

Relazione Progetto di Calcolo Numerico

Benatti Alice, Manuelli Matteo, Qayyum Shahbaz Ali

Gennaio 2022



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA
DIPARTIMENTO DI INFORMATICA - SCIENZA E INGEGNERIA

Indice

1 Presentazione del problema	3
1.1 Generazione dataset	3
1.2 Generazione Immagini Corrotte	4
1.3 Osservazioni	11
1.3.1 Analisi immagine geometrica	11
1.3.2 Analisi immagine fotografica	12
1.3.3 Analisi immagine con testo	13
2 Ricostruzione di un immagine rispetto una versione corrotta	14
2.1 Metodo del Gradiente Coniugato (naive)	14
2.2 Metodo del Gradiente	15
2.3 Metodo del Gradiente e Metodo del Gradiente Coniugato a confronto	17
3 Metodi di Regolarizzazione	17
3.1 Metodo di Regolarizzazione di Tikhonov	17
3.1.1 Immagini geometriche regolarizzate	18
3.1.2 Immagini fotografiche e con testo regolarizzate	20
4 Variazione Totale	21
5 Conclusioni	22

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 μ 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.

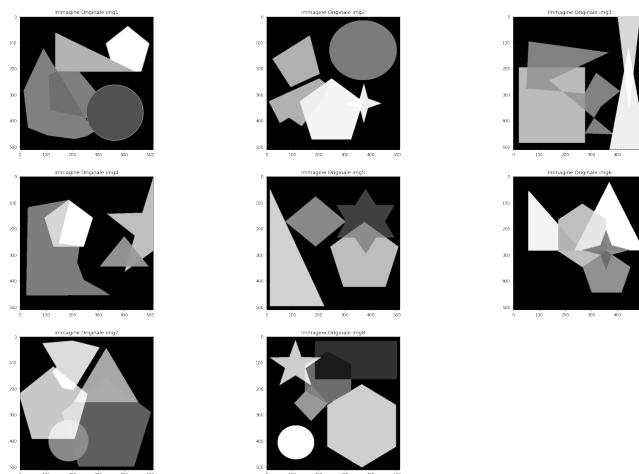


Figura 1: Immagini geometriche utilizzate

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.

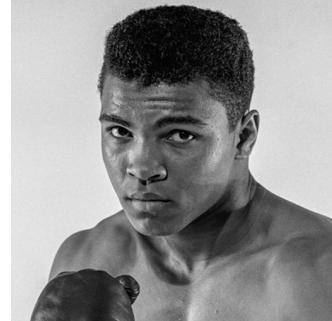
Le immagini selezionate sono le seguenti:

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

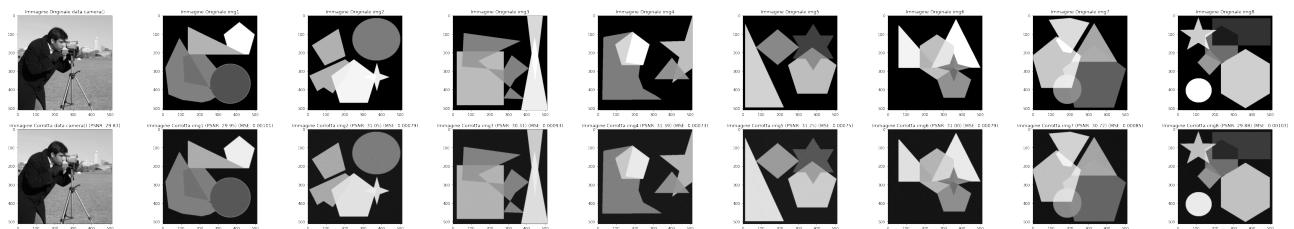
Figura 2: Immagini fotografiche analizzate

1.2 Generazione Immagini Corrotte

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

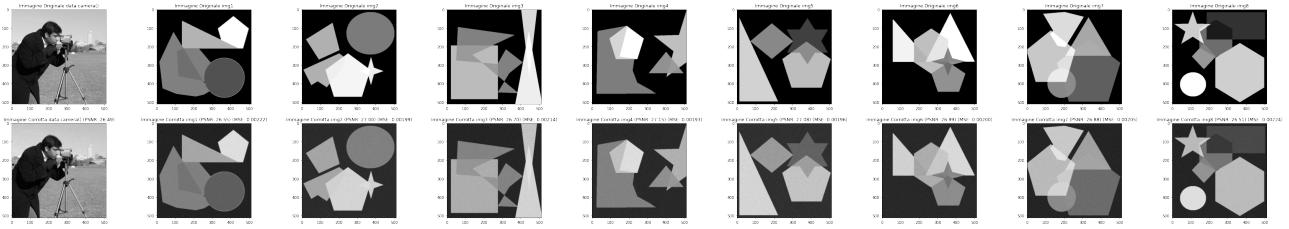
- $\sigma = 0.5$ dimensione 5×5
- $\sigma = 1$ dimensione 7×7
- $\sigma = 1.3$ dimensione 9×9

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



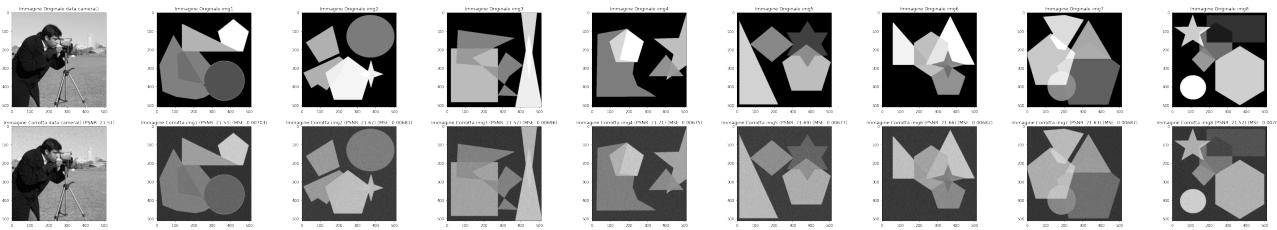
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	5	0.5	0.02	29.9455	0.00101263
img2.png	5	0.5	0.02	31.0498	0.000785276
img3.png	5	0.5	0.02	30.3053	0.000932107
img4.png	5	0.5	0.02	31.3947	0.000725313
img5.png	5	0.5	0.02	31.2456	0.00075065
img6.png	5	0.5	0.02	30.9971	0.00079486
img7.png	5	0.5	0.02	30.7187	0.000847478
img8.png	5	0.5	0.02	29.8802	0.00102797
pugile.png	5	0.5	0.02	32.3195	0.0005862
giornale.png	5	0.5	0.02	23.7827	0.00418534

Figura 3 & Tabella 1: Immagini corrotte con $\sigma = 0.5$ dimensione 5×5 e noise=0.02



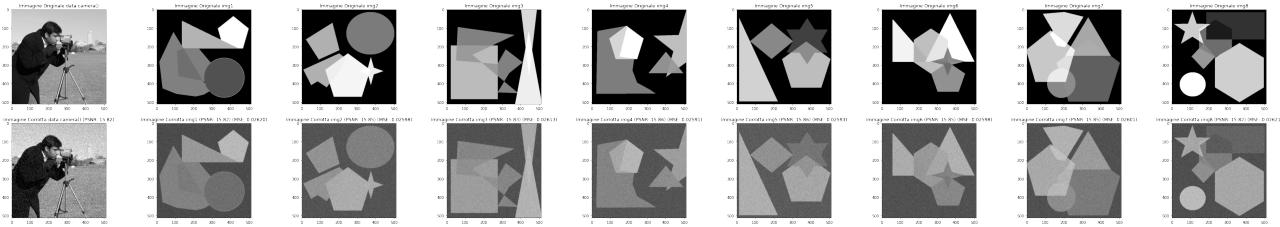
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	5	0.5	0.04	26.5452	0.00221553
img2.png	5	0.5	0.04	27.0007	0.00199492
img3.png	5	0.5	0.04	26.6982	0.00213887
img4.png	5	0.5	0.04	27.1549	0.00192537
img5.png	5	0.5	0.04	27.0846	0.00195677
img6.png	5	0.5	0.04	26.9876	0.00200098
img7.png	5	0.5	0.04	26.8806	0.00205086
img8.png	5	0.5	0.04	26.5063	0.00223545
pugile.png	5	0.5	0.04	27.4667	0.0017919
giornale.png	5	0.5	0.04	22.7045	0.0053647

Figura 4 & Tabella 2: Immagini corrotte con $\sigma = 0.5$ dimensione 5×5 e noise=0.04



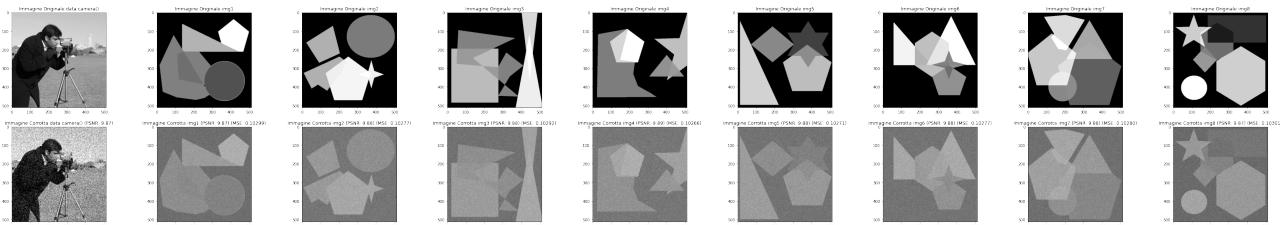
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	5	0.5	0.08	21.5332	0.00702558
img2.png	5	0.5	0.08	21.6702	0.00680731
img3.png	5	0.5	0.08	21.5736	0.00696048
img4.png	5	0.5	0.08	21.7089	0.00674704
img5.png	5	0.5	0.08	21.6929	0.00677191
img6.png	5	0.5	0.08	21.6626	0.00681931
img7.png	5	0.5	0.08	21.633	0.00686601
img8.png	5	0.5	0.08	21.519	0.00704858
pugile.png	5	0.5	0.08	21.7986	0.0066091
giornale.png	5	0.5	0.08	19.9035	0.0102246

Figura 5 & Tabella 3: Immagini corrotte con $\sigma = 0.5$ dimensione 5×5 e noise=0.08



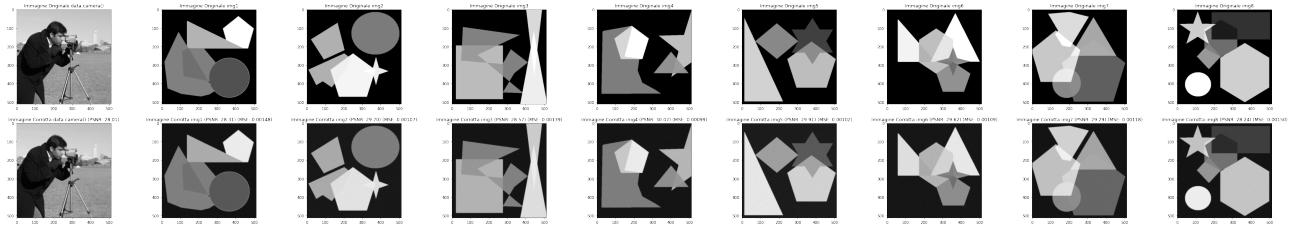
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	5	0.5	0.16	15.8162	0.0262048
img2.png	5	0.5	0.16	15.8543	0.025976
img3.png	5	0.5	0.16	15.8278	0.0261347
img4.png	5	0.5	0.16	15.8649	0.0259124
img5.png	5	0.5	0.16	15.8618	0.0259313
img6.png	5	0.5	0.16	15.8537	0.0259797
img7.png	5	0.5	0.16	15.849	0.0260077
img8.png	5	0.5	0.16	15.8161	0.0262051
pugile.png	5	0.5	0.16	15.89	0.0257632
giornale.png	5	0.5	0.16	15.3339	0.0292829

Figura 6 & Tabella 4: Immagini corrotte con $\sigma = 0.5$ dimensione 5×5 e noise=0.16



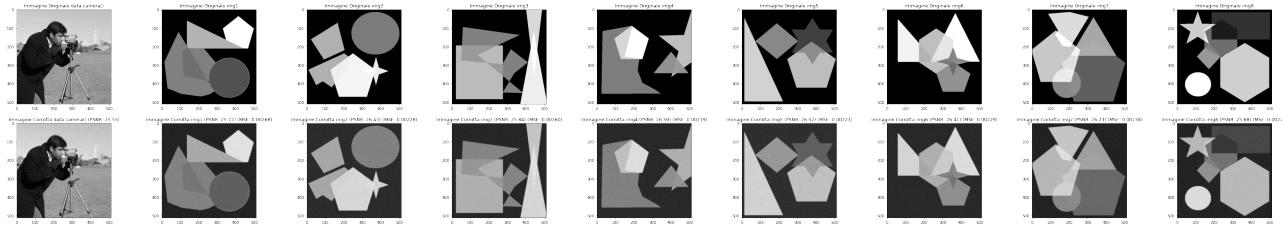
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	5	0.5	0.32	9.8721	0.102989
img2.png	5	0.5	0.32	9.88127	0.102772
img3.png	5	0.5	0.32	9.87516	0.102916
img4.png	5	0.5	0.32	9.8862	0.102655
img5.png	5	0.5	0.32	9.88405	0.102706
img6.png	5	0.5	0.32	9.88154	0.102765
img7.png	5	0.5	0.32	9.88	0.102802
img8.png	5	0.5	0.32	9.87102	0.103014
pugile.png	5	0.5	0.32	9.89115	0.102538
giornale.png	5	0.5	0.32	9.73853	0.10620

Figura 7 & Tabella 5: Immagini corrotte con $\sigma = 0.5$ dimensione 5×5 e noise=0.32



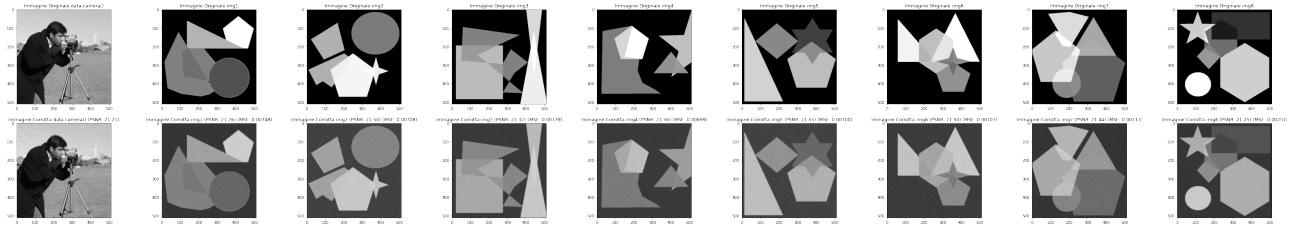
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	7	1	0.02	28.3054	0.00147726
img2.png	7	1	0.02	29.7038	0.00107057
img3.png	7	1	0.02	28.5689	0.00139029
img4.png	7	1	0.02	30.0656	0.000985011
img5.png	7	1	0.02	29.9064	0.0010218
img6.png	7	1	0.02	29.6228	0.00109074
img7.png	7	1	0.02	29.2918	0.00117711
img8.png	7	1	0.02	28.2434	0.00149851
pugile.png	7	1	0.02	30.9284	0.000807527
giornale.png	7	1	0.02	20.9194	0.0080921

Figura 8 & Tabella 6: Immagini corrotte con $\sigma = 1$ dimensione 7×7 e noise=0.02



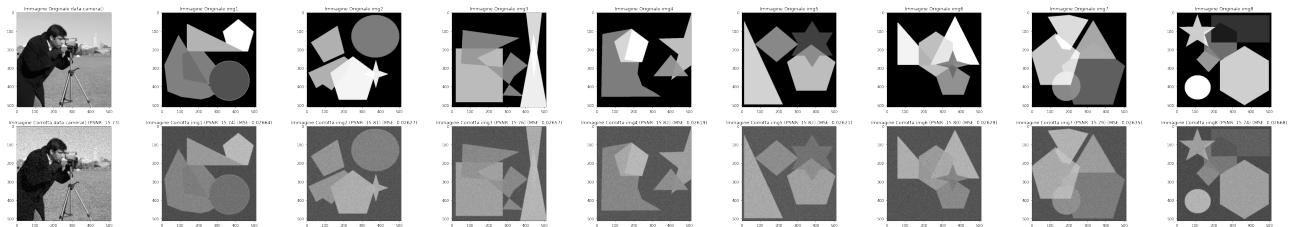
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	7	1	0.04	25.7196	0.00267942
img2.png	7	1	0.04	26.4282	0.00227604
img3.png	7	1	0.04	25.8439	0.00260379
img4.png	7	1	0.04	26.5898	0.0021929
img5.png	7	1	0.04	26.5209	0.00222797
img6.png	7	1	0.04	26.4067	0.00228732
img7.png	7	1	0.04	26.2277	0.00238357
img8.png	7	1	0.04	25.6828	0.00270224
pugile.png	7	1	0.04	26.9696	0.00200928
giornale.png	7	1	0.04	20.3045	0.0093227

Figura 9 & Tabella 7: Immagini corrotte con $\sigma = 1$ dimensione 7×7 e noise=0.04



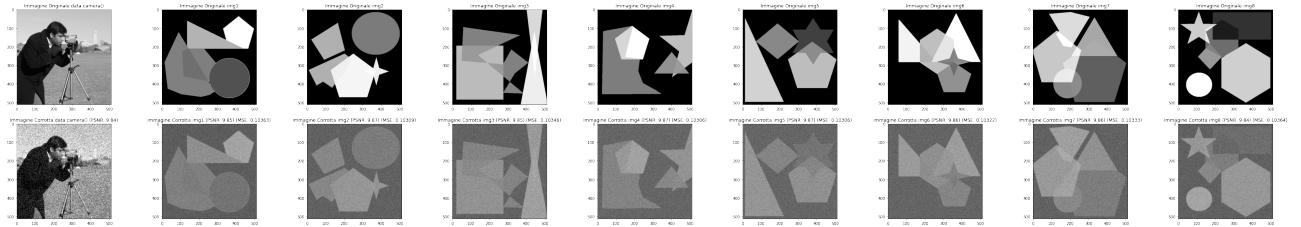
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	7	1	0.08	21.2586	0.00748406
img2.png	7	1	0.08	21.5025	0.00707543
img3.png	7	1	0.08	21.3156	0.0073866
img4.png	7	1	0.08	21.5582	0.00698523
img5.png	7	1	0.08	21.5463	0.00700433
img6.png	7	1	0.08	21.5043	0.00707252
img7.png	7	1	0.08	21.4429	0.00717307
img8.png	7	1	0.08	21.2464	0.00750518
pugile.png	7	1	0.08	21.6733	0.00680245
giornale.png	7	1	0.08	18.5164	0.0140722

Figura 10 & Tabella 8: Immagini corrotte con $\sigma = 1$ dimensione 7×7 e noise=0.08



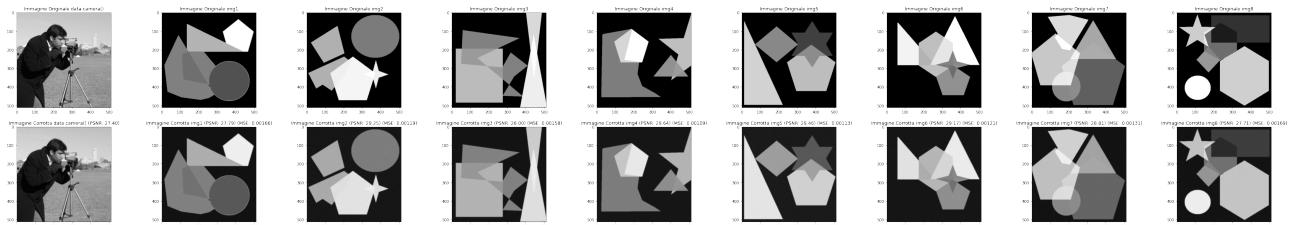
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	7	1	0.16	15.7444	0.0266415
img2.png	7	1	0.16	15.8054	0.0262698
img3.png	7	1	0.16	15.7565	0.0265673
img4.png	7	1	0.16	15.8194	0.0261856
img5.png	7	1	0.16	15.8151	0.0262111
img6.png	7	1	0.16	15.8029	0.0262853
img7.png	7	1	0.16	15.7926	0.0263473
img8.png	7	1	0.16	15.7385	0.0266776
pugile.png	7	1	0.16	15.8496	0.0260039
giornale.png	7	1	0.16	14.7736	0.0333153

Figura 11 & Tabella 9: Immagini corrotte con $\sigma = 1$ dimensione 7×7 e noise=0.16



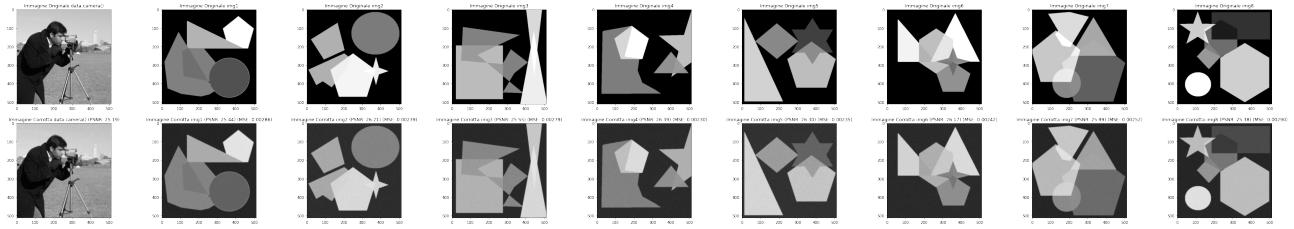
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	7	1	0.32	9.84505	0.103632
img2.png	7	1	0.32	9.86765	0.103094
img3.png	7	1	0.32	9.85214	0.103463
img4.png	7	1	0.32	9.86913	0.103059
img5.png	7	1	0.32	9.869	0.103062
img6.png	7	1	0.32	9.86252	0.103216
img7.png	7	1	0.32	9.85775	0.10333
img8.png	7	1	0.32	9.84472	0.10364
pugile.png	7	1	0.32	9.87496	0.102921
giornale.png	7	1	0.32	9.57992	0.110156

Figura 12 & Tabella 10: Immagini corrotte con $\sigma = 1$ dimensione 7×7 e noise=0.32



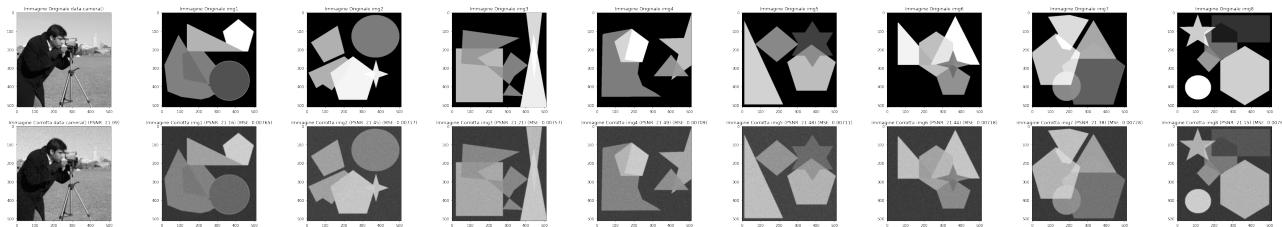
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	9	1.3	0.02	27.7886	0.00166394
img2.png	9	1.3	0.02	29.2539	0.00118745
img3.png	9	1.3	0.02	28.0014	0.00158437
img4.png	9	1.3	0.02	29.6382	0.00108688
img5.png	9	1.3	0.02	29.4566	0.00113329
img6.png	9	1.3	0.02	29.1727	0.00120986
img7.png	9	1.3	0.02	28.8132	0.00131424
img8.png	9	1.3	0.02	27.7142	0.00169269
pugile.png	9	1.3	0.02	30.3576	0.0009209
giornale.png	9	1.3	0.02	20.0318	0.00992697

Figura 13 & Tabella 11: Immagini corrotte con $\sigma = 1.3$ dimensione 9×9 e noise=0.02



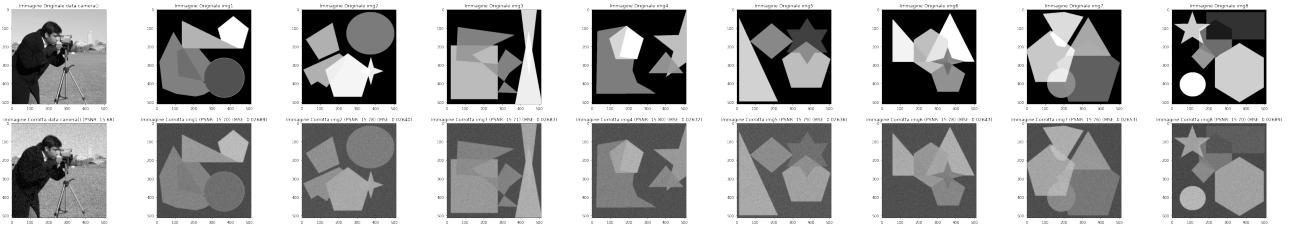
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	9	1.3	0.04	25.4375	0.00285925
img2.png	9	1.3	0.04	26.2124	0.00239202
img3.png	9	1.3	0.04	25.5474	0.0027878
img4.png	9	1.3	0.04	26.389	0.00229666
img5.png	9	1.3	0.04	26.2972	0.00234571
img6.png	9	1.3	0.04	26.1696	0.00241568
img7.png	9	1.3	0.04	25.9907	0.00251728
img8.png	9	1.3	0.04	25.3782	0.00289853
pugile.png	9	1.3	0.04	26.7366	0.00212
giornale.png	9	1.3	0.04	19.5402	0.0111168

Figura 14 & Tabella 12: Immagini corrotte con $\sigma = 1.3$ dimensione 9×9 e noise=0.04



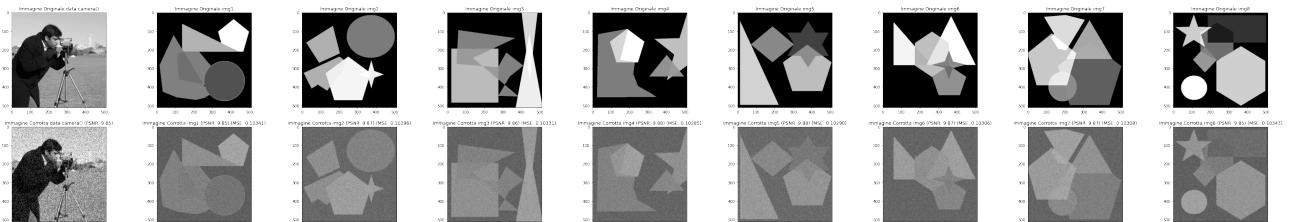
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	9	1.3	0.08	21.1615	0.0076534
img2.png	9	1.3	0.08	21.446	0.00716802
img3.png	9	1.3	0.08	21.2099	0.00756858
img4.png	9	1.3	0.08	21.4931	0.00709073
img5.png	9	1.3	0.08	21.4791	0.0071136
img6.png	9	1.3	0.08	21.436	0.0071845
img7.png	9	1.3	0.08	21.3758	0.00728476
img8.png	9	1.3	0.08	21.1529	0.00766845
pugile.png	9	1.3	0.08	21.6072	0.0069068
giornale.png	9	1.3	0.08	17.9887	0.0158901

Figura 15 & Tabella 13: Immagini corrotte con $\sigma = 1.3$ dimensione 9×9 e noise=0.08



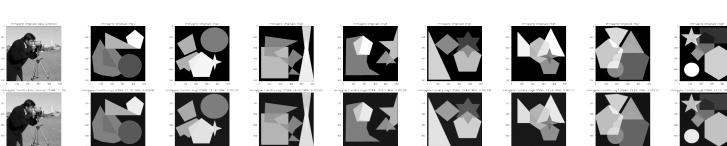
Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	9	1.3	0.16	15.7044	0.026888
img2.png	9	1.3	0.16	15.7843	0.0263979
img3.png	9	1.3	0.16	15.7146	0.0268251
img4.png	9	1.3	0.16	15.7966	0.026323
img5.png	9	1.3	0.16	15.7911	0.0263568
img6.png	9	1.3	0.16	15.7794	0.0264277
img7.png	9	1.3	0.16	15.7632	0.0265265
img8.png	9	1.3	0.16	15.7044	0.0268884
pugile.png	9	1.3	0.16	15.8334	0.0261013
giornale.png	9	1.3	0.16	14.5301	0.0352365

Figura 16 & Tabella 14: Immagini corrotte con $\sigma = 1.3$ dimensione 9×9 e noise=0.16



Nome Img	DimKer	Sigma	Noise Dev	PSNR	MSE
img1.png	9	1.3	0.32	9.85454	0.103406
img2.png	9	1.3	0.32	9.87346	0.102957
img3.png	9	1.3	0.32	9.85875	0.103306
img4.png	9	1.3	0.32	9.87778	0.102854
img5.png	9	1.3	0.32	9.87599	0.102897
img6.png	9	1.3	0.32	9.86911	0.10306
img7.png	9	1.3	0.32	9.8681	0.103084
img8.png	9	1.3	0.32	9.85336	0.103434
pugile.png	9	1.3	0.32	9.88385	0.10271
giornale.png	9	1.3	0.32	9.52564	0.111541

Figura 17 & Tabella 15: Immagini corrotte con $\sigma = 1.3$ dimensione 9×9 e noise=0.32



Istanza del Problema	
Media PSNR	28.721034907424552
Media MSE	0.0013616018508439164
Dev. Std. PSNR	0.7263716130347668
Dev. Std. MSE	0.00032199072780323465

Figura 18 & Tabella 16: Immagini corrotte

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..

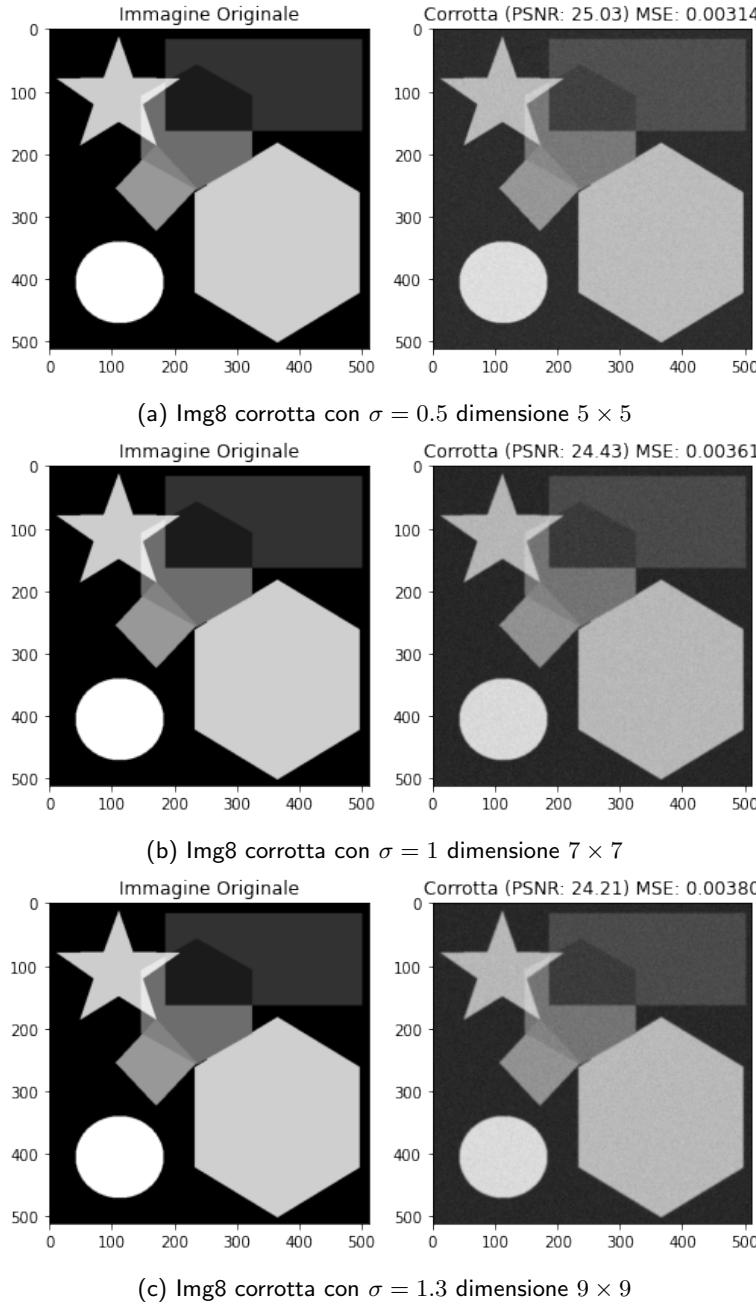


Figura 19: Immagine geometrica corrotta al variare di

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.

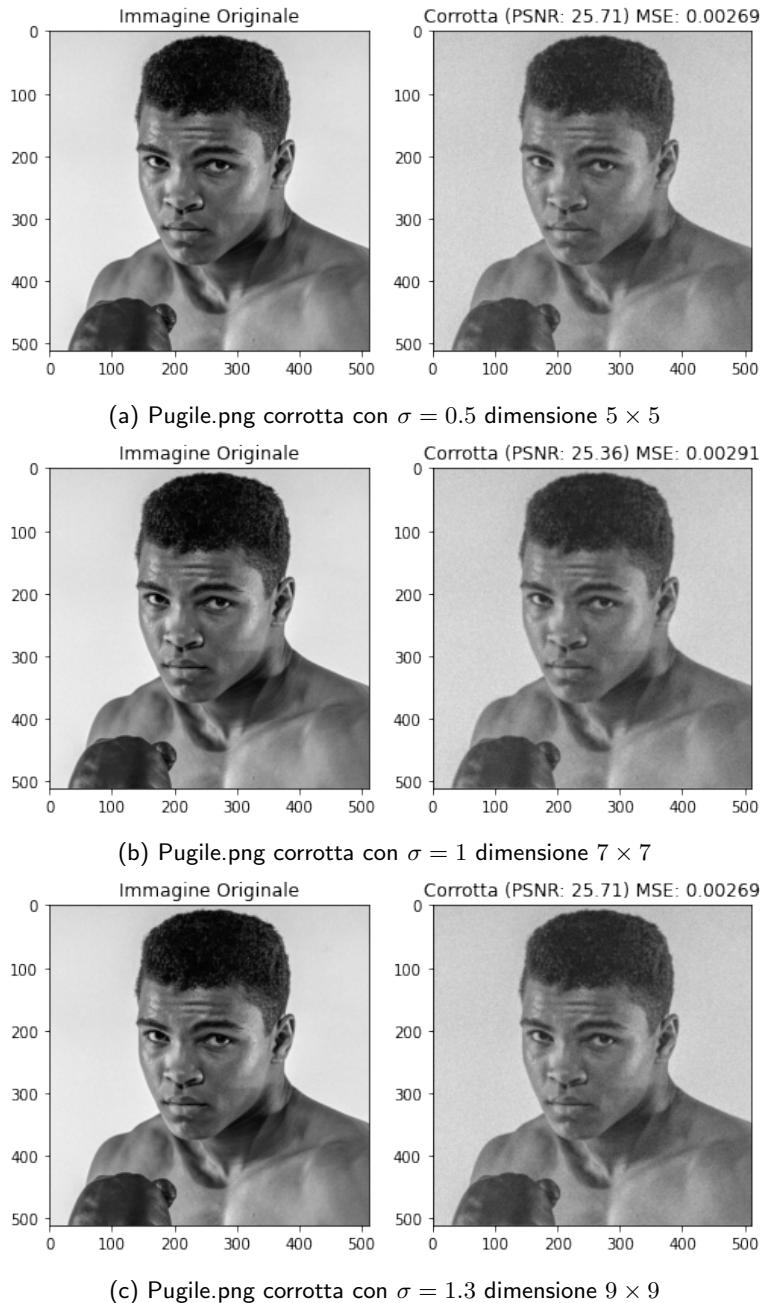
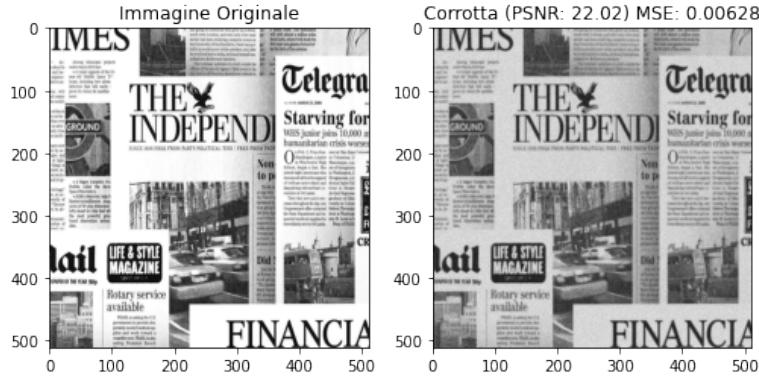


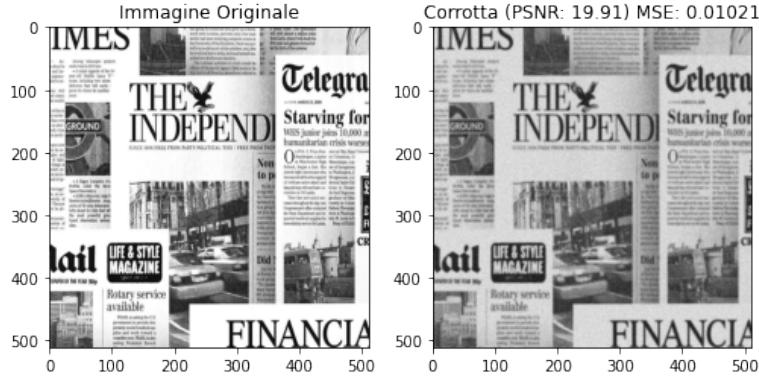
Figura 20: Immagine fotografica corrotta al variare di σ

1.3.3 Analisi immagine con testo

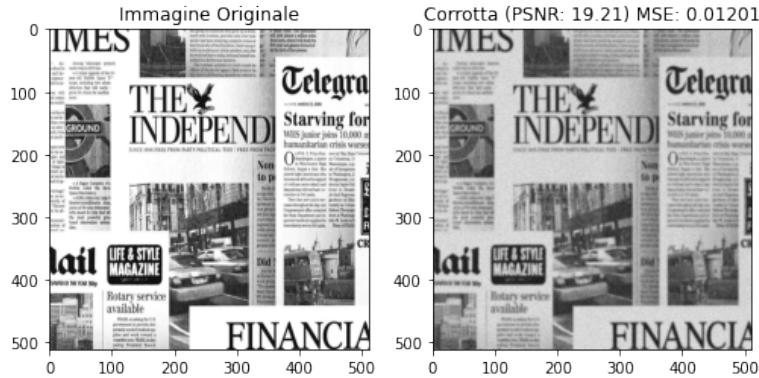
In questa immagine abbiamo una raccolta di prime pagine di giornale che ci permettono di osservare e valutare meglio la differenza tra l'immagine originale e la versione corrotta, per esempio con $\sigma = 0,5$ (vedi immagine 21a) otteniamo un'immagine con del testo ancora leggibile sebbene meno nitida, la difficoltà inizia ad essere maggiore invece con $\sigma = 1$ (vedi immagine 21b) dove le scritte più piccole diventano quasi illeggibili, con $\sigma = 1.3$ (vedi immagine 21c) 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 21: 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.

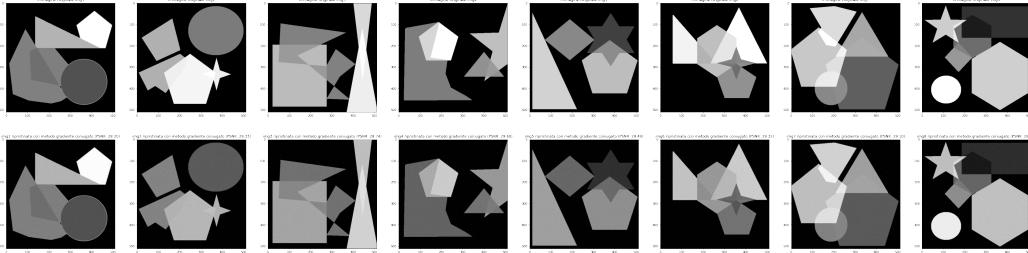
Abbiamo mostrato questi due grafici sovrastanti perché si evince che all'aumentare del numero delle iterazioni l'immagine ricavata si allontana dalla sua versione originale poiché è presente una deviazione.

2.1 Metodo del Gradiente Coniugato (naive)

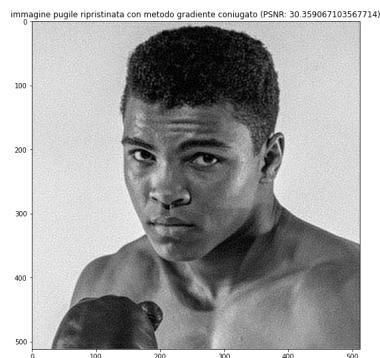
Il metodo del gradiente coniugato è un algoritmo per la risoluzione numerica di un sistema lineare la cui matrice sia simmetrica e definita positiva e consente di risolvere il sistema in un numero di iterazioni che è al massimo n .

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



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



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

Figura 22: 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(f'Ripristinata con metodo gradiente coniugato (PSNR: {PSNR})')
20 plt.imshow(RES, cmap='gray', vmin=0, vmax=1)

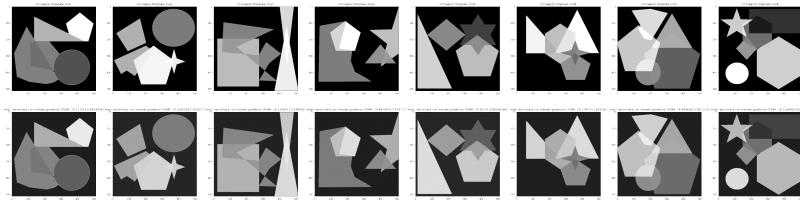
```

Figura 23: Codice Metodo del Gradiente Coniugato applicato ad una singola immagine

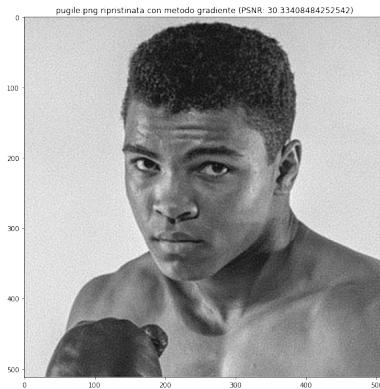
2.2 Metodo del Gradiente

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 24: Immagini analizzate ripristinate con il Metodo del Gradiente

```

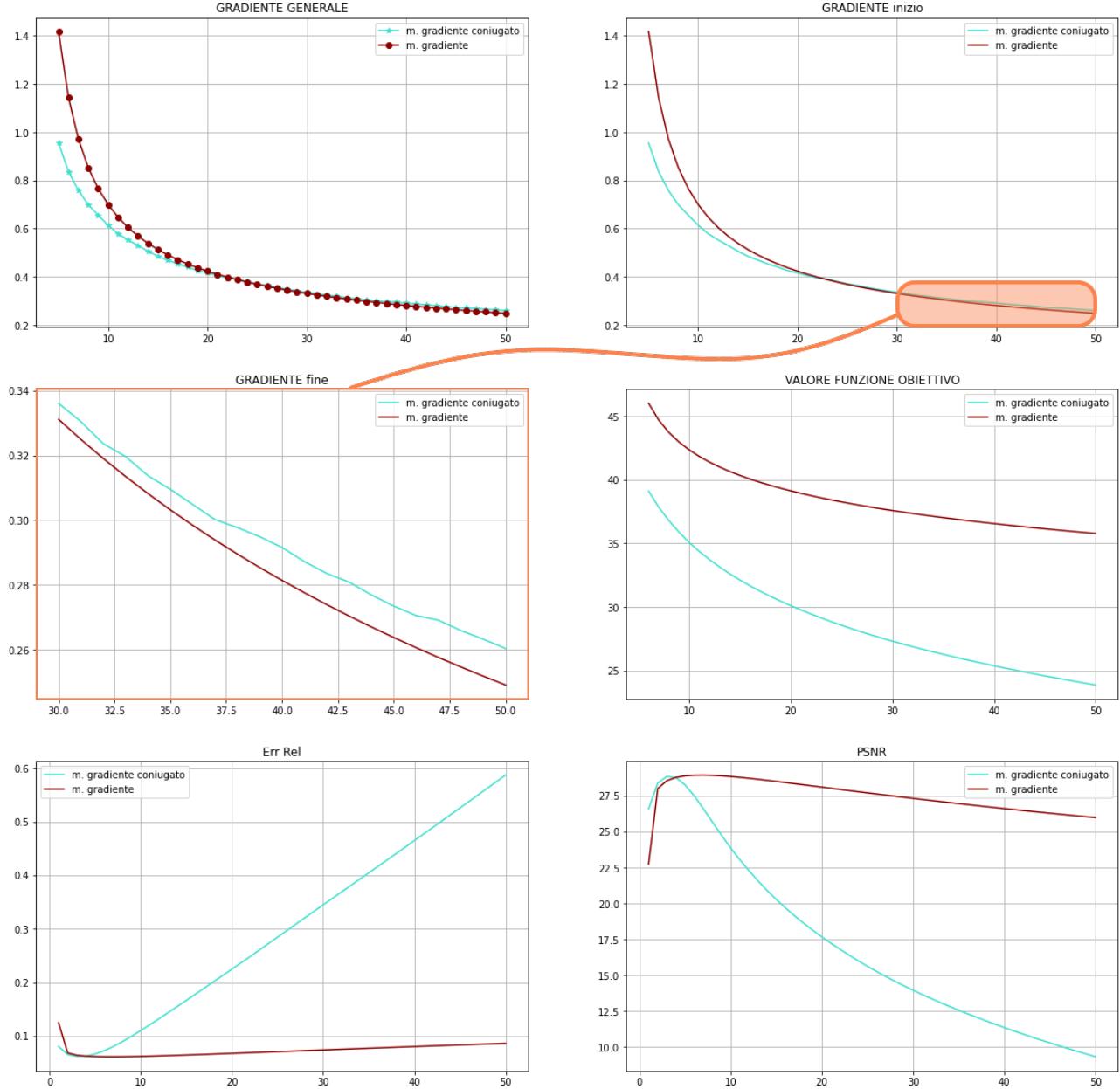
● ● ●
1 listaValGradG = []
2 listaErrRelG = []
3 listaPSNRG = []
4 listaValFunG = []
5
6 def next_step(x, grad, B, labda = 0):
    # backtracking procedure for the choice of the steplength
7     alpha=1.1
8     c1 = 0.25
9     p=-grad
10    j=0
11    jmax=10
12
13    #condizioni che servono per soddisfare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

```

Figura 25: 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 dà come risultato delle immagini con un PSNR definitivamente più alto rispetto al secondo, le immagini sono qualitativamente più simili alle immagini originali.



3 Metodi di Regolarizzazione

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.

3.1.1 Immagini geometriche regolarizzate

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

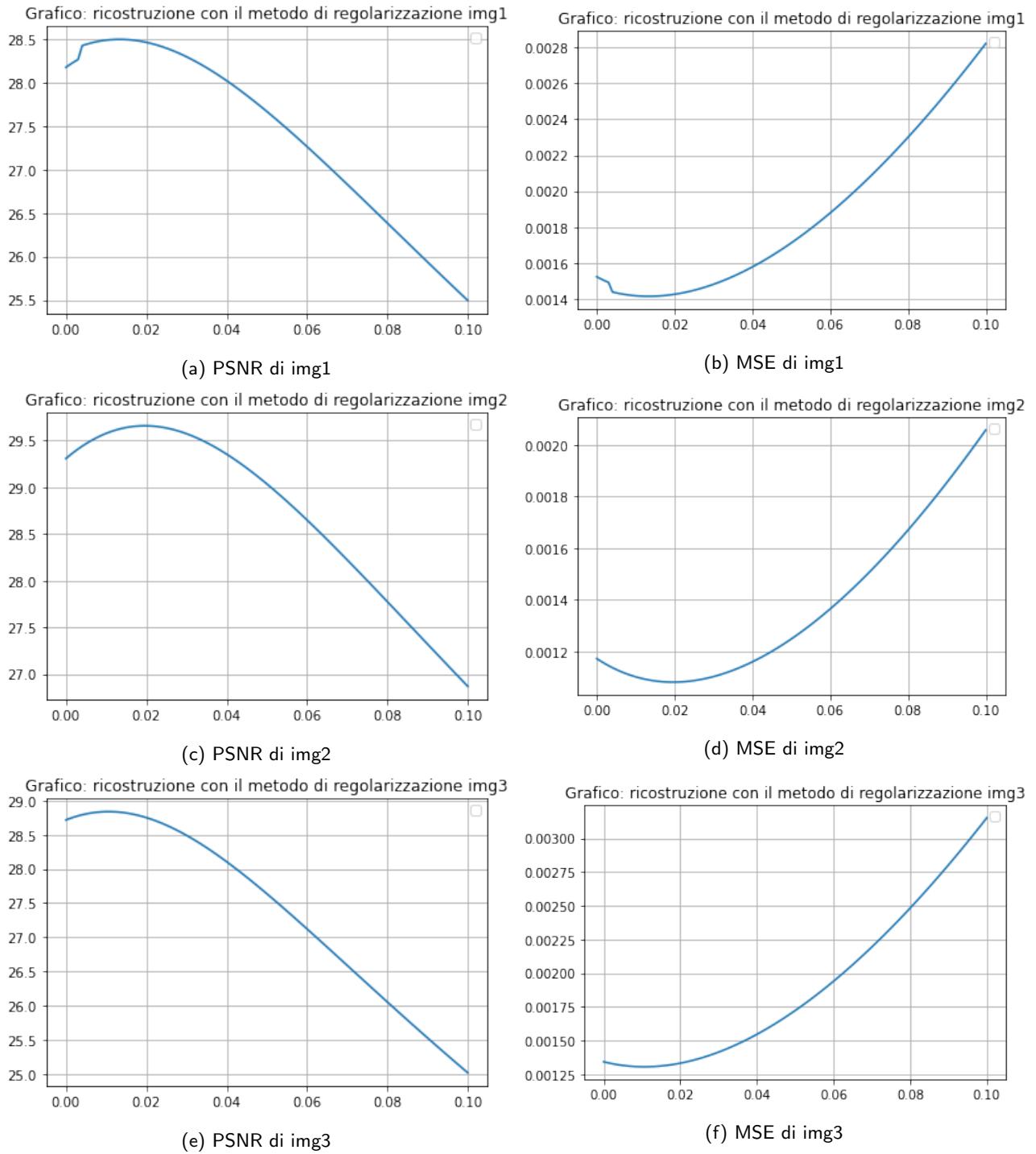
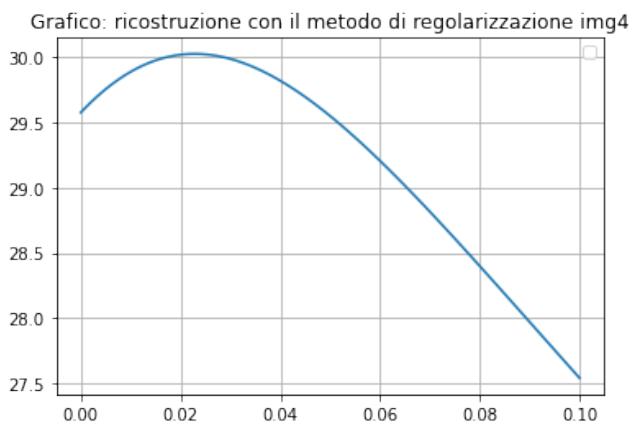


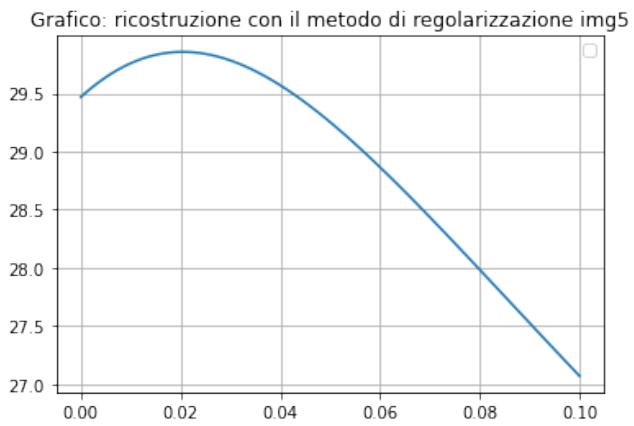
Figura 26: Grafici andamento PSNR e MSE nella ricostruzione delle immagini con il metodo di regolarizzazione



(g) PSNR di img4



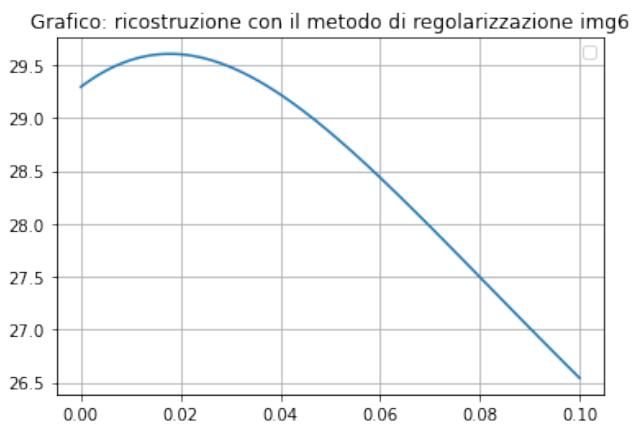
(h) MSE di img4



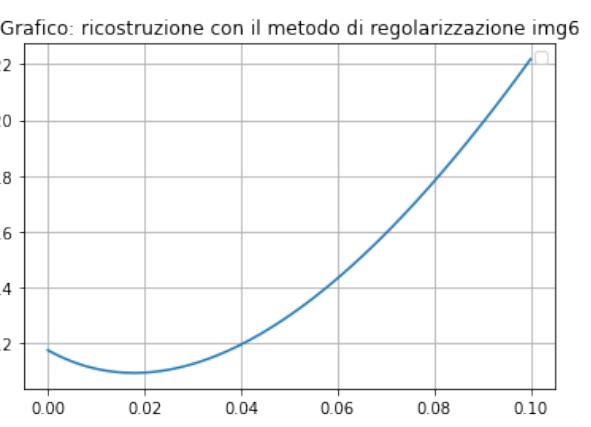
(i) PSNR di img5



(j) MSE di img5



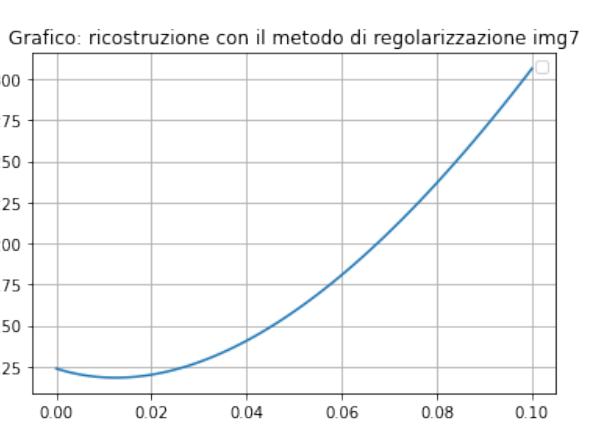
(k) PSNR di img6



(l) MSE di img6



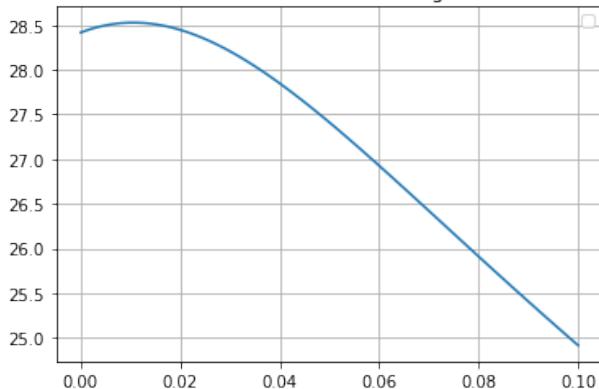
(m) PSNR di img7



(n) MSE di img7

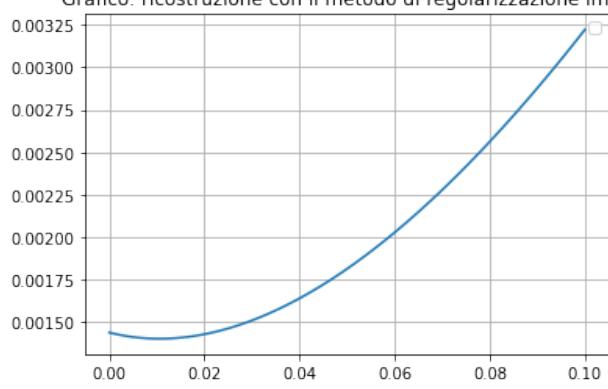
Figura 26: Grafici andamento PSNR e MSE nella ricostruzione delle immagini con il metodo di regolarizzazione

Grafico: ricostruzione con il metodo di regolarizzazione img8



(o) PSNR di img8

Grafico: ricostruzione con il metodo di regolarizzazione img8



(p) MSE di img8

Figura 26: Grafici andamento PSNR e MSE nella ricostruzione delle immagini con il metodo di regolarizzazione

3.1.2 Immagini fotografiche e con testo regolarizzate

Per quanto riguarda le immagini fotografiche, abbiamo riscontrato un curioso errore.

```

Warning: Desired error not necessarily achieved due to precision loss.
Current function value: 52.745880
Iterations: 0
Function evaluations: 82
Gradient evaluations: 70
lambda: 1e-07 PSNR: 4.002496960607354 MSE: 0.3978783460453535
Warning: Desired error not necessarily achieved due to precision loss.
Current function value: 105.423313
Iterations: 0
Function evaluations: 85
Gradient evaluations: 73
lambda: 0.0010102000000000002 PSNR: 4.002496960607354 MSE: 0.3978783460453535
Warning: Desired error not necessarily achieved due to precision loss.
Current function value: 158.100746
Iterations: 0
Function evaluations: 77
Gradient evaluations: 65
lambda: 0.0020203 PSNR: 4.002496960607354 MSE: 0.3978783460453535
Warning: Desired error not necessarily achieved due to precision loss.
Current function value: 210.778179
Iterations: 0
Function evaluations: 88
Gradient evaluations: 76
lambda: 0.0030304000000000004 PSNR: 4.002496960607354 MSE: 0.3978783460453535

```

```

Warning: Desired error not necessarily achieved due to precision loss.
Current function value: 263.455611
Iterations: 0
Function evaluations: 79
Gradient evaluations: 67
lambda: 0.004040500000000001 PSNR: 4.002496960607354 MSE: 0.3978783460453535
Warning: Desired error not necessarily achieved due to precision loss.
Current function value: 316.133044
Iterations: 0
Function evaluations: 74
Gradient evaluations: 62
lambda: 0.005050600000000001 PSNR: 4.002496960607354 MSE: 0.3978783460453535
Warning: Desired error not necessarily achieved due to precision loss.
Current function value: 368.810477
Iterations: 0
Function evaluations: 70
Gradient evaluations: 58
lambda: 0.006060700000000001 PSNR: 4.002496960607354 MSE: 0.3978783460453535

```

Figura 27: Errore immagini fotografiche

Possiamo affermare che l'incremento è talmente impercettibile che non viene colto dal calcolatore poiché inferiore alla sua precisione macchina, con la regolarizzazione si ottengono comunque risultati migliori.

4 Variazione Totale

Tramite un algoritmo possiamo recuperare immagini sfocate basandoci sulla Variazione totale partendo da una Blurring Point-Spread function di un'immagine.

La variazione totale è definita dalla seguente formula:

$$TV(u) = \sum_i^n \sum_j^m \sqrt{||\nabla u(i,j)||_2^2 + \epsilon^2}$$

Per calcolare il gradiente dell'immagine ∇u usiamo la funzione `np.gradient` che approssima la derivata per ogni pixel calcolando la differenza tra pixel adiacenti.

I risultati sono due immagini della stessa dimensione dell'immagine in input, una che rappresenta il valore della derivata orizzontale dx e l'altra della derivata verticale dy . Il gradiente dell'immagine nel punto (i,j) è quindi un vettore di due componenti, uno orizzontale contenuto in dx e uno verticale in dy .

$$x^* = \arg \min_x \frac{1}{2} \|Ax - b\|_2^2 + \lambda TV(u)$$

il cui gradiente ∇f è dato da:

$$\nabla f(x) = (A^T Ax - A^T b) + \lambda \nabla TV(x)$$

Variazione totale dell'immagine fotografica:

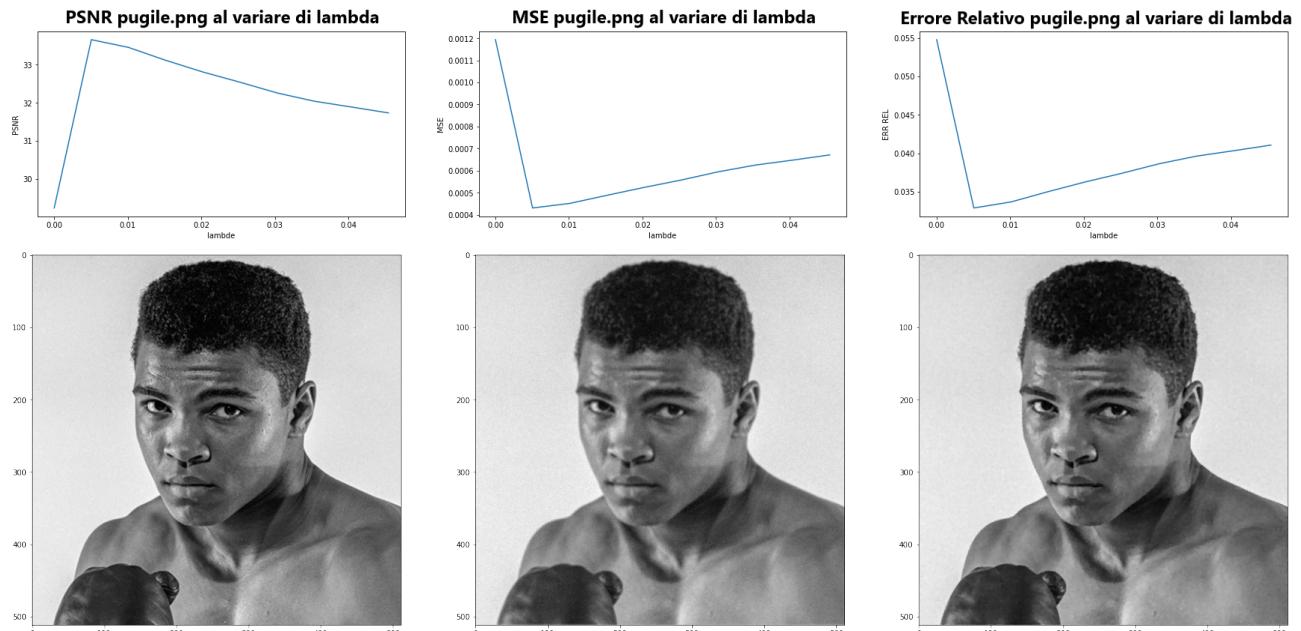


Figura 28: Immagine Originale, Immagine Corrotta, Immagine Ricostruita e i relativi grafici di variazione PSNR, MSE ed Errore Relativo (da sinistra a destra)

Variazione totale dell'immagine con testo:

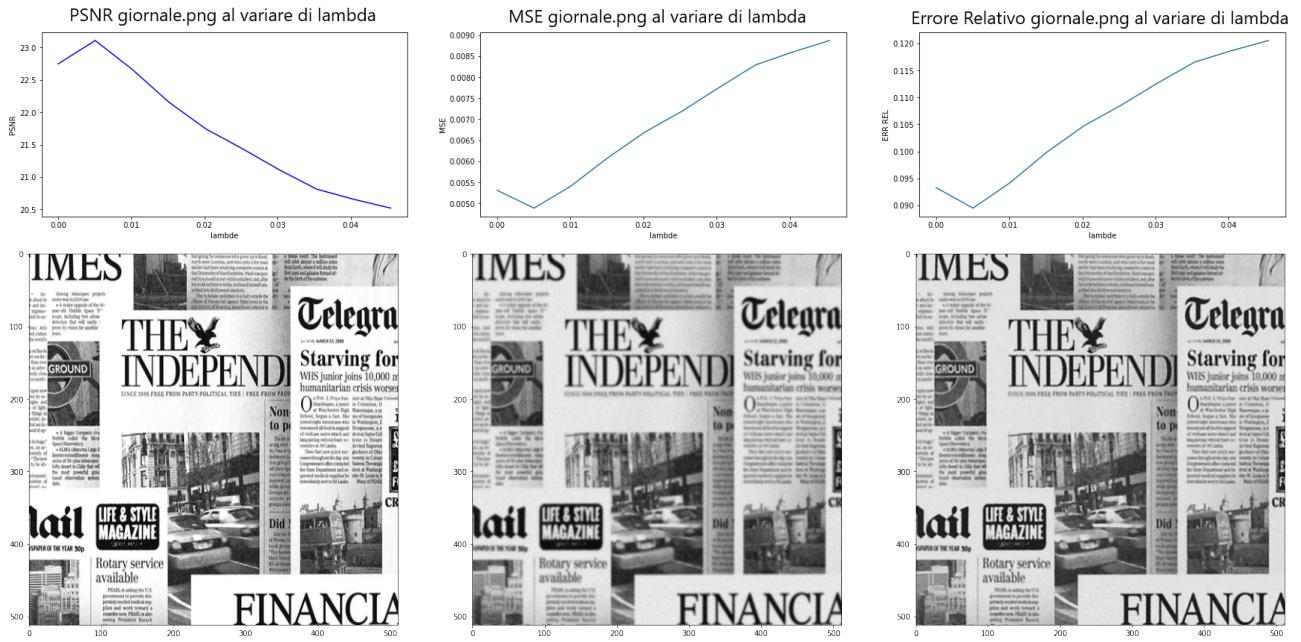


Figura 29: Immagine Originale, Immagine Corrotta, Immagine Ricostruita e i relativi grafici di variazione PSNR, MSE ed Errore Relativo (da sinistra a destra)

5 Conclusioni