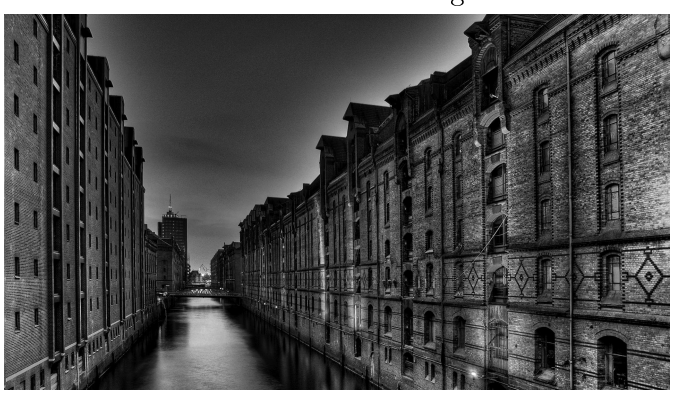
L'estensione é praticamente uguale in entrambi i casi, peró nel primo caso il tempo di esecuzione é stato di circa 5.994746 secondi, nel secondo caso é stato di circa 42.870877 secondi, un rapporto di circa 1:7, che potremmo dire un po' eccessivo considerando un miglioramento non netto, e occasionalmente evidente.

In conclusione anche se puó essere vantaggioso fare uso della derivata si consiglia di usufruirne il meno possibile.

Effettuiamo adesso un altra prova applicando la tecnica synthetic BC costruendo una cornice di 14 pixel.

Questa é l'immagine ottenuta con $h1 = 10$, $h_2 = 4$ e allargando la cornice di 14 pixel. Possiamo notare come la deformazione del libro si accentua sempre di più.

Mostriamo ora delle altre immagini ottenute con una cornice di 10 pixel.







Mostriamo adesso i codici più significativi:

Codice 1:

```
function x = myssd(A,B,b)
%function x=myssd(A,B)
%Questa funzione serve per calcolare la SSD
%A é la prima matrice
%B é la seconda matrice
%b é un numero intero che puó essere 0 oppure 1:

%Quando durante l'esecuzione vengono costruiti gli nbh
%puó succedere che le loro dimensioni non siano compatibili,
%o meglio i loro numeri di righe non lo saranno,
%questo succederà nei pressi del bordo di sopra, oppure nei pressi
%del bordo di sotto

 s1=size(A);
 s2=size(B);
 if(s1==s2)
 x=norm(A-B,'fro')^2;
 return;
 end

 s=min(s1(1),s2(1));

 if(b==1)
 x=norm(A(end-s+1:end,:)-B(end-s+1:end,:),'fro')^2;
 %Se b==1 vuol dire che alla matrice che ha più righe
 %le righe gliele dobbiamo togliere da sopra perché siamo
 %verso il bordo di sopra
 elseif(b==0)
 x=norm(A(1:s,:)-B(1:s,:),'fro')^2;
 %altrimenti da sotto perché siamo verso il bordo di sotto
 end

end
```

Codice 2:

```
function B=mynbh(A,ri,cj,d1,d2)
%mynbh(A,ri,cj,d1,d2)
%Questa funzione mi serve per creare gli nbh relativi all'elemento A(i,j) della
%
%ri é la riga i, cj é la riga j, d1 é la lunghezza dell'nbh, d2 é la semialtezza

s=size(A); s=s(1);
h=1; k=s;

if( ri-d2>0 )
 h=ri-d2;
end
 %Questo é un controllo che é necessario effettuare perché nel caso ri-d2
```

```
14  %sia <=0 la riga di A corrispondente a tale numero non esiste quindi dovremo pa
15  %dalla riga 1
16 if ( (ri+1+d2) < s+1)
17  k= ri+1+d2;
18 end
19  %Qui é lo stesso discorso
20
21 B=A(h:k,(cj-d1+1):cj);
22 end
```

Codice 3:

```
1  function x=mycopy(A,ii,l1,l2,m,n)
2  %mycopy(A,ii,l1,l2,m,n)
3  %Questa funzione serve per costruire il quadratino 2x2
4  %utilizzando la tecnica proposta
5  %e lo voglio attaccare sul bordo destro
6  %
7  %A é la nostra matrice
8  %ii é  la nostra riga
9  %l1,l2 sono la lunghezza e la semilarghezza dell'area in cui cercare i pixel
10 %m,n sono le dimensioni degli nbh
11
12
13 s=size(A);
14 s1=s(1);s2=s(2);
15
16 bi=0;
17 if(ii<(s1/2))
18   bi=1;
19 end
20 %Le immagini sono molto grandi
21 %Sia rispetto agli nbh
22 %Sia rispetto all'area di ricerca
23 %Quindi é ragionevole usare la metà di s2 come riferimento
24 %Per assegnare a b il valore 1 oppure 0 per la nostra myssd(.,.,b)
25
26 p=1;q=s1-1;
27 if( (ii-l2)>0 )
28 p=ii-l2;
29 end
30 %Come valeva il discorso per gli nbh
31 %vale anche per l'area di ricerca
32
33 if( (ii+l2)<s1 ) %non s1+1
34 q=ii+l2;
35 end
36 %non s1+1 perché cosé se il nostro argmin sta in riga s1-1
37 %possiamo prendere anche la riga s1 per il nostro quadratino 2x2
38
39
```

```
40 %Ora cerchiamo il minimo
41
42 minimo=-1;
43 argminimo=[0,0];
44
45 N=mynbh(A,ii,s2,m,n);
46
47 b=(s2-1):-1:(s2-l1); %Per le colonne
48 v=p:q; %Per le righe
49
50
51 for t=b %scorri le colonne
52  for r=v %scorri le righe
53  Nrt=mynbh(A,r,t-1,m,n);
54  %Perché non t? Perché a me serve l'intornino adiacente al pixel (r,t),
55  %non quello che lo prende
56     k=myssd(N,Nrt,bi);
57     if (k<minimo | minimo==-1)
58      minimo=k;
59           argminimo=[r,t];
60     end
61  end
62 end
63
64
65
66 kmin=argminimo(1);
67 lmin=argminimo(2);
68
69 x=A(kmin:kmin+1,lmin:lmin+1);
70
71 end
```

La function myssd.m serve per calcolare la funzione $SSD(A,B) = \|A-B\|_F^2$, mentre la function mynbh.m per costruire un intorno adiacente alla zona di indice ri, cj, mentre mycopy.m per inserire nella cornice il quadratino 2×2 di cui abbiamo parlato in precedenza.
Questi sono in effetti i codici più significativi, gli altri sono solo semplici iterazioni.

Tutti i codici si possono trovare nel seguente link .
In particolar modo le function bordodxmul.m,bordonordmul.m,bordosxmul.m,bordosudmul.m costruiscono rispettivamente il bordo destro, il bordo di sopra, il bordo di sinistra, il bordo di sotto.
Nella pagina raggiungibile tramite il link appena descritto é inoltre possibile trovare maggiori dettagli sulle immagini esposte finora.

3.Deblurring

Ora ci concentriamo sulla ricostruzione di un'immagine sfocata.
Prima di tutto spieghiamo come viene ottenuta un'immagine sfocata.
Sia k un intero positivo.
Sia A la matrice $(m+2k) \times (n+2k)$ che rappresenta l'immagine originale.
Sia F una matrice $(2k+1) \times (2k+1)$.
Chiameremo f_{rs} i suoi elementi con gli indici che li conteremo da $-k$ a k, cioé $r = -k, ..., k$ e $s = -k, ..., k$. Formalmente l'elemento f_{rs} si troverà nella posizione F(r+k+1,s+k+1) (linguaggio matlab). Un'altra caratteristica di questa matrice é che $\sum_{r,s=-k}^{k} f_{rs} = 1$ ed inoltre gli elementi f_{rs} devono essere non negativi.
La tabella dei numeri f_{rs} é chiamata PSF.
Gli indici degli elementi della matrice A li numeriamo da $1-k$ a $m+k$ per le righe e da $1-k$ a $n+k$ per le colonne, posizionati esattamente come sopra detto.
Gli elementi b_ij della matrice sfocata B $m \times n$ con $i = 1, ..., m$ $j = 1, ..., n$ sono ottenuti nel seguente modo:

$$b_{ij} = \sum_{r,s=-k}^{k} f_{rs} a_{i-r,j-s}$$

Per maggiori dettagli vi rimandiamo all'articolo [1].

Dunque, se chiamiamo $a = vec(A)$ e $b = vec(B)$ otteniamo che b é ottenuto tramite una matrice H dalla relazione $Ha = b$
La matrice H é fatta nel seguente modo:

$$H = \begin{bmatrix} F_k & F_{k-1} & \cdots & F_{-k+1} & F_{-k} & & & \\ & F_k & F_{k-1} & \ddots & F_{-k+1} & F_{-k} & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & & F_k & F_{k-1} & \cdots & F_{-k+1} & F_{-k} \end{bmatrix}$$

Dove le matrici F_s hanno dimensione $(m+2k) \times m$ hanno la seguente forma:

$$F_s = \begin{bmatrix} f_{k,s} & f_{k-1,s} & \cdots & f_{-k+1,s} & f_{-k,s} & & & \\ & f_{k,s} & f_{k-1,s} & \ddots & f_{-k+1,s} & f_{-k,s} & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & & f_{k,s} & f_{k-1,s} & \cdots & f_{-k+1,s} & f_{-k,s} \end{bmatrix}$$

Matrici come la matrice F_s vengono chiamate matrici di Toeplitz. Per maggiori dettagli vi rimandiamo agli articoli [1] e [2].

Poiché l'immagine A noi non la conosciamo il nostro problema é risolvere il sistema lineare $Hx = b$.
Procederemo così allora:
La soluzione che cerchiamo é $\tilde{a} = H^+ b$ dove $H^+ = H^T (HH^T)^{-1}$ é l'inversa generalizzata destra di H.
Abbiamo peró dei problemi:
-Le dimensioni di H sono proibitive
-Ci tocca effettuare un'inversione di matrice

Dunque la prima cosa che calcoliamo é $(HH^T)^{-1}b$, lo faremo risolvendo il sistema lineare $(HH^T)x = b$ e x sarà il valore che cerchiamo.
Poi moltiplicheremo la matrice H^T per il vettore x. La matrice del sistema, come abbiamo già detto é molto grossa, quindi per risolvere questo sistema sarà necessario usare un metodo iterativo. Noi nei nostri esperimenti abbiamo usato il metodo del gradiente coniugato. [1]
Il gradiente coniugato si usa con matrici definite positive,e dato che vogliamo applicarlo sul sistema lineare $(HH^T)x = b$ possiamo farlo. Il gradiente coniugato in genere fa uso di un precondizionatore, ma noi ne abbiamo fatto a meno avendo effettuato gli esperimenti su immagini sufficientemente piccole da poter eseguire il metodo in tempi molto ragionevoli.
La matrice H é molto grande da memorizzare, ma ció che richiede il metodo del gradiente coniungato non é H , ma solamente i prodotti Hy e H^Tx dove x, y sono due qualsiasi vettori colonna. La particolare struttura della matrice H permette di effettuare con comodità questi prodotti con Matlab.
Vediamo come:
Volendo calcolare $y = H^Tx$ e $z = Hy$, poiché i nostri vettori sono in realtà ottenuti da oggetti bidimensionali possiamo considerare le matrici X, Y e Z che corrispondono proprio ai nostri x, y e z.
Possiamo dunque digitare i comandi Matlab

```
Z = conv2(X, FR, 'full');
Y = conv2(Z, F, 'valid');
```

Il comando conv2 implementa la convoluzione. [1]
Mostriamo adesso i codici:

Codice per sfocare l'immagine

```
function b = sfoca(a, psf)
% function b = sfoca(a, psf)
% sfoca una immagine memorizzata nella variabile a
% usando la PSF memorizzata nella variabile psf
% in uscita l'immagine sfocata b

%a = double(a);
s = size(a);
b = conv2(a,psf,'valid');
end
%b = uint8(b);
```

Codice per ricostruire l'immagine tramite il gradiente coniugato

```
function x = gc(b, psf, passi)
% function x = gc(b, psf, passi)
% risolve il sistema ai minimi quadrati Hx=b col gradiente
% coniugato
% b: termine noto del sistema (immagine sfocata) come matrice mxn
% psf: point-spread function che definisce la matrice H del sistema
% la matrice H ha più colonne che righe
% passi: numero di iterazioni del metodo del gradiente coniugato
% Viene usata la seguente formula
% x = H^T (HH^T)^{-1}b, dove il vettore
```

```
% y = (HH^T )^{-1}b viene calcolato risolvendo
% il sistema (HH^T)y=b col metodo del gradiente coniugato


% calcolo la psf associata a H^T
psft = psf(end:-1:1, end:-1:1);

% dimensioni degli oggetti
[mb,nb] = size(b);
% inizio le iterazioni di GC
 b = double(b);
r = b;
y = zeros(mb,nb);

for iter=1:passi

 rho = sum(sum(r.*r));
if iter==1
p = r;
else
beta = rho/rhop; %rhop sta per rho precedente, che viene fissato, perché poi rho
p = r + beta*p;
end

tmp = conv2(p, psft, 'full');
q = conv2(tmp, psf, 'valid');
alpha = rho/sum(sum(p.*q));
y = y + alpha*p;
r = r-alpha * q;
rhop = rho;
end
x = conv2(y, psft, 'full');

%x = uint8(x);

end
```

Abbiamo inoltre un ulteriore problema:
Ció che finora non abbiamo detto é che in realtà l'immagine sfocata che noi vediamo non é esattamente B, ma una B con l'aggiunta di un rumore, ovvero una $B + E$ con $E = \{\varepsilon_{ij}\}$.
Questo errore puó essere causato dal sistema fisico di rivelazione, tipo il sensore di una macchina fotografica, o pu'o essere causato dal "rumore di quantizzazione" numerica dovuto al fatto che l'intensità luminosa viene rappresentata in modo discreto con un intero compreso tra 0 e 255 (nel caso si decida di usare tale rappresentazione).
Questo comporta che il sistema che noi andiamo a risolvere non é $Hx = b$ ma $Hy = b + \varepsilon$ dove $\varepsilon = vec(E)$. Sia $H = U\Sigma V^H$ una decomposizione ai valori singolari di H, se chiamiamo v_i le colonne di V, cioé i vettori singolari destri, si puó verificare che $y - x$ é combinazione lineare di questi vettori, e in particolar modo l'errore viene amplificato lungo i v_i relativi ai valori singolari σ_i più piccoli.
Tanto più é piccolo σ_i tanto é maggiore l'amplificazione dell'errore.
Generalmente i vettori singolari relativi ai valori singolari più piccoli sono quelli che contribui-

scono a creare il microdettaglio dell'immagine, vengono perció chiamati *vettori di alta frequenza*. Essendo il rumore annidato nelle componenti di alta frequenza il rumore si nota come una granulosità fitta e sottile.
Con il metodo del gradiente coniugato si riesce a verificare che con poche iterazioni si riesce a mettere a fuoco l'immagine, con molte iterazioni invece il rumore viene amplificato parecchio.
Per maggiori dettagli vi rimandiamo al testo [2] .
Mostriamo un esempio:

Immagine originale

Immagine sfocata

Immagine ricostruita con il gc con 10 iterazioni
(senza condizioni al bordo)

Come abbiamo detto l'immagine viene rimessa a fuoco, peró i microdettagli non sono stati evidentemente ricostruiti.
Ma che succede se effettuiamo tante iterazioni?

Immagine ricostruita con il gc con 160 iterazioni
(senza condizioni al bordo)

Notiamo chiaramente che non riusciamo a migliorare i dettagli dell'immagine, perché si manifesta chiaramente la presenza del rumore, come detto in precedenza.

Dunque il problema diventa proprio questo:
Come risolvere questo problema?
Usiamo la *regolarizzazione di Tikhonov*. [2]
La tecnica consiste in questo:
Utilizzando il gc ció che contribuisce in maniera significativa all'errore é la matrice HH^T, che é malcondizionata. Dunque si cerca un parametro λ "piccolo" tale che invece di risolvere il sistema $HH^Tx = b$ si va a risolvere il sistema $(HH^T + \lambda I)w = b$. In questo modo,scegliendo opportunamente λ gli autvalori più piccoli di questa nuova matrice si allontanano molto da quelli della matrice HH^T e lasciano praticamente invariati quelli più grandi.
Mostriamo ora il codice del gc introducendo questo nuovo parametro.

Codice:

```
function x = gc(b, psf, passi,reg)
% function x = gc(b, psf, passi)
% risolve il sistema ai minimi quadrati Hx=b col gradiente
% coniugato
% b: termine noto del sistema (immagine sfocata) come matrice mxn
% psf: point-spread function che definisce la matrice H del sistema
% la matrice H ha più colonne che righe
% passi: numero di iterazioni del metodo del gradiente coniugato
% Viene usata la seguente formula
% x = H^T (HH^T)^{-1}b, dove il vettore
% y = (HH^T )^{-1}b viene calcolato risolvendo
% il sistema (HH^T)y=b col metodo del gradiente coniugato
%reg é il parametro di regolarizzazione

% calcolo la psf associata a H^T
psft = psf(end:-1:1, end:-1:1);

% dimensioni degli oggetti
[mb,nb] = size(b);
% inizio le iterazioni di GC
  b = double(b);
r = b;
y = zeros(mb,nb);
```

```
24
25 for iter=1:passi
26
27  rho = sum(sum(r.*r));
28 if iter==1
29 p = r;
30 else
31 beta = rho/rhop; %rhop sta per rho precedente, che viene fissato, perché poi rho
32 p = r + beta*p;
33 end
34
35 tmp = conv2(p, psft, 'full');
36 q = conv2(tmp, psf, 'valid');
37 q=q+reg*p; %la modifica sta proprio qui
38 alpha = rho/sum(sum(p.*q));
39 y = y + alpha*p;
40 r = r-alpha * q;
41 rhop = rho;
42 end
43 x = conv2(y, psft, 'full');
44
45 %x = uint8(x);
46
47 end
```

Mostriamo ora un esempio ricostruendo l'immagine precedente:

Immagine ricostruita con il gc con 160 iterazioni (senza condizioni al bordo), scegliendo $\lambda = 0.001$

Si puó ben notare la differenza.
Dunque, il parametro λ e il numero di iterazioni sono due fattori importanti per la ricostruzione dell'immagine.

Mostreremo ora un po' di esperimenti procedendo nel seguente modo:
Sia A $m \times n$ la matrice rappresentante l'immagine originale, F psf di sempiampiezza k. Costruiamo B $(m-2k) \times (n-2k)$ sfocando A tramite la psf F.
Aggiungiamo a B una matrice εW dove W =rand($m-2k$,$n-2k$), e ε sarà sempre 10^{-2} oppure 10^{-1} (cioé effettuiamo esperimenti con aggiunta di rumori di diverso tipo).

Sia B_ε questa nuova matrice.
Porremo le nostre condizioni al bordo fino ad ottenere una matrice $\tilde{B_\varepsilon}$ $m \times n$.
Poi usando il gc con un parametro λ ricostruiamo l'immagine.
λ lo sceglieremo dell'ordine di ε.

Porremo diverse condizioni al bordo sull'immagine B_ε, e confronteremo le ricostruzioni con le varie condizioni. Per farlo useremo la funzione psnr di matlab (vedere [3]).
Se $\tilde{A}$ é l'immagine ricostruita psnr(A,$\tilde{A}$) ci darà un valore che ,più alto sarà, più $\tilde{A}$ si avvicina ad A.Infatti psnr$(A, \tilde{A})$ é data dalla seguente formula:

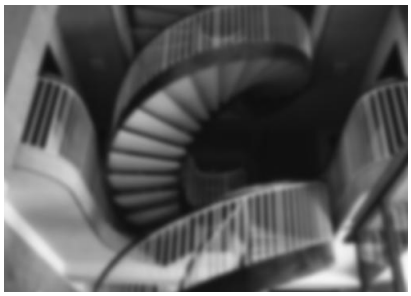
$$\text{psnr}(A, \tilde{A}) = 10\ log_{10} \frac{mnR}{\|A-\tilde{A}\|_F^2}$$

Dove R indicherà il massimo valore dei pixel dell'immagine.

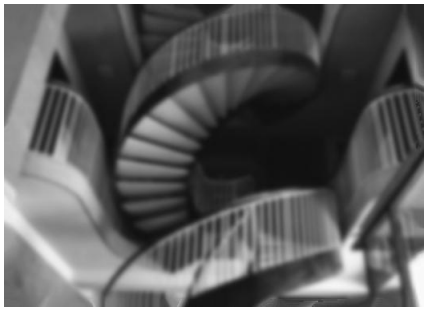
Prendiamo questa immagine.

La sfochiamo usando una psf gaussiana ([5]);

Sceglieremo la semiampiezza $k = 10$.



Sfocata con rumore 0.01



Estesa con la tecnica Synthetic BC

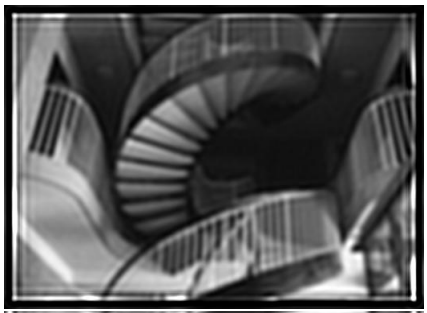
Ricostruita con $\lambda = 0.01$
psnr=19.9758

Ricostruita con $\lambda = 0.02$
psnr=19.6338

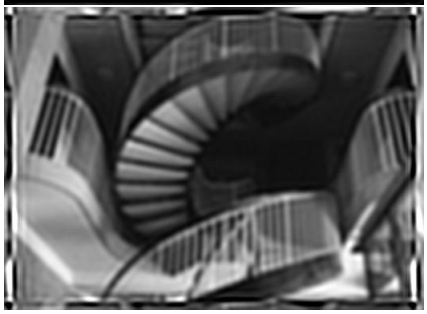
Ricostruita con $\lambda = 0.05$
psnr=19.0768

Si vede che il parametro migliore scelto é $\lambda = 0.01$. Anche in altri esperimenti sarà sempre così.
Dunque sceglieremo sempre $\lambda = \varepsilon$.

Mostriamo adesso le ricostruzioni usando le altre estensioni citate in precedenza.



Ricostruita con $\lambda = 0.01$ col bordo nero
psnr=13.4261

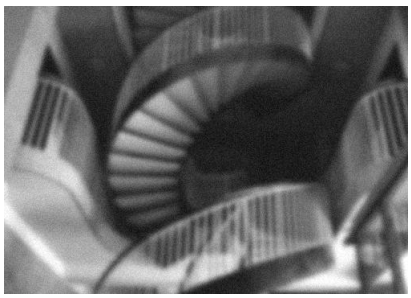


Ricostruita con $\lambda = 0.01$ col bordo periodico
psnr=16.0933

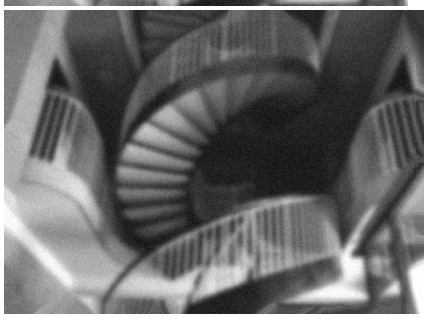
Ricostruita con $\lambda = 0.01$ col bordo riflettente
psnr=18.9427

Ricostruita con $\lambda = 0.01$ col bordo anti-riflettente
psnr=18.3629

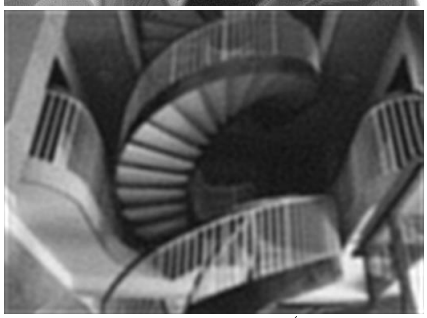
Adesso aumentiamo λ e otteniamo questi risultati:



Sfocata con rumore $\varepsilon = 0.1$



Estesa con le condizioni al bordo sintetiche



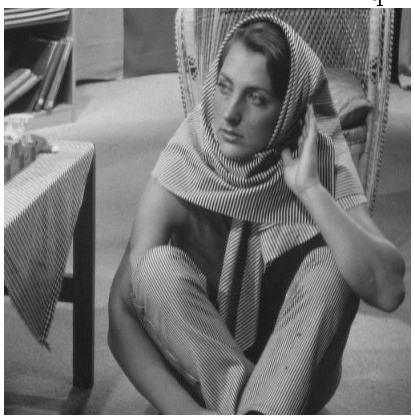
Ricostruita con le condizioni al bordo sintetiche con $\lambda = 0.1$
psnr=18.5335

Delle altre mostriamo (nello stesso ordine) solo le psnr:
-psnr=14.2870 (bordo nero)
-psnr=16.4052 (bordo periodico)
-psnr=17.2588 (bordo riflettente)
-psnr=17.6669 (bordo antiriflettente)

Se si vogliono anche le immagini e anche altri risultati é possibile andare sul seguente link .

Abbiamo dunque visto che i casi più interessanti sono sempre quelli con il bordo riflettente o anti-riflettente. Nei nostri esperimenti questo accadrà sempre. I risultati migliori li daranno, oltre all'estensione col bordo sintetico, sempre le estensioni col bordo riflettente e antiriflettente. Quindi sarà con quelle due estensioni con cui varrà la pena confrontare la nostra tecnica.

Proviamo adesso usando questa immagine



Questa volta sarà $\sigma = 10$, cioé la sfocatura sarà intensa. ε lo scegliamo 0.01.
Carichiamo adesso le immagini nel seguente ordine:
-sfocata con rumore
-estesa con le Synthetic BC
-ricostruita con la Synthetic BC (psnr=22.3634)
-ricostruita con le condizioni riflettenti (psnr=22.1514)
-ricostruita con le condizioni anti-riflettenti(psnr=22.3093)

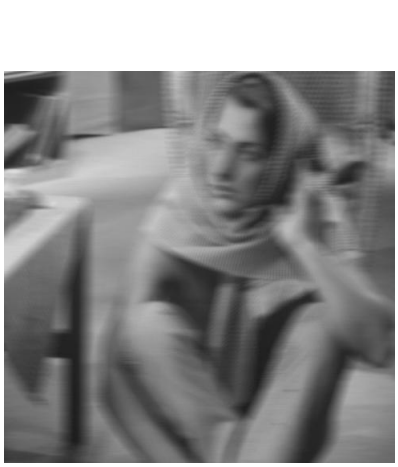
Possiamo notare che il sintetico dà un risultato migliore.
La differenza la fa soprattutto il bordo.

Aumentiamo adesso il rumore, e scegliamo $\varepsilon = 0.1$ e questa volta $\sigma = 7$
Carichiamo di nuovo le immagini nel seguente ordine:
-sfocata con rumore
-estesa con le Synthetic BC
-ricostruita con la Synthetic BC (psnr=21.5584)
-ricostruita con le condizioni riflettenti (psnr=21.4018)
-ricostruita con le condizioni anti-riflettenti(psnr=21.4993)

Mostriamo ora un esempio usando peró un'altra psf. $F = \{f_{rs}\}$ con $r = -10, ...10$ e $s = -10, ..., 10$, e f_{rs} é tale che

$$f_{rs} = \begin{cases} \frac{1}{21} & \text{se } r = 0 \\ 0 & \text{altrimenti} \end{cases}$$

Cioé é una psf che agisce solo in orizzontale.
Scegliamo $\varepsilon = 0.01$, e mostriamo le immagini nel seguente ordine.
-sfocata con rumore
-estesa con le Synthetic BC
-ricostruita con la Synthetic BC (psnr=23.6675)
-ricostruita con il bordo nero (psnr=15.6406)
-ricostruita con il bordo periodico (psnr=20.2931)
-ricostruita con le condizioni riflettenti (psnr=23.5582)
(La psnr corrispondente all'antiriflettente é uguale a 23.0012)






Notiamo che le condizioni al bordo non solo possono influire negativamente sul bordo, ma anche sul resto dell'immagine.

Facciamo un altro esempio scegliendo $\varepsilon = 0.1$ e questa volta scegliamo F con

$$f_{rs} = \begin{cases} \frac{1}{21} & \text{se } r = s \\ 0 & \text{altrimenti} \end{cases}$$

Cioé la psf é diagonale.
L'ordine delle immagini sarà nuovamente

-sfocata con rumore
-estesa con le Synthetic BC
-ricostruita con la Synthetic BC (psnr=21.2089)
-ricostruita con il bordo nero (psnr=16.3314)
-ricostruita con il bordo periodico (psnr=19.883)
-ricostruita con le condizioni riflettenti (psnr=21.1196)
(La psnr corrispondente all'antiriflettente é uguale a 21.1628)

Notiamo che a seconda dei casi tra riflettente e antiriflettente puó risultare migliore o uno o l'altro, ma in tutti i casi il sintetico dà sempre il risultato migliore.

Altri risultati é possibile trovarli sempre nel solito link .

Parametro gcv (generalized crossing validation)

Finora negli esperimenti abbiamo scelto sempre λ dell'ordine di ε.

Un altro approccio usato in letteratura consiste nella scelta di λ che minimizza la seguente funzione:

$$V(\lambda) = \frac{\|I - H(H^T H + \lambda I)^{-1} H^T b\|_2^2}{(trace(I - H(H^T H + \lambda I)^{-1} H^T))^2} \text{ [4]}$$

Dove H é la nostra matrice che trasforma A in B, e $b = vec(B + E)$.
Le dimensioni di H peró sono troppo grandi per la valutazione di questa funzione. Esistono tecniche per l'utilizzo di questa funzione anche su immagini grandi, sfruttando la struttura della matrice H.
Questi metodi vanno peró al di là dello scopo del presente lavoro.
Noi mostreremo un esperimento su un'immagine piccola (100×100) e procederemo così:
-Useremo la function csvd.m per effettuare una decomposizione svd compatta della matrice H ($H = U\Sigma V^T$)

-Useremo la function gcv.m (che fa uso della funzione gcvfun.m) che prenderà in input la matrice U, s vettore contenente i valori singolari, e b).

Queste function si trovano nel pacchetto RegTools sviluppato da Per Christian Hansen, e sono scaricabili dal seguente link.
`https://it.mathworks.com/matlabcentral/fileexchange/52-regtools`

Questa é l'immagine originale:

Ora la sfochiamo con una gaussiana di sempiampiezza 10 e con $\sigma = 4$ e aggiungiamo un rumore con $\varepsilon = 0.01$.

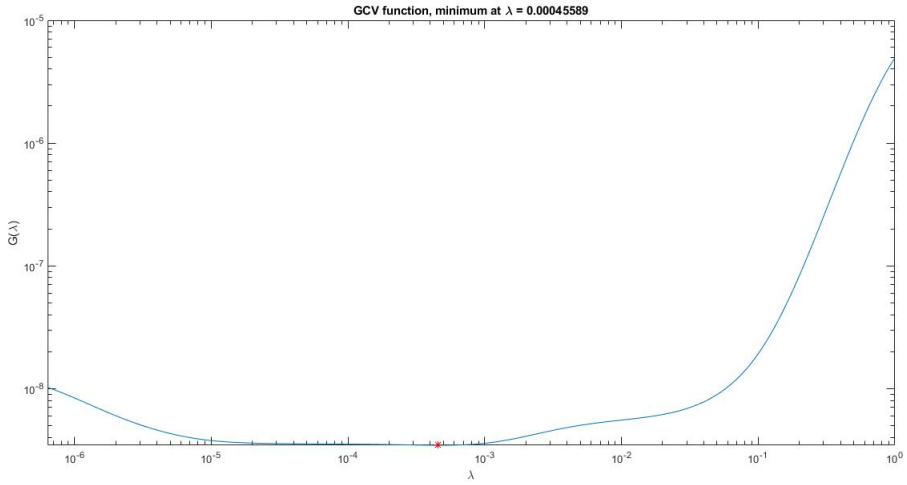
.

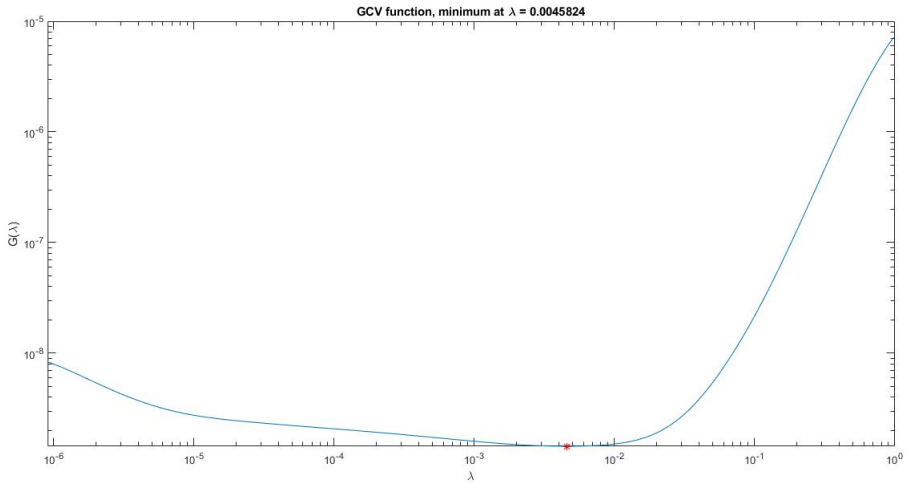
A questo punto ci chiediamo:

Il vettore b lo diamo in input ponendo le condizioni al bordo? Oppure senza?

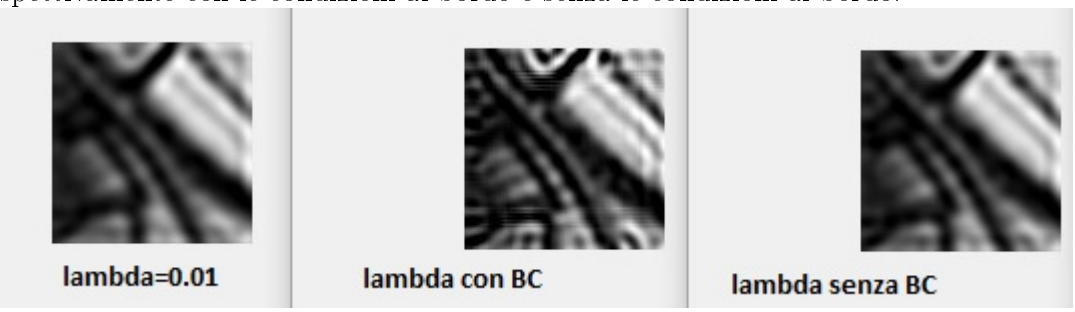
Facciamo allora un confronto. Osserviamo che dando b con le condizioni al bordo la nostra H da dare in input sarà una matrice $120^2 \times 100^2$ con la stessa struttura mostrata sopra. Mentre con b senza le condizioni al bordo diamo in input una H $100^2 \times 80^2$.
Mostriamo ora l'estesa con le condizioni al bordo sintetiche.

E ora i grafici delle funzioni gcv con i relativi punti di minimo.
La prima immagine é relativa alla b con le condizioni al bordo, la seconda senza.





Confrontiamo adesso le immagini ricostruite usando $\lambda = 0.01$ e poi gli altri due λ ottenuti rispettivamente con le condizioni al bordo e senza le condizioni al bordo.



Le psnr sono rispettivamente 14.2172 13.7029 14.2219.

Dunque, il λ ottenuto senza le condizioni al bordo é rislutato il λ migliore, non solo rispetto al λ ottenuto con le condizioni al bordo, ma anche rispetto a $\lambda = \varepsilon$.
Dunque non é detto che λ dell'ordine di ε sia il λ migliore, inoltre se si vuole cercare il λ con le condizioni al bordo l'estensione deve essere molto accurata affinché il valore cercato sia buono. Possiamo anche notare dal grafico relativo al λ con le condizioni al bordo che la funzione tende ad essere abbastanza piatta in un intervallo di ampiezza non trascurabile. In quell'intervallo si

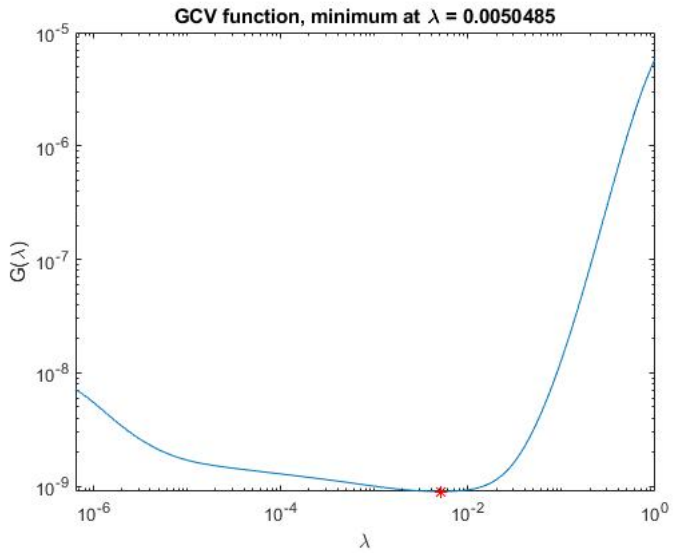
trova anche il nostro parametro di regolarizzazione trovato dalla gcv.m, ma anche altre scelte di λ in quell'intervallo potrebbero andare bene. Motivo in più per cui dando in input b con le condizioni al bordo si potrebbero avere maggiori problemi.

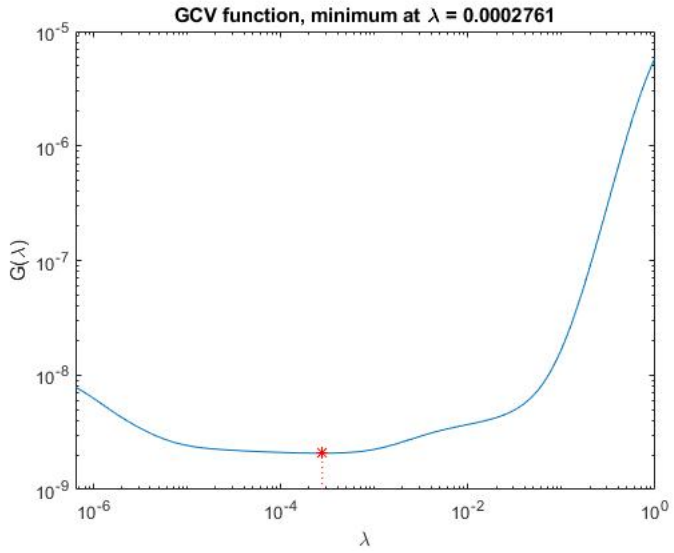
Mostriamo adesso un esempio con la seguente immagine:

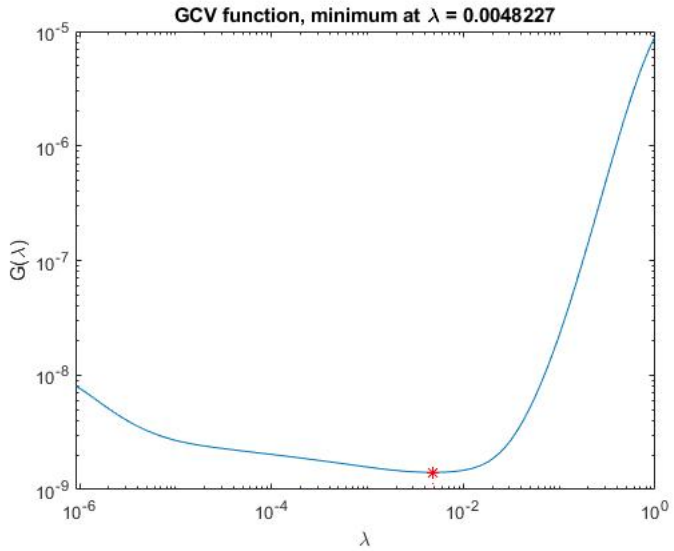
Sfocandola sempre con rumore $\varepsilon = 0.01$ otteniamo queste 3 immagini:

Queste immagini sono rispettivamente:
-l'immagine sfocata + rumore (B)
-l'immagine estesa con la synthetic BC (B.2)
-l'immagine estesa con il vero bordo (che in un problema reale non avremo) (B.3)

Mostriamo adesso i rispettivi λ ottenuti:







Se proviamo a ricostruire l'immagine (B.2) dando in input rispettivamente i 3 λ otteniamo le seguenti tre psnr:
-13.9074
-13.3935
-13.9042

Possiamo dunque osservare che i λ ottenuti senza l'estensione al bordo e con il bordo vero sono molto simili, di ordine di grandezza diverso rispetto a quello ottenuto con il bordo sintetico.
Inoltre anche l'immagine ricostruita con il λ ottenuto con il bordo sintetico é peggiore rispetto alle altre due.
In questo caso addirittura il λ ottenuto senza le condizioni al bordo é risultato il migliore
Dunque si suggerisce di cercare il λ senza porre a b le condizioni al bordo.

Dunque per la ricerca del λ si puó procedere così:

Data A $m \times n$ che rappresenta l'immagine originale, data B_ε $(m-2k) \times (n-2k)$ l'immagine sfocata con aggiunta di rumore i passi da eseguire sono i seguenti:

- calcolare $b = vec(B_\varepsilon)$
- effettuare la decomposizione svd (compatta) della matrice H tramite la function csvd.m
- calcolare λ tramite la function gcv.m che prende in input U,s,b (che abbiamo già detto in precedenza cosa sono)
- porre le condizioni al bordo all'immagine B_ε
- ricostruire l'immagine

Questo vale anche per altri tipi di rumore. Mostriamo i risultati con $\varepsilon = 0.1$.
Senza condizioni al bordo otteniamo $\lambda = 0.025416$ mentre con le condizioni al bordo otteniamo $\lambda = 0.014235$. Dunque le psnr che otteniamo sono 13.8593 ($\lambda = 0.1$), 13.8870 ($\lambda = 0.025416$) e 13.8400 ($\lambda = 0.014235$). Ancora una volta il λ ottenuto senza condizioni al bordo ha prodotto il risultato migliore.
Per vedere le immagini e altri risultati andare sul solito link .

4.Conclusioni

Come abbiamo potuto vedere in moltissimi esperimenti, al variare dei λ scelti, del rumore aggiunto e della PSF scelta, la ricostruzione col bordo sintetico é sempre risultato il migliore. Questo lo si é visto matematicamente dalle relative psnr, ma anche ad occhio si vede chiaramente la differenza. Lo svantaggio di questa tecnica rispetto alle altre proposte é che la costruzione del bordo dipende dall'immagine mentre le altre no. Quindi le altre tecniche sono ugualmente applicabili a tutte le immagini di qualunque tipo e di qualunque dimensione. Questa tecnica invece dipende dalle dimensioni, non solo perché l'accuratezza del bordo risulterebbe ancora più difficile come già spiegato nel Capitolo 2, ma anche perché per avere l'approssimazione più soddisfacente possibile in generale potrebbe essere richiesto più di un tentativo, il che diventa un po' arduo per immagini molto grandi. Bisogna quindi prestare particolare attenzione. Ma il lato positivo di questa tecnica é che produce estensioni migliori e ricostruzioni migliori.All'aumentare della larghezza delle cornici e/o delle dimensioni delle immagini, se l'estensione sarà ben fatta, la differenza con le altre tecniche sarà sicuramente più evidente.

Bibliografia

[1] Dario A. Bini. il restauro di immagini digitali (dispense del corso di calcolo scientifico 2018-2019). 2018.

[2] Dario A. Bini. il rumore e la regolarizzazione (dispense del corso di calcolo scientifico 2018-2019). 2018.

[3] Ying Wai Daniel Fan and James G Nagy. Synthetic boundary conditions for image deblurring. *Linear Algebra and Its Applications*, 434(11):2244–2268, 2011.

[4] Gene H Golub and Urs Von Matt. Generalized cross-validation for large-scale problems. *Journal of Computational and Graphical Statistics*, 6(1):1–34, 1997.

[5] Per Christian Hansen, James G Nagy, and Dianne P O'leary. *Deblurring images: matrices, spectra, and filtering*, volume 3. Siam, 2006.