

Relazione Progetto di Calcolo Numerico

Benatti Alice, Manuelli Matteo, Qayyum Shahbaz Ali

Gennaio 2022

Indice

1 Presentazione del problema	3
1.1 Generazione dataset	3
1.2 Generazione Immagini Corrotte	4
1.3 Osservazioni	5
1.3.1 Analisi immagine geometrica	5
1.3.2 Analisi immagine fotografica	5
1.3.3 Analisi immagine con testo	6
2 Ricostruzione di un immagine rispetto una versione corrotta	7
2.1 Metodo del Gradiente Coniugato (naive)	7
2.2 Metodo del Gradiente	8
2.3 Metodo del gradiente e Metodo del gradiente coniugato a confronto	10
3 Osservazioni	10
3.1 Metodo di Regolarizzazione di Tikhonov	10

1 Presentazione del problema

Il progetto ha come scopo quello di comprendere e mettere in atto metodi per ricostruire immagini blurate e svolgere il lavoro opposto, quindi generare immagini corrotte (dal rumore) a partire da un'immagine originale.

Il problema che ci è stato presentato riguarda la ricostruzione di immagini corrotte attraverso il blur Gaussiano.

Verrà analizzata inizialmente l'immagine `data.camera()` importata da `skimage`, successivamente verranno analizzate un set di 8 immagini con oggetti geometrici di colore uniforme su sfondo nero, realizzate da noi.

Il problema di deblur consiste nella ricostruzione di un'immagine a partire da un dato acquisito mediante il seguente modello:

$$b = Ax + \eta$$

dove b rappresenta l'immagine corrotta, x l'immagine originale che vogliamo ricostruire, A l'operatore che applica il blur Gaussiano ed η il rumore additivo con distribuzione Gaussiana di media \mathbb{E} e deviazione standard σ .

Per svolgere il progetto si farà uso dei moduli `numpy`, `skimage` e `matplotlib` utilizzando il linguaggio Python.

Affinché risultino chiari i valori a cui andremo a riferirci nella relazione, bisogna tenere ben presente il significato di questi due parametri.

PSNR (Peak Signal to Noise Ratio): Misura la qualità di un'immagine ricostruita rispetto all'immagine originale, la formula per calcolarlo è la seguente:

$$PSNR = \log_{10}\left(\frac{\max x^*}{\sqrt{MSE}}\right)$$

MSE (Mean Squared Error): Con la sigla ci riferiamo all'errore quadratico medio ed è così ottenuto:

$$MSE = \sqrt{\frac{\sum_{i=1}^n \sum_{j=1}^m (x_{ij}^* - x_{ij})^2}{nm}}$$

I due valori sono inversamente proposizionali, quindi più è alto il PSNR e basso l'MSE, più l'immagine sarà simile all'immagine originale. Il PSNR dipende dall'MSE.

Deviazione standard: E' un indice che ci permette di capire in maniera riassuntiva le differenze dei valori per ogni osservazioni rispetto alla media delle variabili.

1.1 Generazione dataset

E' richiesto un set di immagini con le seguenti specifiche:

- 8 Immagini di dimensione 512×512 ;
- Formato PNG in scala dei grigi;
- Devono contenere tra i 2 ed i 6 oggetti geometrici;
- Oggetti di colore uniforme su uno sfondo nero.

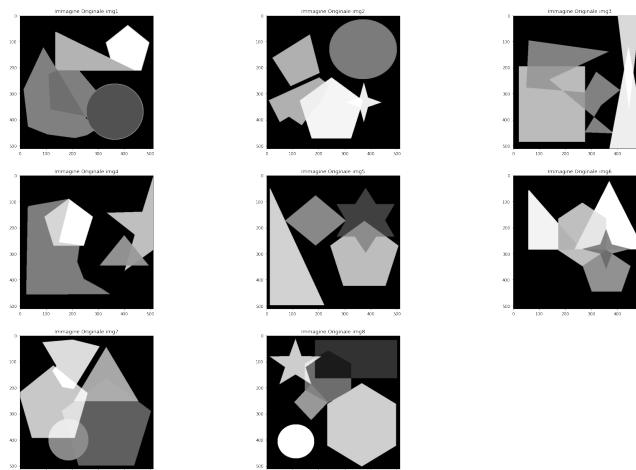


Figura 1: Immagini geometriche utilizzate

Useremo anche altre due immagini di tipo fotografico/medico/astronomico a scelta trovate su internet. Quest'ultime saranno importate all'interno del progetto con la libreria `skimage`, impostando il flag `as_gray=True` per averle in bianco e nero.

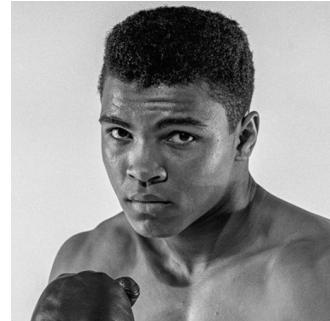
Le immagini selezionate sono le seguenti:

Immagine con Testo Composizione di prime pagine di giornale con i relativi articoli.

Immagine Fotografica Che ritrae il volto di una persona in modo dettagliato e con varie tonalità di grigio.



(a) Immagine con testo



(b) Immagine fotografica

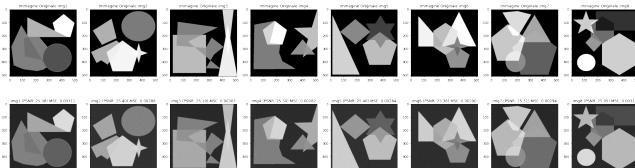
Figura 2: Immagini fotografiche analizzate

1.2 Generazione Immagini Corrotte

Obiettivo: Degradare le immagini applicando, mediante le funzioni riportate nella cella precedente, l'operatore di blur con parametri

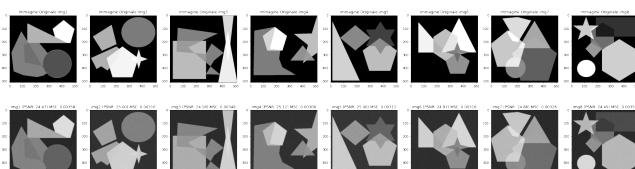
- $\sigma = 0.5$ dimensione 5×5
- $\sigma = 1$ dimensione 7×7
- $\sigma = 1.3$ dimensione 9×9

ed aggiunge rumore gaussiano con deviazione standard (0, 0.05)



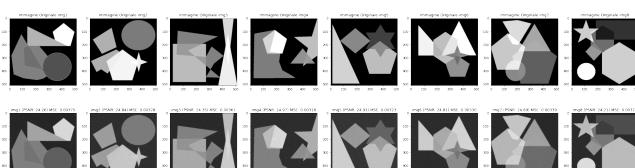
Valore	Risultato
Media PSNR	25.3137
Media MSE	0.002943
Dev. Std. PSNR	0.156402
Dev. Std. MSE	0.000106

Figura 3 & Tabella 1: Immagini corrotte con $\sigma = 0.5$ dimensione 5×5



Valore	Risultato
Media PSNR	24.7938
Media MSE	0.003321
Dev. Std. MSE	0.000194
Dev. Std. PSNR	0.252202

Figura 4 & Tabella 2: Immagini corrotte con $\sigma = 1$ dimensione 7×7



Valore	Risultato
Media PSNR	24.6292
Media MSE	0.003452
Dev. Std. MSE	0.000235
Dev. Std. PSNR	0.293261

Figura 5 & Tabella 3: Immagini corrotte con $\sigma = 1.3$ dimensione 9×9

1.3 Osservazioni

Osserviamo il risultato su un'immagine scelta casualmente del set creato e sulle due immagini aggiuntive.

1.3.1 Analisi immagine geometrica

Analizziamo l'immagine img8.png al variare del valore σ :

Ricordiamo che più è alto il valore del PSNR maggiore sarà la vicinanza dell'immagine corrotta rispetto alla versione originale.

Le figure di sinistra rappresentano l'immagine originale, invece a destra sono riportate le immagini corrotte con i rispettivi valori di PSNR. Notiamo che all'aumentare delle dimensioni di sigma il valore di PSNR diminuisce che denota un peggioramento della qualità dell'immagine, infatti le immagini subiscono un'appiattimento dell'intensità della scala dei colori e i contorni delle varie figure geometriche perdono di fermezza. Inoltre è curioso notare..

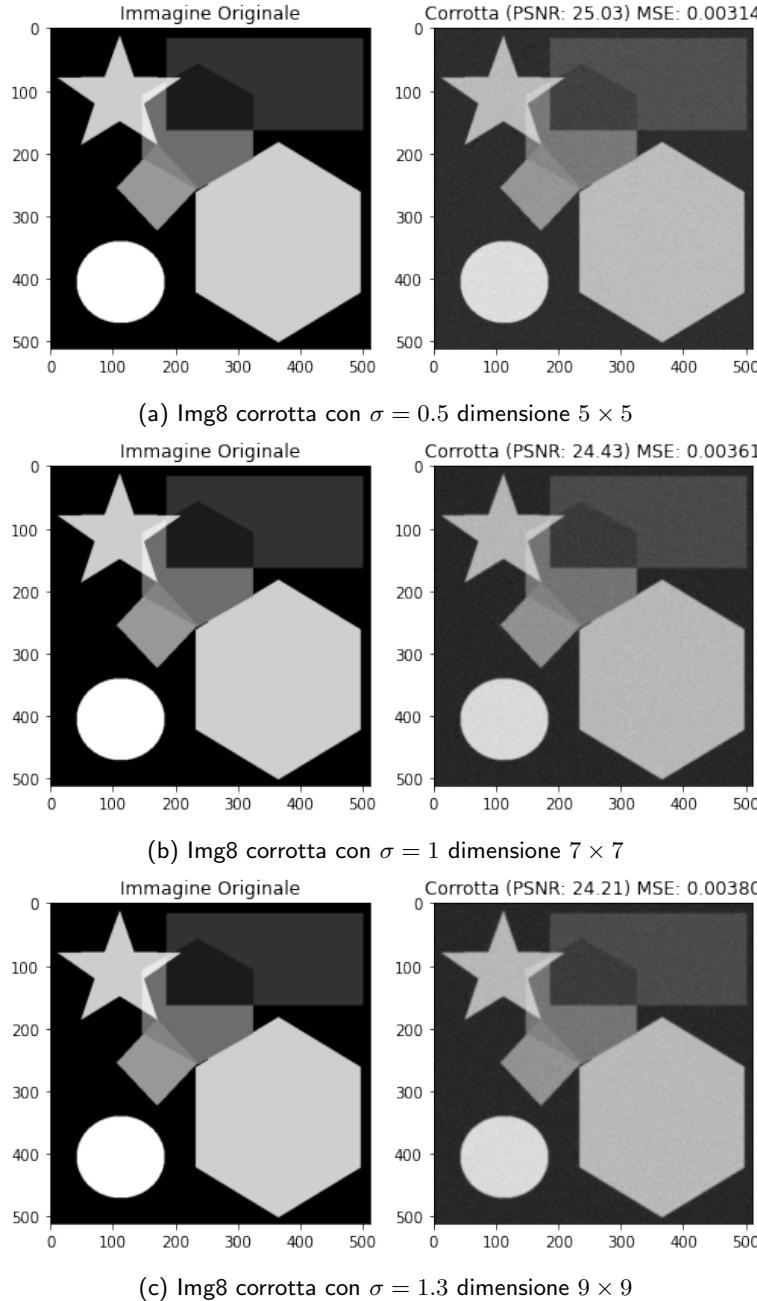


Figura 6: Immagine geometrica corrotta al variare di

1.3.2 Analisi immagine fotografica

Si nota un'altra volta che all'aumentare delle dimensioni di σ diminuisce il PSNR e l'immagine perde di incisività, le versioni corrotte benché risultino visivamente peggiori, si riesce ancora a ben distinguere il soggetto in primo piano,

anche se sfocato, in tutte le immagini.

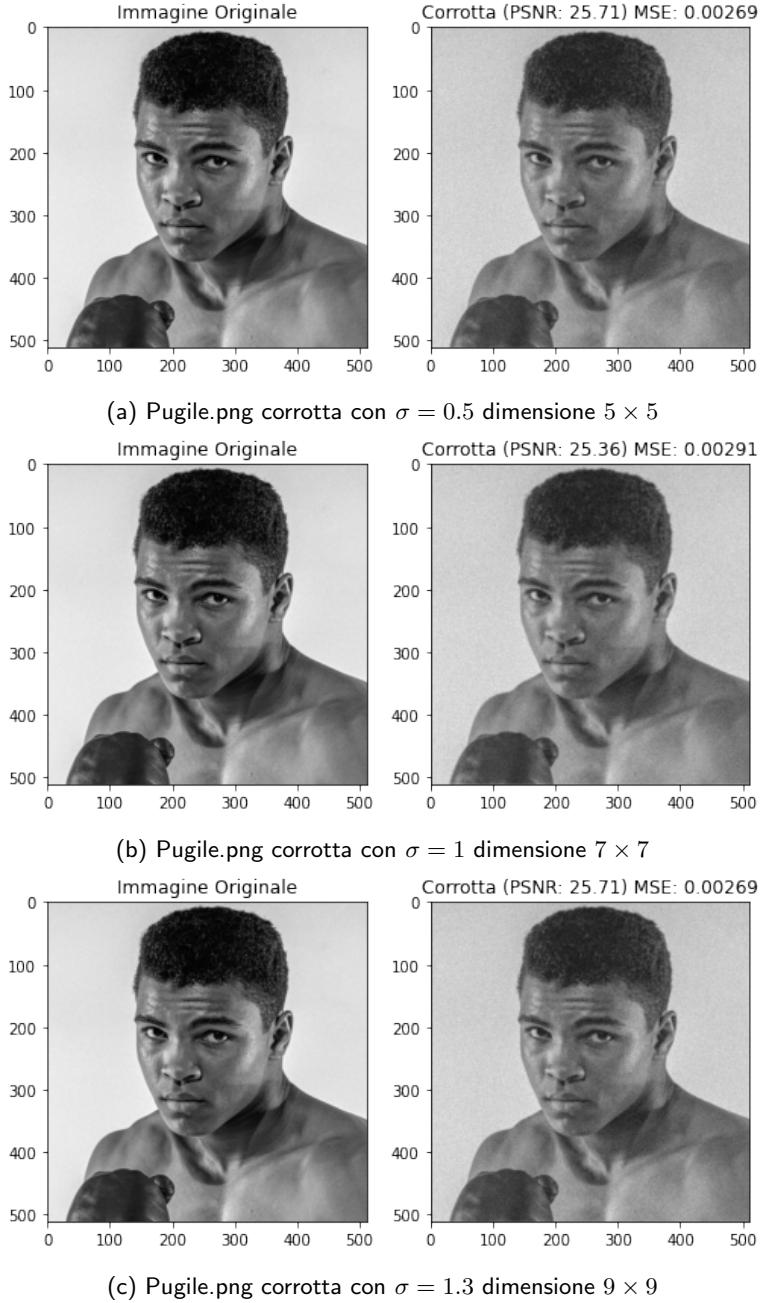
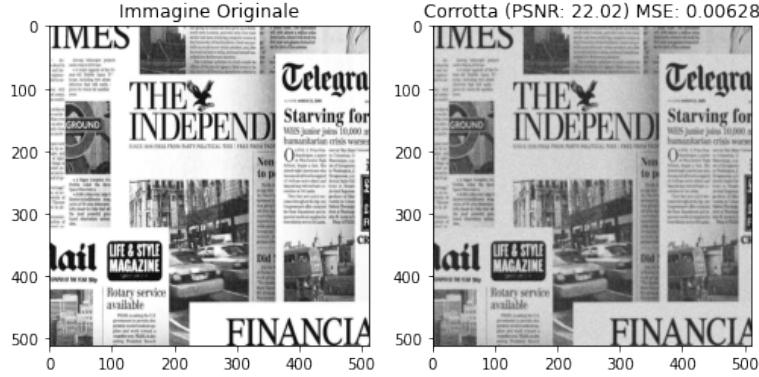


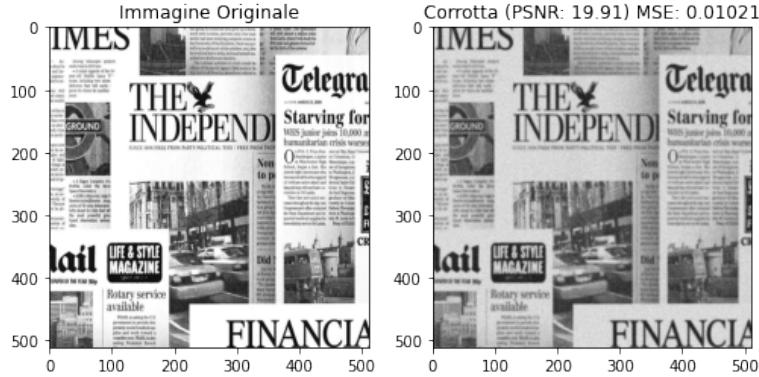
Figura 7: Immagine fotografica corrotta al variare di σ

1.3.3 Analisi immagine con testo

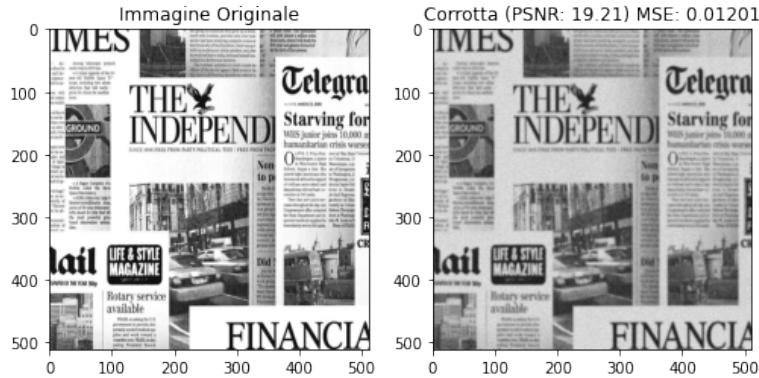
In questa immagine abbiamo una raccolta di prime pagine di giornale che ci permettono di osservare e valutare meglio la differenza tra l'immagine originale e la versione corrotta, per esempio con $\sigma = 0,5$ (vedi immagine 8a) otteniamo un'immagine con del testo ancora leggibile sebbene meno nitida, la difficoltà inizia ad essere maggiore invece con $\sigma = 1$ (vedi immagine 8b) dove le scritte più piccole diventano quasi illeggibili, con $\sigma = 1.3$ (vedi immagine 8c) il PSNR diminuisce ancora sebbene non molto rispetto rispetto a sigma uguale a 1, ma in questo caso anche le scritte più grandi, fatta eccezione per i titoli, perdono di chiarezza.



(a) giornale.png corrotta con $\sigma = 0.5$ dimensione 5×5



(b) giornale.png corrotta con $\sigma = 1$ dimensione 7×7



(c) giornale.png corrotta con $\sigma = 1.3$ dimensione 9×9

Figura 8: Immagine con testo corrotta al variare di σ

2 Ricostruzione di un immagine rispetto una versione corrotta

Una possibile ricostruzione dell'immagine originale x partendo dall'immagine corrotta b è la soluzione naïve data dal minimo del seguente problema di ottimizzazione:

$$x^* = \arg \min_x \frac{1}{2} \|Ax - b\|_2^2$$

Nell'analisi delle immagini ricostruite useremo $\sigma = 1$ dimensione 7×7 come valore standard.

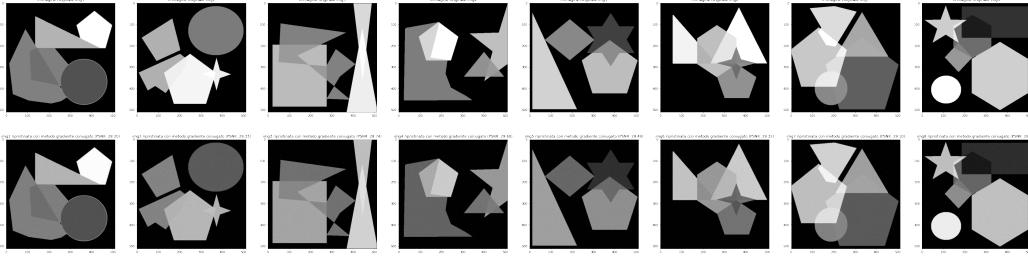
Abbiamo mostrato questi due grafici sovrastanti perché si evince che all'aumentare del numero delle iterazioni l'immagine ricavata si allontana dalla sua versione originale poiché è presente una deviazione.

2.1 Metodo del Gradiente Coniugato (naive)

Il metodo del gradiente coniugato è un algoritmo per la risoluzione numerica di un sistema lineare la cui matrice sia simmetrica e definita positiva e consente di risolvere il sistema in un numero di iterazioni che è al massimo n .

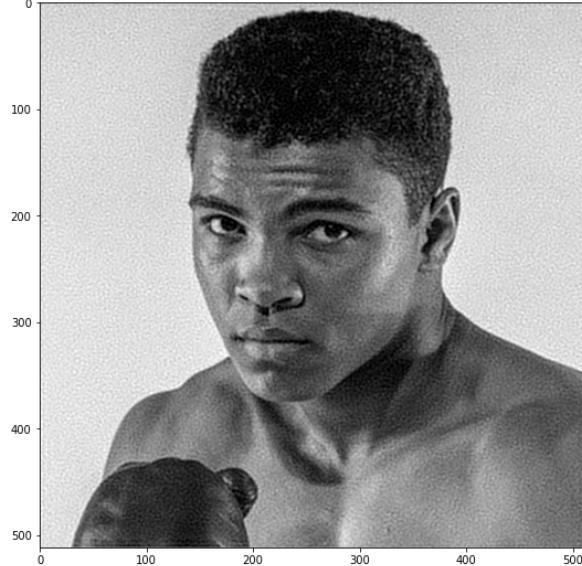
La funzione f da minimizzare è data dalla formula $f(x) = \frac{1}{2}||Ax - b||_2^2$, il cui gradiente ∇f è dato da $\nabla f(x) = A^T Ax - A^T b$.

Utilizzando il metodo del gradiente coniugato implementato dalla funzione `minimize` abbiamo calcolato la soluzione naïve.



(a) Immagini geometriche ripristinate con il Metodo del Gradiente Coniugato

immagine pugile ripristinata con metodo gradiente coniugato (PSNR: 30.359067103567714)



(b) Immagine fotografica ripristinata con il Metodo del Gradiente Coniugato

immagine giornale ripristinata con metodo gradiente coniugato (PSNR: 20.033233957558416)



(c) Immagine con testo ripristinata con il Metodo del Gradiente Coniugato

Figura 9: Immagini analizzate ripristinate con il Metodo del Gradiente Coniugato

```

1  from scipy.optimize import minimize
2  def f(x, B, labda = 0):
3      X = x.reshape(m,n)
4      res = 0.5*(np.linalg.norm(A(X, K)-B))**2 + 0.5*labda*(np.linalg.norm(X))**2
5      return np.sum(res)
6  def df(x, B, labda=0):
7      X = x.reshape(m,n)
8      res = AT(A(X, K)-B, K) + labda*X
9      RES = np.reshape(res, m*n)
10     return RES
11
12 func = lambda x: f(x, B) #o exec
13 grad_func = lambda x: df(x, B)
14 res = minimize(func, np.zeros(b.shape), method="CG", jac=grad_func, options={
15     'disp': True, 'maxiter':6})
16 RES = res.x.reshape(m,n)
17 PSNR = metrics.peak_signal_noise_ratio(X, RES) #alfabeto[4] solo lettere
18 plt.figure(figsize=(20,15))
19 plt.title('Ripristinata con metodo gradiente coniugato (PSNR: {PSNR})')
20 plt.imshow(RES, cmap='gray', vmin=0, vmax=1)

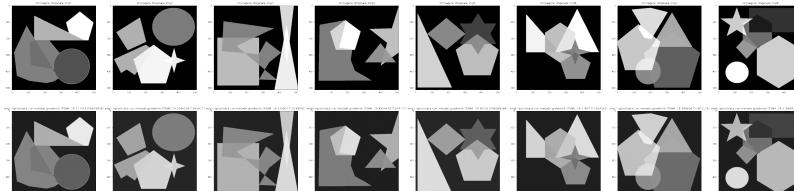
```

Figura 10: Codice Metodo del Gradiente Coniugato applicato ad una singola immagine

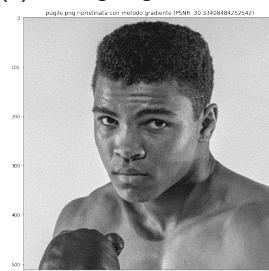
2.2 Metodo del Gradiente

Il metodo del gradiente è un algoritmo che calcola il vettore di minimo globale, ovvero: Un vettore x^* è un punto di minimo globale di $f(x)$ se $f(x^*) \leq f(x) \forall x \in R^n$.

Analogamente, un vettore x^* è un punto di minimo globale in senso stretto di $f(x)$ se $f(x^*) < f(x) \forall x \in R + nx \neq x^*$.



(a) Immagini geometriche ripristinate con il metodo del gradiente



(b) Immagine fotografica ripristinata con il metodo del gradiente



(c) Immagine con testo ripristinata con il metodo del gradiente

Figura 11: Immagini analizzate ripristinate con il Metodo del Gradiente

```

● ● ●
1 listaValGradG = []
2 listaErrRelG = []
3 listaPSNRG = []
4 listaValFunG = []
5
6 def next_step(x, grad, B, labda = 0):
    # backtracking procedure for the choice of the steplength
7     alpha=1.1
8     c1 = 0.25
9     p=-grad
10    j=0
11    jmax=10
12
    #condizioni che servono per soddisfare dei criteri di convergenza - condizioni di Wolfe
13    while ((f(x + alpha*p, B, labda) > f(x, B, labda)+c1*alpha*grad.T@p) and j <jmax):
14        alpha = alpha*0.5
15        j+=1
16    return alpha
17
18 def gradient_minimize(B, labda= 0, maxit=19, abstop = 1.e-6):
19     x_last =np.zeros(m*n)
20     k = 0
21     while (np.linalg.norm(df(x_last, B, labda))>abstop and k < maxit):
22         k=k+1
23         grad = df(x_last, B, labda)
24         step = next_step(x_last, grad, B, labda)
25
26         listaValFunG.append(f(x_last, B))
27         x_last=x_last-step*grad
28
29         listaValGradG.append(np.linalg.norm(grad.reshape(m,n), "fro"))
30         listaErrRelG.append(np.linalg.norm(x_last.reshape(m,n) - X, "fro")/np.
31             linalg.norm(X, "fro"))
32         listaPSNRG.append(metrics.peak_signal_noise_ratio(X, x_last.reshape(m,n)))
33
34     z_naive = gradient_minimize(B, labda = 0, maxit = 50)
    #maxit = 27. Messo maxit = 50 per studiare metodi e loro velocità
35     PSNR = metrics.peak_signal_noise_ratio(X, z_naive.reshape(m,n))
    #cambiare anche qui ogni volta nome immagine originale
36     plt.figure(figsize=(30,10))
37     plt.title(f'Ripristinata con metodo gradiente (PSNR: {PSNR})')
38     plt.imshow(z_naive.reshape(m,n), cmap="gray")
39

```

Figura 12: Codice Metodo del gradiente applicato ad una singola immagine

2.3 Metodo del gradiente e Metodo del gradiente coniugato a confronto

Notiamo che tra i due metodi che il primo ci da come risultato delle immagini con un PSNR definitivamente più alto rispetto al secondo, le immagini sono qualitativamente più simili alle immagini originali.

3 Osservazioni

I metodi di regolarizzazione rinunciano a trovare la soluzione esatta del problema del precedente problema di ottimizzazione, ma invece calcolano la soluzione di un problema leggermente diverso ma meglio condizionato. Quest'ultimo viene chiamato problema regolarizzato.

3.1 Metodo di Regolarizzazione di Tikhonov

Per ridurre gli effetti del rumore nella ricostruzione è necessario introdurre un termine di regolarizzazione di Tikhonov.

Si considera quindi il seguente problema di ottimizzazione:

Si deve risolvere $Ax_\epsilon = b_\epsilon$ con $b_\epsilon = b + \epsilon$. Al posto di risolvere direttamente il sistema lineare (se il sistema è quadrato) o di minimizzare la norma 2 del residuo (se il sistema è rettangolare) $\|Ax_\epsilon - b_\epsilon\|_2^2$,

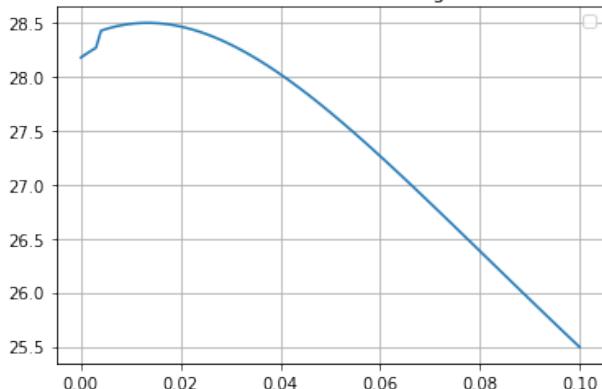
si aggiunge un vincolo di regolarità alla soluzione e si minimizza, ad esempio

$$\|Ax_\epsilon - b_\epsilon\|_2^2 + \gamma_\epsilon \|x_\epsilon\|_2^2$$

che rappresenta la **forma standard** della regolarizzazione di Tikhonov.

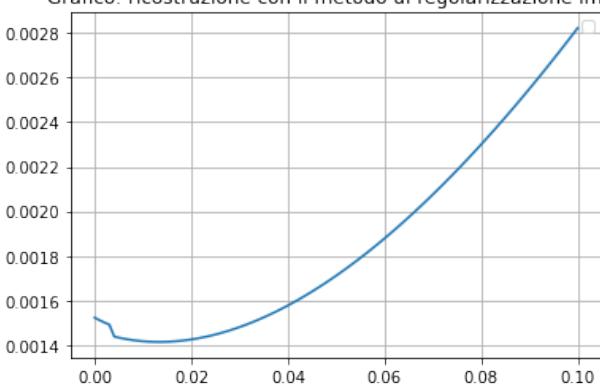
Analizziamo i grafici ottenuti cercando di ridurre il rumore nella ricostruzione delle immagini del dataset.

Grafico: ricostruzione con il metodo di regolarizzazione img1



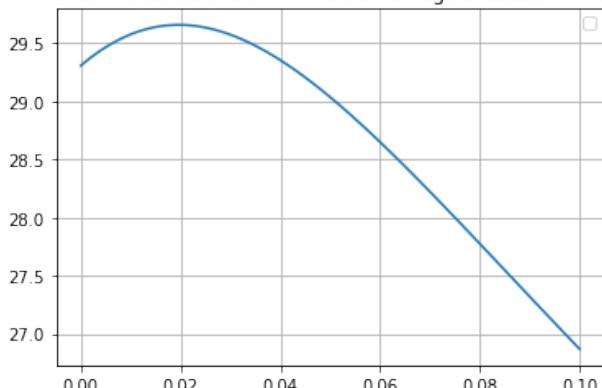
(a) PSNR di img1

Grafico: ricostruzione con il metodo di regolarizzazione img1



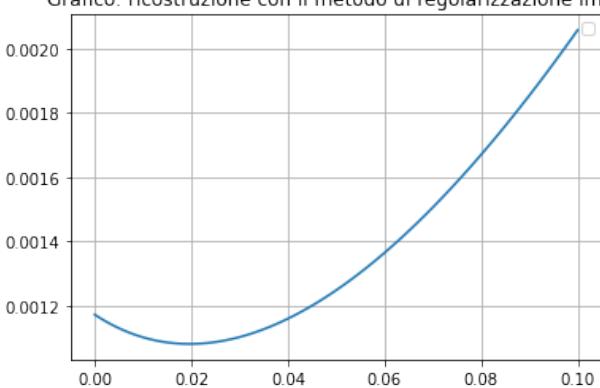
(b) MSE di img1

Grafico: ricostruzione con il metodo di regolarizzazione img2



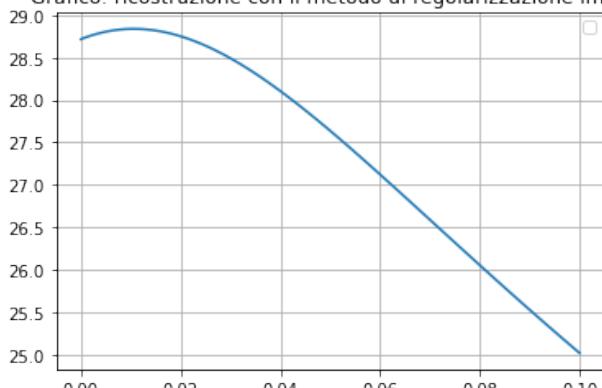
(c) PSNR di img2

Grafico: ricostruzione con il metodo di regolarizzazione img2



(d) MSE di img2

Grafico: ricostruzione con il metodo di regolarizzazione img3



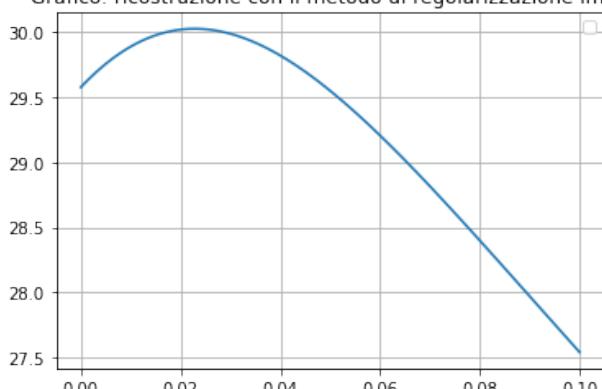
(e) PSNR di img3

Grafico: ricostruzione con il metodo di regolarizzazione img3



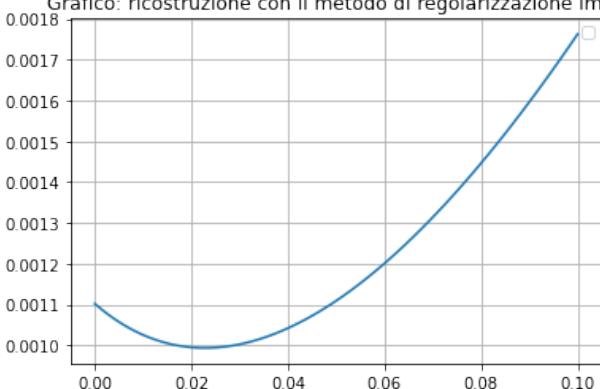
(f) MSE di img3

Grafico: ricostruzione con il metodo di regolarizzazione img4



(g) PSNR di img4

Grafico: ricostruzione con il metodo di regolarizzazione img4



(h) MSE di img4

Grafico: ricostruzione con il metodo di regolarizzazione img5

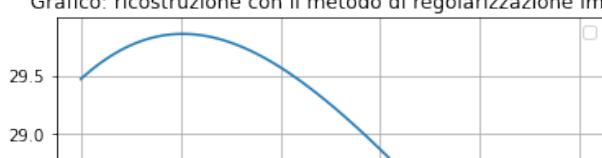
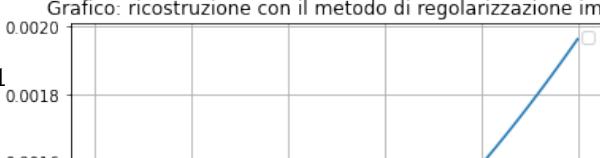


Grafico: ricostruzione con il metodo di regolarizzazione img5



(j) MSE di img5