Alma Mater Studiorum · Università di Bologna

facoltà di scienze matematiche, fisiche e naturali

Corso di Laurea Magistrale in Matematica

-Indirizzo Applicativo-

Interfaccia grafica per software MATLAB per la ricostruzione di immagini

Tesi di Laurea in Analisi Numerica e Laboratorio

Relatore:
Chiar.ma Prof.ssa
Elena Loli Piccolomini

Presentata da:
Teresa Beltrani

Prima Sessione
Anno Accademico 2010/2011

Indice

Elenco delle figure

Introduzione

Lo scopo di questa tesi è la realizzazione di un'interfaccia grafica per software Matlab per l'utilizzo di un pacchetto di algoritmi per la ricostruzione di immagini digitali.

Il problema della ricostruzione di immagini appartiene alla classe dei problemi inversi: partendo da un segnale di output misurato, si vuole determinare un input sconosciuto. Il modello matematico che descrive il processo di acquisizione e registrazione di un'immagine trasforma il problema nella risoluzione di un sistema lineare di grandi dimensioni. La matrice di questo sistema tuttavia è mal condizionata, è perció necessario ricorrere a tecniche alternative. Viene introdotto il metodo di regolarizzazione: si vuole cosí determinare la soluzione di un problema di minimo vincolato.

Nel primo capitolo vengono introdotti i concetti fondamentali di ottimizzazione non vincolata e vincolata. Viene descritto inoltre il metodo del gradiente proiettato ed in particolare la proiezione a due metriche.

Nel secondo capitolo è esaminato il processo di formazione e registrazione di un'immagine digitale. Viene poi presentato un modello lineare per la descrizione del fenomeno di blurring, le cui principali componenti sono la Point Spread Function (PSF), il rumore e le condizioni al contorno. Il problema di ricostruzione di un'immagine digitale diventa un problema di risoluzione di un sistema lineare. Si vedrá che la matrice del sistema gode di proprietá tali da semplificare i calcoli e diminuire il costo computazionale.

Nel terzo capitolo si studia il problema di regolarizzazione per la ricostruzione di un'immagine digitale. In particolare si vedono due metodi: il Fast

Projected Quasi-Newton (QNP) e il Projected Newton Coniugate Gradient (PNCG).

Nell'ultimo capitolo viene presentata l'interfaccia grafica sviluppata in Matlab 7.0.4. Sono illustrati i passi principali per il suo utilizzo: il caricamento dell'immagine, la scelta del metodo, le opzioni ed il salvataggio dei dati. Sono stati forniti anche alcuni esempi.

Grazie all'interfaccia, è possibile scegliere ed eseguire gli algoritmi, visualizzare le immagini di output e le informazioni relative al metodo in un'apposita finestra ed infine salvare i risultati in un Mat-file.

Capitolo 1

Ottimizzazione

In questo capitolo sono introdotti i concetti e le definizioni fondamentali dell'ottimizzazione, la branca della matematica applicata che studia teoria e metodi per la ricerca dei punti di massimo e minimo di una funzione matematica.

Problemi d'ottimizzazione sorgono in tutte le aree delle scienze, dell'ingegneria e del business. In questi problemi si richiede di ottimizzare una funzione di costo o di efficienza: tra tutte le soluzioni possibili si cerca quella che meglio soddisfa un determinato obiettivo. Nella maggior parte dei problemi di ottimizzazione la soluzione deve soddisfare delle condizioni, detti vincoli. Per approfondire l'argomento si consulti [1], [2] e [3].

Un problema di ottimizzazione puó essere espresso matematicamente come il problema di determinare un argomento per cui una data funzione assume un valore estremo (minimo o massimo) su un dato dominio.
Formalmente, data una funzione $f : R^n \to R$ e un insieme $X \subseteq R^n$ si cerca $x^* \in X$ tale per cui f raggiunga un minimo su X in x^*, cioè $f(x^*) \leq f(x)$ $\forall x \in X$. x^* è detto *minimo* di f. Il *massimo* di f è il minimo di $-f$.
La *funzione obiettivo* f, che di solito si assume differenziabile, puó essere lineare o non lineare. L'insieme X generalmente è definito da un set di equazioni e disequazioni, dette *vincoli*, che possono essere lineari o non lineari. Un vettore $x \in X$ che soddisfa i vincoli si dice *ammissibile* e X è detto *insieme*

di ammissibilitá. Se $X = R^n$ il problema è non vincolato.

In generale, un problema di ottimizzazione continuo ha la forma

$$\begin{aligned} \min\ & f(x) \\ & g(x) = 0 \\ & h(x) \leq 0 \end{aligned}$$

dove $f : R^n \to R$, $h : R^n \to R^m$, $h : R^n \to R^p$.

I problemi di ottimizzazione si classificano a seconda delle proprietá delle funzioni coinvolte. Se f, g, h sono tutte lineari, allora si ha problema di *programmazione lineare*; se invece una delle funzioni è non lineare si parla di *programmazione non lineare.*

Prima di procedere, riprendiamo alcune definizioni:

Definizione 1.1. Se f è differenziabile su X, si definisce *gradiente* di f al punto $x \in X$ il vettore colonna

$$\nabla f(x) = \left(\frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, ..., \frac{\partial f(x)}{\partial x_n} \right)^T.$$

Se $\nabla f(x) = 0$, x si dice *punto critico* per $f(x)$.

Definizione 1.2. Se $f \in C^2$ su X, si definisce *matrice hessiana* di f in x la matrice simmetrica $n \times n$

$$H_f(x) = \begin{pmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \cdots & & & \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{pmatrix}.$$

Cerchiamo di essere piú precisi su cosa si intenda per soluzione di un problema di ottimizzazione.

Definizione 1.3. Sia f una funzione a valori reali definita su $X \subseteq R^n$. Un punto x^* in X è:

- un *minimo globale* per $f(x)$ su X se $f(x^*) \leq f(x)\ \forall x \in X$;

- un *minimo globale stretto* per $f(x)$ su X se $f(x^*) < f(x)$ $\forall x \in X$ tali che $x^* \neq x$;
- un *minimo locale* per $f(x)$ se esiste un $\delta > 0$ tale che $f(x^*) \leq f(x)$ $\forall x \in X$ tali che $x \in D(x^*, \partial)$;
- un *minimo locale stretto* per $f(x)$ se esiste un $\delta > 0$ tale che $f(x^*) < f(x)$ $\forall x \in X$ tali che $x \in D(x^*, \partial)$;

1.1 Ottimizzazione non vincolata

Si vuole risolvere il seguente problema:

$$\begin{aligned} \min \quad & f(x) \\ & x \in R^n, \end{aligned} \tag{1.1}$$

dove $f : R^n \to R$ è una funzione data. In generale si cercano soluzioni locali.

Si enunciano ora le condizioni necessarie del primo e del secondo ordine.

Proposizione 1.4 (**Condizioni necessarie**). *Sia x^* un minimo locale per f; si assume che f sia differenziabile con continuitá in qualche intorno di x^*. Allora*

$$\nabla f(x^*) = 0.$$

Inoltre se $f \in C^2$ su un intorno di x^*, allora

$$\nabla^2 f(x^*) \geq 0.$$

In generale la condizione di stazionarietá $\nabla f(x^*) = 0$ non garantisce un minimo: un punto critico puó essere un massimo, un minimo o nessuno nei due. Serve un criterio per verificare se i punti critici sono massimi/minimi o meno.

Proposizione 1.5 (**Condizione sufficiente**). *Sia $f \in C^2$ per qualche intorno di x^* e sia x^* tale che*

$$\nabla f(x^*) = 0,$$

$$\nabla^2 f(x^*) \geq 0.$$

Allora x^* è un minimo stretto locale.

Se $f \in C^1$ convessa, si ha la seguente condizione necessaria e sufficiente:

Proposizione 1.6. *Sia $f \in C^1$ convessa su R^n. Allora x^* è un minimo globale se e solo se $\nabla f(x^*) = 0$.*

1.2 Ottimizzazione vincolata

Si consideri il problema di ottimizzazione vincolata

$$\begin{aligned} min \quad & f(x) \\ & x \in X. \end{aligned}$$

In seguito X sará un insieme non vuoto, chiuso e convesso. Si enunciano le seguenti condizioni di ottimalitá.

Teorema 1.7. *Sia $f \in C^1$ su qualche intorno di $x^* \in X$. Se x^* è un minimo locale di f su X, allora*

$$\nabla f(x^*)'(x - x^*) \geq 0, \ \forall x \in X. \tag{1.2}$$

Se f è convesso su X, allora la condizione precedente è anche sufficiente affinchè x^* sia un minimo di f su X

Un vettore x^* che soddisfi la (1.2) è un punto stazionario. In assenza di convessitá di f questa condizione puó essere soddisfatta anche da punti di massimo e di sella. Si noti infatti che, se $X = R^n$ o se x^* è un punto interno di X, la (1.2) si riduce alla condizione di stazionarietá $\nabla f(x^*)$ dell'ottimizzazione non vincolata.

Si illustra ora la condizione (1.2) con un esempio.

Esempio 1.8. Si consideri l'ortante positivo

$$X = \{x | x > 0\}.$$

La condizione necessaria (1.2) affinchè x^* sia un minimo locale è

$$\sum_{i=1}^{n} \frac{\partial f(x^*)}{\partial x_i^*}(x_i - x_i^*) \geq 0, \quad \forall x_i \geq 0, i = 1, ..., n. \tag{1.3}$$

Fissato i e imponendo $x_j = x_j^*$ per $i \neq j$ e $x_i = x_i^* + 1$ nella (1.3), si ottiene

$$\frac{\partial f(x^*)}{\partial x_i^*} \geq 0, \quad \forall i. \tag{1.4}$$

Se $x^* > 0$, imponendo $x_j = x_j^*$ per $j \neq i$ e $x_1 = \frac{1}{2}x_i^*$ in (1.3), si ottiene $\partial f(x^*)/\partial x_i^* < 0$, che, combinato con (1.4) da

$$\frac{\partial f(x^*)}{\partial x_i^*} = 0, \quad x_i^* > 0. \tag{1.5}$$

Si consideri ora il caso in cui

$$X = \{x | \alpha_i \leq x_i \leq \beta_i, i = 1, ..., n\},$$

dove α_i e β_i sono due scalari. Si verifica che se x^* è un minimo locale allora

$$\begin{aligned}
&\frac{\partial f(x^*)}{\partial x_i^*} \geq 0, && x_i^* = \alpha_i \\
&\frac{\partial f(x^*)}{\partial x_i^*} \leq 0, && x_i^* = \beta_i \\
&\frac{\partial f(x^*)}{\partial x_i^*} = 0, && \alpha_i \leq x_i^* \leq \beta_i.
\end{aligned}$$

Generalmente la struttura di X è specificata da equazioni e disequazioni. Si distinguono ora questi due casi.

1.2.1 Problemi con vincoli di uguaglianza

Si consideri il seguente problema:

$$min \quad f(x) \tag{1.6}$$

$$h(x) = 0 \tag{1.7}$$

con $f : R^n \to R$, $h : R^n \to R^m$, con $m \leq n$.

Definizione 1.9. Sia x^* un vettore per cui $h(x^*) = 0$. x^* è un *punto regolare* se $\nabla h_1(x^*), ..., \nabla h_m(x^*)$ sono linearmente indipendenti.

Consideriamo la *funzione lagrangiana* $L : R^{n+m} \to R$ cosí definita

$$L(x, \lambda) = f(x) + \lambda' h(x).$$

Si hanno i seguenti risultati:

Proposizione 1.10 (**Condizioni necessarie**). *Sia x^* un minimo locale per il problema 1.6, siano f e h differenziabili con continuitá su un intorno di x^* e sia x^* un punto regolare. Allora esiste un unico vettore λ^* tale che*

$$\nabla_x L(x^*, \lambda^*) = 0.$$

Se inoltre $f \in C^2$ e $h \in C^2$ in un intorno di x^* allora

$$z' \nabla^2_{xx} L(x^*, \lambda^*) z \geq 0,$$

$\forall z \in R^n$ *con* $\nabla h(x^*)' z = 0$.

Le componenti del vettore λ^* sono dette *moltiplicatori di Lagrange.*

Questa condizione necessaria, insieme con il vincolo $h(x) = 0$, è equivalente al punto critico della Lagrangiana, $\nabla L(x, \lambda) = 0$, che è espressa da un sistema di $n + m$ equazioni in $n + m$ incognite

$$\nabla L(x, \lambda) = \begin{bmatrix} \nabla f(x) + J_h^T(x)\lambda \\ h(x) \end{bmatrix} = 0.$$

E' importante notare che l'Hessiana della Lagrangiana,

$$H_L(x, \lambda) = \begin{bmatrix} \nabla^2_{xx} L(x, \lambda) & J_h^T(x) \\ J_h(x) & 0 \end{bmatrix}$$

è simmetrica ma in generale non è definita positiva, anche se $\nabla^2_{xx} L(x, \lambda)$ è definita positiva. Se l'Hessiana della Lagrangiana non è definita positiva, come si verifica l'ottimalitá dei punti critici della Lagrangiana? Si dimostra che è sufficiente che $\nabla^2_{xx} L(x^*, \lambda^*)$ sia definita positiva sullo spazio tangente al vincolo, cioè sull'insieme dei vettori ortogonali alle righe di $J_h^T(x^*)$. In altre parole, è necessario che $\nabla^2_{xx} L(x^*, \lambda^*)$ sia definita positiva solo rispetto ai vettori localmente ammissibili.

Proposizione 1.11 (**Condizioni sufficienti**). *Sia x^* tale che $h(x^*) = 0$ e siano $f \in C^2$ e $h \in C^2$ su un intorno di x^*. Se esiste un vettore $\lambda^* \in R^n$ tale che*

$$\nabla_x L(x^*, \lambda^*) = 0$$

e

$$z' \nabla^2_{xx} L(x^*, \lambda^*) z > 0$$

$\forall z \neq 0$ con $\nabla h(x^*)' z = 0$ allora x^* è un minimo stretto per 1.6.

1.2.2 Problemi con vincoli di disuguaglianza

Quando sono presenti anche vincoli di disuguaglianza, ossia il problema ha la forma

$$\begin{aligned} min \quad & f(x) \\ & h(x) = 0 \\ & g(x) \leq 0, \end{aligned} \tag{1.8}$$

dove $f : R^n \to R$, $h : R^n \to R^m$ $g : R^n \to R^r$, con $m \leq n$, le condizioni di ottimalitá diventano piú complicate. Si puó ancora definire la Lagrangiana, includendo anche i vincoli di disuguaglianza, ma alcuni di questi possono essere irrilevanti. Per ogni vettore x tale che $g(x) \leq 0$, si definisce con

$$A(x) = \{j | g_j(x) = 0, j = 1, ..., r\}.$$

I vincoli $g_j(x) = 0, j = 1, ..., r$ sono detti *attivi*.

Definizione 1.12. Sia x^* un vettore tale che $h(x^*) = 0$, $g(x^*) \leq 0$ e siano $h \in C^1$, $g \in C^1$. x^* è un punto regolare se i gradienti $\nabla h_1(x^*), ..., \nabla h_m(x^*)$ e $\nabla g_j(x^*)$, per ogni $j \in A(x^*)$ sono linearmente indipendenti.

Si introduce la funzione Lagrangiana $L : R^{n+m+r} \to R$ per il problema 1.8 come

$$L(x, \lambda, \mu) = f(x) + \lambda' h(x) + \mu' g(x).$$

Le condizioni di ottimalitá sono analoghe a quelle del caso di vincoli di uguaglianza.

Proposizione 1.13 (**Condizioni necessarie**). *Sia x^* un punto di minimo locale per il problema 1.8. Si supponga che f, g e h siano differenziabili con continuitá in un intorno di x^* e sia x^* un punto regolare. Allora esistono due unici vettori $\lambda^* \in R^m$, $\mu^* \in R^r$ tali che*

$$\nabla_x L(x^*, \lambda^*, \mu^*) = 0,$$

$\mu_j^* \geq 0, \;\; \mu_j^* g_j(x^*) = 0 \quad \forall j = 1, ..., r.$

Se inoltre f, h e g sono due volte differenziabili con continuitá in un intorno di x^*, allora $\forall z \in R^n$ tale che $\nabla h(x^*)'z = 0$ e $\nabla g_j(x^*)'z = 0$ per $j \in A(x^*)$ si ha che

$$z' \nabla^2_{xx} L(x^*, \lambda^*) z \geq 0.$$

Proposizione 1.14 (**Condizioni sufficienti**). *Sia x^* tale che $h(x^*) = 0$ e $g(x^*) \leq 0$ e siano f, g, h due volte differenziabili con continuitá in un intorno di x^*. Si assume che esistano $\lambda^* \in R^n$ e $\mu^* \in R^r$ tali che*

$$\nabla_x L(x^*, \lambda^*, \mu^*) = 0,$$

$$\mu_j^* \geq 0, \quad \mu_j^* g_j(x^*) = 0 \quad \forall j = 1, ..., r$$

e che $\forall z \neq 0$ tale che $\nabla h(x^*)'z = 0$, $\nabla g(x^*)'z \leq 0$, per ogni $j \in A(x^*)$ (in particolare $\nabla g(x^*)'z = 0$, $\forall j \in A(x^*)$ con $\mu^* > 0$) si abbia

$$z' \nabla^2_{xx} L(x^*, \lambda^*) z > 0.$$

Allora x^* è un minimo locale stretto per 1.8.

1.2.3 Vincoli lineari

Le precedenti condizioni necessarie di ottimalitá sono basate sull'ipotesi di regolaritá del punto di minimo locale x^*. Se x^* non è regolare ci sono due possibilitá: o esistono infiniti moltiplicatori di Lagrange o non ne esistono. Ci sono diverse assunzioni in grado di sostituire l'ipotesi di regolaritá: una di queste è costituita dalla linearitá dei vincoli.

Proposizione 1.15. *Sia x^* un minimo locale del problema*

$$min \quad f(x) \tag{1.9}$$

$$a_j' x^* - b_j \leq 0, \quad j = 1, ..., r, \tag{1.10}$$

dove $f : R^n \to R$ è differenziabile con continuitá in un intorno di x^*, $b \in R^r$ e $a_j \in R^n$ per $j = 1, ..., r$. Allora esiste un vettore $\mu^* \in R^r$ tale che

$$\nabla f(x^*) + \sum_{j=1}^{r} \mu_j^* a_j = 0$$

e

$$\mu_j^* \geq 0, \quad \mu_j^*(a_j^T x^* - b_j) = 0, \quad j = 1, ..., r.$$

1.2.4 Metodo del gradiente proiettato

In questa sezione si descrive il metodo del gradiente proiettato. Il metodo piú semplice ha la forma

$$x_{k+1} = x_k + \alpha_k(\overline{x}_k - x_k) \tag{1.11}$$

dove

$$\overline{x}_k = [x_k - s_k \nabla f(x_k)]^+ .$$

$[.]^+$ indica la proiezione su X, $\alpha_k \in (0, 1]$ indica il passo e s_k è uno scalare positivo. Per ottenere il vettore $\overline{x}_k$ si fa un passo $-s_k \nabla f(x_k)$ lungo il gradiente negativo, come nel metodo della discesa ripida. Si proietta $x_k - s_k \nabla f(x_k)$ su X, ottenendo cosí il vettore $\overline{x}_k$. Infine si fa un passo lungo la direzione ammissibile $\overline{x}_k - x_k$ con passo α_k

Si puó anche vedere s_k come il passo. Infatti, ponendo $\alpha_k = 1$ per ogni k si ha $x_{k+1} = \overline{x}_k$ e il metodo diventa:

$$x_{k+1} = [x_k - s_k \nabla f(x_k)]^+ .$$

Se $x_k - s_k \nabla f(x_k)$ è un vettore ammissibile, il metodo si riduce alla discesa ripida nel caso non vincolato

$$x_k - \alpha_k \nabla f(x_k).$$

Si noti che si avrá

$$x^* = [x^* - s_k \nabla f(x^*)]^+$$

per ogni $s > 0$ se e solo se x^* è stazionario. Quindi il metodo si ferma se e solo se incontra un punto stazionario. Affinchè il metodo abbia successo l'operazione di proiezione non deve essere complessa. E' quindi preferibile che X abbia una struttura abbastanza semplice. Ci sono diversi modi per scegliere il passo del metodo:

- Regola della minimizzazione limitata

  lo scalare s_k è costante

  $$s_k = s, \qquad k = 0, 1, ...$$

  e α_k è scelto in modo da minimizzare

  $$f(x_k + \alpha_k(\overline{x}_k - x_k)) = min f(x_k + \alpha_k(\overline{x}_k - x_k))$$

  su $[0, 1]$

- Regola di Armijo lungo le direzioni ammissibili

  anche in questo caso s_k è costante

  $$s_k = s, \qquad k = 0, 1, ...$$

  e α_k è scelto con la regola di Armijo sull'intervallo $[0, 1]$.

  In particolare, dati gli scalari $\beta \in (0, 1)$, $\sigma \in (0, 1)$, si fissa $\alpha_k = \beta^{m_k}$, dove m_k è il primo intero non negativo m tale per cui

  $$f(x_k) - f(x_k + \beta^m(\overline{x}_k - x_k)) \geq -\sigma\beta^m \nabla f(x_k)'(\overline{x}_k - x_k)$$

- Regola di Armijo sull'arco di proiezione

  α_k è costante

  $$\alpha_k = 1, \qquad k = 0, 1, ...$$

e s_k è determinato con riduzioni successive. x_{k+1} è determinato con una ricerca sull'arco di proiezione

$$\{x_k(s)|s > 0\}$$

dove, per ogni $s > 0$, $x_k(s)$ è definito da

$$x_k(s) = [x_k - s\nabla f(x_k)]^+ .$$

In particolare, fissati gli scalari $\overline{s}$, β e σ, con $\beta \in (0,1)$ e $\sigma \in (0,1)$, si pone $s_k = \beta^{m_k}\overline{s}$, dove m_k è il primo intero non negativo tale per cui

$$f(x_k) - f(x_k(\beta^m(\overline{s}) \geq \sigma\nabla f(x_k)'(x_k - x_k(\beta^m(\overline{s}).$$

- Passo costante

  in questo caso

  $$s_k = s, \quad \alpha_k = 1, \quad k = 0, 1, ...$$

  E' possibile dimostarare che se s_k è sufficientemente piccolo, i limiti della successione generata dal metodo del gradiente proiettato con passo costante sono stazionari.

- Diminishing stepsize

  α_k è fissato all'unitá e

  $$s_k \to 0, \quad \sum_{k=o}^{\infty} s_k = \infty.$$

  In questo caso, la discesa non è garantita ad ogni iterazione, ma diventa piú probabile man mano che s_k diminuisce.

Il metodo del gradiente proiettato presenta le stesse proprietá di convergenza del metodo dello steepest descent del caso non vincolato. Come quest'ultimo, soffre di una bassa velocitá di convergenza. In certi casi puó essere utile effettuare uno *scaling* e considerare il problema con un diverso sistema di coordinate.

Sia H_k una matrice definita positiva e si consideri la trasformazione di variabili

$$x = (H_k)^{-1/2} y. \tag{1.12}$$

Il problema puó essere riscritto come

$$\begin{aligned} min \quad & h_k(y) \equiv f((H_k)^{-1/2} y) \\ & y \in Y_k, \end{aligned} \tag{1.13}$$

dove Y_k è l'insieme

$$Y_k = \left\{ y | (H_k)^{-1/2} y \in X \right\}.$$

L'iterazione del metodo per questo problema prende la forma

$$y_{k+1} = y_k + \alpha_k(\overline{y}_k - y_k), \tag{1.14}$$

dove

$$\overline{y}_k = \left[y_k - s_k \nabla h(y_k) \right]^+ .$$

Si puó dimostrare che la (1.14) si puó scrivere come

$$x_{k+1} = x_k + \alpha_k(\overline{x}_k - x_k),$$

dove $\overline{x}_k$ è dato da

$$\overline{x}_k = argmin \left\{ \nabla f(x_k)'(x - x_k) + \frac{1}{2s_k}(x - x_k)' H_k (x - x_k) \right\}. \tag{1.15}$$

La velocitá di convergenza di questo metodo è governata dal piú piccolo e dal piú grande autovalore dell'Hessiana $\nabla^2 H_k(y_k)$. Il suggerimento per una buona velocitá di convergenza è quello di scegliere H_k quanto piú vicino possibile a $\nabla^2 f(x_k)$.

Se $H_k = \nabla^2 f(x_k)$ si ottiene il metodo di Newton vincolato.

Metodo di Newton vincolato

Supponiamo che f sia due volte differenziabile e che $\nabla^2 f(x_k)$ sia definita positiva per ogni $x \in X$. Si consideri il metodo del gradiente proiettato con cambio di variabili $H_k = \nabla^2 f(x_k)$. Si ha

$$x_{k+1} = x_k + \alpha_k(\overline{x}_k - x_k), \tag{1.16}$$

dove

$$\overline{x}_k = argmin \left\{ \nabla f(x_k)'(x - x_k) + \frac{1}{2s_k}(x - x_k)'\nabla^2 f(x_k)(x - x_k) \right\}. \tag{1.17}$$

Se $s_k = 1$, la (1.17) è l'espansione in serie di Taylor di f intorno a x_k al secondo ordine. In particolare se $s_k = 1$ e $\alpha_k = 1$, x_{k+1} è il vettore che minimizza l'espansione al secondo ordine della serie di Taylor intorno a x_k. Ci si aspetta quindi che per un valore di partenza x_0 abbastanza vicino al minimo locale x^*, il metodo con $s_k = 1$ e $\alpha_k = 1$ converga a x^* in modo superlineare.

L'inconveniente principale del metodo del gradiente proiettato è il costo computazionale del calcolo della proiezione ad ogni iterazione. Per ottenere una buona velocitá di convergenza, spesso è necessario ricorrere al cambio di coordinate. Sfortunatamente, in questo caso, anche se i vincoli sono semplici, il corrispondente problema di programmazione quadratica puó essere dispendioso in termini di tempo.

1.2.5 I metodi di proiezione a due metriche

Per risolvere il problema della scarsa velocitá di convergenza e della complessa implementazione del metodo del gradiente proiettato, si puó utilizzare in alternativa il *metodo di proiezione a due metriche*, definito cosí

$$x_{k+1} = [x_k - \alpha_k D_k \nabla f(x_k)]^+ . \tag{1.18}$$

D_k è una matrice definita positiva, non necessariamente diagonale e [.] è la proiezione su X.

Questo metodo è chiamato *metodo di proiezione a due metriche* perchè incorpora due diversi tipi di matrici: D_k, che scala il gradiente, e la matrice identitá I usata nella proiezione.

In questa sezione si discute il metodo nel caso dell'ortante positivo

$$X = \{x | x \geq 0\},$$

cioè il problema

$$min \quad f(x) \quad x \geq 0. \tag{1.19}$$

In questo metodo c'è un problema fondamentale: in generale la direzione non è di discesa. In particolare si puó avere $f(x_{k+1}) > f(x_k)$ per ogni scelta del passo α_k.

Esiste tuttavia una classe di matrici D_k per la quale la discesa è garantita. Questa classe è abbastanza ampia da permettere una convergenza di tipo superlineare quando D_k incorpora informazioni sulla derivata seconda (ad esempio $D_k = (\nabla^2 f(x_k))^{-1}$). Definiamo allora questa classe si matrici.

Definizione 1.16. Si denoti per ogni $x \geq 0$

$$I^+(x) = \left\{ i | x_i = 0, \frac{\partial f(x)}{\partial x_i} > 0 \right\}.$$

Si dice che una matrice simmetrica $n \times n$ D con elementi d_{ij} è *diagonale rispetto al sottoinsieme di indici* $I \subset \{1, 2, ..., n\}$, se

$$d_{ij} = 0, \quad \forall i \in I, \quad j = 1, 2, ..., n, \quad j \neq i.$$

Proposizione 1.17. *Sia $x \geq 0$ e sia D una matrice simmetrica definita positiva diagonale rispetto a $I^+(x)$. Si denoti*

$$x(\alpha) = [x - \alpha D \nabla f(x)]^+, \quad \forall \alpha \geq 0.$$

1. *Il vettore x è stazionario se e solo se*

$$x = x(\alpha), \quad \forall \alpha \geq 0.$$

2. Se x non è stazionario, allora esiste uno scalare $\overline{\alpha} > 0$ tale che

$$f(x(\alpha)) < f(x), \quad \forall \alpha \in (0, \overline{\alpha}].$$

Grazie alla proposizione 1.17, si conclude che, per garantire la discesa, la matrice D nell'iterazione

$$x_{k+1} = [x_k - \alpha_k D_k \nabla f(x_k)]^+$$

deve essere scelta diagonale rispetto a un sottoinsieme di indici che contenga

$$I^+(x_k) = \left\{ i | x_i^k = 0, \frac{\partial f(x_k)}{\partial x_i} > 0 \right\}.$$

Capitolo 2

Il problema della ricostruzione di immagini

Un'immagine digitale è una matrice di elementi chiamati *pixel*. A ogni pixel è assegnata un'intensitá, che caratterizza il colore di una piccola sezione rettangolare della scena. Un'immagine varia da $256^2 = 65536$ pixel ai 10 milioni di pixel per le immagini ad alta risoluzione.

In questo capitolo sono introdotte le basi del il problema della ricostruzione delle immagini digitali. Nella prima sezione è descritto il processo di acquisizione di un'immagine. In seguito viene analizzato il modello lineare che descrive il processo di formazione e registrazione dell'immagine: le componenti del modello sono la PSF, il rumore e le condizioni al contorno.

Infine vengono studiate le proprietá delle matrici coinvolte che permettono la facilitazione dei calcoli e una notevole diminuzione del costo computazionale. Per lo studio di questi argomenti si è consultato [4].

2.1 Acquisizione delle immagini

Un'immagine digitale viene acquisita con strumenti specifici per l'oggetto da rappresentare: un telescopio per un'immagine di una galassia, un microscopio elettronico per l'immagine di un batterio.

Generalmente, uno strumento per l'acquisizione è composto da due parti: la prima serve a captare le radiazioni emesse dall'oggetto da rappresentare. Ad esempio, nel caso del telescopio tale dispositivo è il C.C.D (Charge Copuled Device): è composto da materiali semiconduttori che convertono la luce incidente in cariche elettriche. La seconda parte dello strumento trasforma il segnale fisico in un segnale digitale, ovvero in un segnale trattabile dagli elaboratori; nel caso del telescopio, questo è costituito dal dispositivo che tramuta le cariche elettriche in dati numerici gestibili dal computer.

L'acquisizione delle immagini è affetta da errori. La rappresentazione dell'oggetto infatti ha una degradazione dovuta ai seguenti due fattori: quello di formazione dell'immagine, chiamato *blurring*, e quello di registrazione dell'immagine, il rumore (*noise*). Il blurring è dovuto alla struttura dello strumento ed è causato da fenomeni fisici come la diffrazione o l'aberrazione; il rumore invece è dovuto al fatto che si effettua un passaggio da uno spazio continuo, la realtà, ad uno spazio discreto, la memoria del computer. Inoltre, si introducono anche degli errori statistici dovuti al processo di digitalizzazione. I dati raccolti sono in realtá la realizzazione di variabili aleatorie.

E' sfortunatamente impossibile recuperare esattamente l'immagine originale. Lo scopo dell'image deblurring è quello di sviluppare algoritmi affidabili ed efficienti per recuperare quante piú informazioni possibili dall'immagine data.

Per recuperare l'immagine originale viene usato un modello matematico che descrive il processo di blurring. La chiave del problema sta nel fatto che i dettagli sulle informazioni perse sono presenti nell'immagine sfuocata, ma queste informazioni sono nascoste e possono essere recuperate se si conoscono i dettagli del processo di sfuocamento.

2.2 Modellizzazione del problema

Si assume innanzitutto che il blurring possa essere descritto da un modello matematico lineare. Infatti in molte situazioni è lineare o ben approssimato da un modello lineare.

Si assume inoltre l'esistenza dell'immagine esatta, l'immagine ideale che si vorrebbe catturare, quella che si otterrebbe se blurring e noise non esistessero. Ha la stessa dimensione dell'immagine sfuocata B e la si indica con X.

Per ottenere il modello lineare è necessario riordinare gli elementi delle matrici che rappresentano le immagini in modo da formare vettori di lunghezza $N = n \cdot m$. Si userá la seguente notazione:

$$x = \mathrm{vec}(X) = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{bmatrix} \in R^N, \qquad b = \mathrm{vec}(B) = \begin{bmatrix} b_1 \\ b_2 \\ \cdot \\ \cdot \\ \cdot \\ b_n \end{bmatrix} \in R^N.$$

Poichè si è assunto che il blurring sia un operazione lineare, deve esistere una matrice A di dimensioni $N = n \times m$ tale che b e x sono legate da

$$Ax = b. \tag{2.1}$$

Infine bisogna aggiungere la componente che rappresenta il rumore $e = \mathrm{vec}(E)$:

$$b = Ax + e. \tag{2.2}$$

Come ricavare A, la matrice che rappresenta il processo che modifica l'immagine reale in immagine sfocata? Si vedrá che A è determinata da una funzione di dispersione, la PSF, che definisce la deformazione di ogni pixel, e dalle condizioni al contorno, che ipotizzano l'oggetto al di fuori dell'immagine registrata.

Prima di procedere con l'analisi del modello, si analizza in dettaglio a cosa è dovuto il rumore.

2.3 Noise

Una componente di degradazione dell'immagine è data dal rumore (*noise*). Puó avere diverse cause, ed essere di tipo lineare o nonlineare, additivo o moltiplicativo.

In questo modello, il rumore deriva principalmente da 3 cause:

- il dispositivo a scorrimento di carica ha il compito di trasformare la luce in cariche elettriche. In pratica conta i fotoni incidenti sul chip di materiale semiconduttore e converge questo conteggio in cariche elettriche. Tale conteggio è imperfetto e normalmente questo tipo di errore segue una distribuzione di Poisson

  $$p_\lambda(x) = e^{-\lambda}\frac{\lambda^k}{k!}.$$

  Questo errore dipende dall'immagine considerata, in quanto λ dipende dal prodotto di convoluzione Ax. Bisogna anche considerare le emissioni di fotoni dell'ambiente circostante: anche questo numero di fotoni segue una distribuzione di Poisson di parametro γ fissato.

- Il processo di discretizzazione e la conversione da dati analogici a digitali generano il cosiddetto *rumore di lettura*, che si verifica sia quando si tramuta il conteggio dei fotoni in cariche elettriche, sia quando queste cariche elettriche vengono riconosciute e convertite in dati numerici. Tale rumore è costituito da valori casuali identicamente e indipendentemente distribuiti, ed è chiamato *whitenoise*; il suo modello matematico è la curva gaussiana

  $$p_{\mu,\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp - \frac{(x-\mu)^2}{2\sigma^2}$$

  in cui la media è zero e la deviazione standard è proporzionale all'ampiezza del rumore.

- La conversione da dato analogico a dato digitale restituisce un altro tipo di errore, detto *errore di quantizzazione*, quando il numero di bit utilizzati è relativamente basso. Anch'esso è modellizzato da rumore bianco

uniformemente distribuito, la cui deviazione standard è inversamente proporzionale al numero di bit usati.

In termini di algebra lineare, si puó descrivere il contributo del rumore additivo all'immagine sfocata come segue:

$$B = B_{esatta} + E,$$

dove E è una matrice $m \times n$ che contiene, per esempio, elementi di una distribuzione di Poisson o di Gauss (o una somma di entrambe).

2.4 Determinazione della PSF

Si ritorna ora al probelma della determinazione della matrice A del modello (2.2).

Si immagini il seguente esperimento: si considera come immagine esatta un'immagine completamente nera ad eccezione di un pixel bianco. Se si fa una foto a questa immagine, ció che si ottiene è un'immagine sfocata del pixel. Il singolo pixel acceso è chiamato *point source* mentre la funzione che descrive il blurring e l'immagine risultante è detta *Point Spread Function* (PSF), ovvero funzione di allargamento del punto. L'azione della PSF sul punto, e quindi su tutta l'immagine, viene detta blurring, o sfocamento.

Solitamente l'intensità della luce della PSF è confinata in una piccola area intorno al centro della PSF (il point source), mentre fuori da un certo raggio l'intensitá è zero. In altre parole, il blurring è un fenomeno locale. Grazie a questa ipotesi, per rappresentare la PSF basta una matrice di dimensioni molto minori rispetto a quella dell'immagine sfocata. Tuttavia, molti algoritmi di deblurring richiedono una matrice di dimensioni pari a quella dell'immagine. In tal caso, si ricorre allo zero padding per ingrandire l'immagine con dei bordi neri.

Assumendo che il processo di acquisizione dell'immagine catturi tutta la luce, allora la somma dei valori dei pixel deve dare zero. In questo modello si assume inoltre che la PSF sia la stessa indipendentemente dalla posizione

(a) Sorgente puntiforme di intensitá unitaria (b) Immagine della sorgente puntiforme

Figura 2.1: Determinazione della PSF

del pixel acceso. In questo caso si dice che il blurring è *spazio-invariante*. In seguito, la matrice che rappresenta la PSF verrá indicata con P.

In alcuni casi la PSF puó essere descritta analiticamente e cosí P puó essere costruita a partire dalla funzione. Talvolta invece la conoscenza del processo fisico che causa lo sfocamento fornisce una formulazione della PSF. In questo caso gli elementi della matrice P sono dati da una precisa funzione matematica.

Sono ora illustrati alcuni esempi di PSF:

- Blur di turbolenza atmosferica: è descritto da una funzione gaussiana bidimensionale, gli elementi di P sono dati da

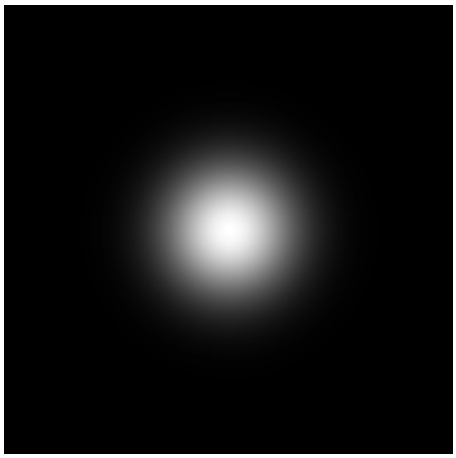
$$p_{ij} = C_0 \exp\left(\frac{1}{2}\begin{bmatrix} i-k \\ j-l \end{bmatrix}^T \begin{bmatrix} s_1^2 & \rho^2 \\ \rho^2 & s_2^2 \end{bmatrix}^{-1} \begin{bmatrix} i-k \\ j-l \end{bmatrix}\right) \tag{2.3}$$

  dove (k, l) è il centro della PSF, mentre s_1 e s_2 sono i parametri regolanti la larghezza e l'altezza e ρ rappresenta l'orientamento.

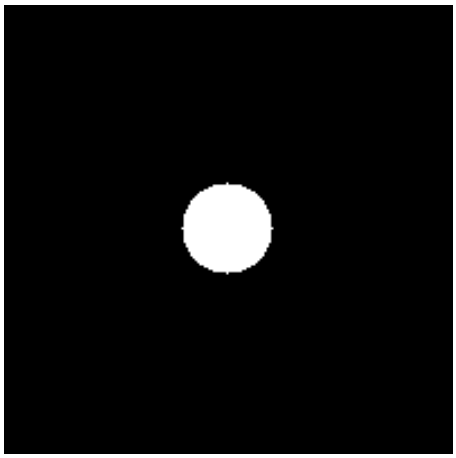
- Out-of-focus Blur: gli elementi di P sono dati da

$$p_{ij} = \begin{cases} C_1 \frac{1}{\pi r^2} & (i-k)^2 + (j-k)^2 \leq r^2 \\ 0 & \text{altrimenti} \end{cases}$$

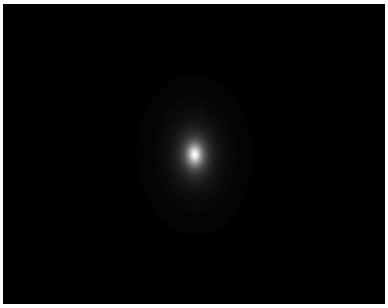
  dove (k, l) è il centro della PSF e r è il raggio dello sfocamento;



(a) Blur di turbolenza atmosferica



(b) Out-of-focus Blur



(c) Blur di Moffat

Figura 2.2: Esempi di PSF

- Blur di Moffat: usato per immagini astronomiche, gli elementi di P sono dati da

$$p_{ij} = C_2 \left(1 + \begin{bmatrix} i-k \\ j-l \end{bmatrix}^T \begin{bmatrix} s_1^2 & \rho^2 \\ \rho^2 & s_2^2 \end{bmatrix}^{-1} \begin{bmatrix} i-k \\ j-l \end{bmatrix} \right)^{-\beta} \tag{2.4}$$

  dove s_1, s_2, ρ sono parametri, (k, l) il centro, mentre β controlla l'andamento della funzione, che in generale è piú lento di una PSF gaussiana.

Le costanti C_i che appaiono nelle formule servono a normalizzare la funzione, in modo che l'integrale su tutto lo spazio abbia valore 1.

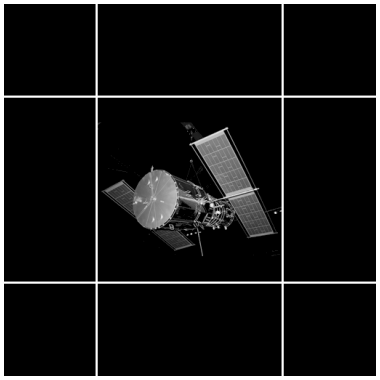
2.5 Condizioni al contorno

Si è visto che nel processo di blurring un pixel viene determinato dalle informazioni contenute nel pixel corrispondente dell'immagine esatta e nei pixel ad esso adiacenti. E' chiaro allora che mancano delle informazioni sui pixel dell'immagine esatta che si trovano ai bordi dell'immagine registrata. Nella formazione della PSF si cerca di recuperare queste informazioni mancanti assumendo delle opportune condizioni al contorno, cioè decidendo il comportamento dell'immagine al di fuori dei confini che si stanno considerando. Le condizioni al contorno possono essere:

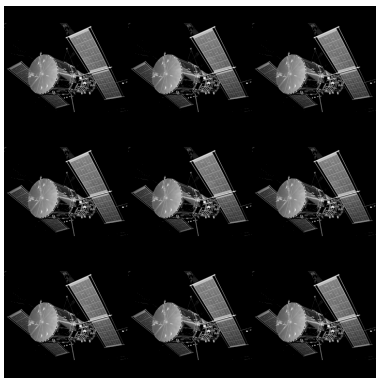
- **Zero Boundary conditions**:

  si suppone che l'immagine sia immersa in uno spazio totalmente nero. Si effettua il cosiddetto *zero padding*, si aggiungono cioè degli zeri, ovvero dei pixel neri, intorno all'immagine considerata. Ad esempio se F è scritta per blocchi nel seguente modo

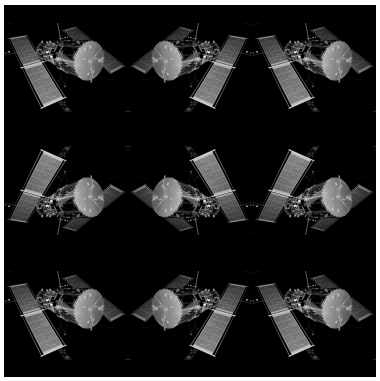
$$\begin{pmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{pmatrix}$$



(a) Zero Boundary Conditions



(b) Periodic Boundary Conditions



(c) Reflexive Boundary Conditions

Figura 2.3: Condizioni al contorno

allora il suo zero padding è

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & F_{11} & F_{12} & 0 \\ 0 & F_{21} & F_{22} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

- **Periodic Boundary conditions**:

  si suppone che l'immagine si ripeta identicamente in tutte le direzioni. Per far ció si circonda l'immagine con tante copie della stessa. Ad esempio, se l'immagine è rappresentata da una matrice X, utilizzando queste condizioni al contorno si considererá l'immagine rappresentata dalla seguente matrice:

$$\begin{pmatrix} X & X & X \\ X & X & X \\ X & X & X \end{pmatrix}$$

- **Reflexive Boundary conditions**:

  in questo caso, si suppone che l'immagine sia circondata da riflessioni speculari di se stessa. Se ad esempio un'immagine è rappresentata da una matrice di questo tipo

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

allora imponendo queste condizioni al contorno si ottiene

$$\left(\begin{array}{ccc|ccc|ccc} 9 & 8 & 7 & 7 & 8 & 9 & 9 & 8 & 7 \\ 6 & 5 & 4 & 4 & 5 & 6 & 6 & 5 & 4 \\ 3 & 2 & 1 & 1 & 2 & 3 & 3 & 2 & 1 \\ \hline 3 & 2 & 1 & 1 & 2 & 3 & 3 & 2 & 1 \\ 6 & 5 & 4 & 4 & 5 & 6 & 6 & 5 & 4 \\ 9 & 8 & 7 & 7 & 8 & 9 & 9 & 8 & 7 \\ \hline 9 & 8 & 7 & 7 & 8 & 9 & 9 & 8 & 7 \\ 6 & 5 & 4 & 4 & 5 & 6 & 6 & 5 & 4 \\ 3 & 2 & 1 & 1 & 2 & 3 & 3 & 2 & 1 \end{array}\right)$$

La scelta delle condizioni è dettata dal tipo di problema che si sta considerando. Ad esempio lo zero padding è un metodo adatto nei casi in cui lo sfondo degli oggetti da rappresentare sia nero, come nel caso di immagini astronomiche.

2.6 Matrici strutturate

Ora che sono state illustrate le componenti di base del modello di blurring, si puó dare una descrizione della matrice A, definita nel modello lineare (2.2). Conoscendo infatti la PSF e stabilendo le condizioni al contorno, si è in grado di calcolare la matrice A.

Si è visto nelle sezioni precedenti che nel processo di blurring un pixel viene determinato dalle informazioni contenute nel pixel corrispondente dell'immagine esatta e nei pixel ad esso adiacenti. In particolare, il valore del singolo pixel è una sorta di media pesata del corrispondente pixel nell'immagine esatta e dei pixel circostanti. Gli elementi della matrice P giocano il ruolo dei pesi.

Dall'analisi matematica si sa che il prodotto di convoluzione[1] svolge que-

[1]Si considerino due funzioni $f(t), g(t) : R \to R$, dove $f(t)$ è a supporto compatto e $g(t)$ integrabile secondo Lebesgue sui compatti di R. Si definisce **convoluzione** di f e g la

sta operazione. Nel caso discreto[2] l'integrale del prodotto di convoluzione viene sostitito da una sommatoria di termini finiti.

Si analizza dapprima il caso di convoluzione unidimensionale, successivamente si generalizza a problemi bidimensionali.

2.6.1 Caso Unidimensionale

Supponiamo che i vettori dell'immagine esatta x e della PSF p siano dati rispettivamente da

$$\begin{bmatrix} w_1 \\ w_2 \\ \hline x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ \hline y_1 \\ y_2 \end{bmatrix} \qquad \text{e} \qquad \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{bmatrix},$$

dove w_i e y_i rappresentano i pixel appena fuori dai bordi dell'immagine registrata. Si vuole determinare il vettore b della convoluzione tra x e p. Il suo i-esimo elemento si ottiene nel modo seguente:

- si ruota p di 180 gradi;

---

funzione cosí definita:

$$(f * g)(t) := \int_{-\infty}^{+\infty} f(\tau) g(t - \tau) d\tau.$$

È cioè l'integrale del prodotto delle due funzioni dopo che una delle funzioni di partenza è stata rovesciata e traslata.

[2]Per funzioni discrete, si può usare la versione discreta della convoluzione:

$$(f * g)(m) := \sum_n f(n) g(m - n).$$

- si moltiplicano elemento per elemento il vettore p ruotato e il vettore di 5 elementi che si ottiene da x, considerando come elemento centrale x_i;

- si sommano tutti i prodotti risultanti.

La convoluzione b tra x e p si puó scrivere come il prodotto matrice-vettore:

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \begin{bmatrix} p_5 & p_4 & p_3 & p_2 & p_1 & & & & \\ & p_5 & p_4 & p_3 & p_2 & p_1 & & & \\ & & p_5 & p_4 & p_3 & p_2 & p_1 & & \\ & & & p_5 & p_4 & p_3 & p_2 & p_1 & \\ & & & & p_5 & p_4 & p_3 & p_2 & p_1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \hline x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ \hline y_1 \\ y_2 \end{bmatrix}. \tag{2.5}$$

A seconda delle condizioni al contorno illustrate nella sezione precedente si hanno i seguenti casi:

- Zero Boundary conditions: in questo caso $w_i = y_i = 0$ e cosí la (2.5) si puó riscrivere come:

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \begin{bmatrix} p_3 & p_2 & p_1 & & \\ p_4 & p_3 & p_2 & p_1 & \\ p_5 & p_4 & p_3 & p_2 & p_1 \\ & p_5 & p_4 & p_3 & p_2 \\ & & p_5 & p_4 & p_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}. \tag{2.6}$$

  Una matrice i cui elementi siano costanti su ogni diagonale come in (2.6) è chiamata **matrice di Toeplitz**.

- Periodic Boundary conditions: in questo caso $w_1 = x_4$, $w_2 = x_5$, $y_1 = x_1$ e $y_2 = x_2$. La (2.5) puó essere riscritta come:

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \left(\begin{bmatrix} p_3 & p_2 & p_1 & & \\ p_4 & p_3 & p_2 & p_1 & \\ p_5 & p_4 & p_3 & p_2 & p_1 \\ & p_5 & p_4 & p_3 & p_2 \\ & & p_5 & p_4 & p_3 \end{bmatrix} + \begin{bmatrix} & & & p_5 & p_4 \\ & & & & p_5 \\ & & & & \\ p_1 & & & & \\ p_2 & p_1 & & & \end{bmatrix} \right) \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}$$

$$= \begin{bmatrix} p_3 & p_2 & p_1 & p_5 & p_4 \\ p_4 & p_3 & p_2 & p_1 & p_5 \\ p_5 & p_4 & p_3 & p_2 & p_1 \\ p_1 & p_5 & p_4 & p_3 & p_2 \\ p_2 & p_1 & p_5 & p_4 & p_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}. \tag{2.7}$$

  Una matrice di Toeplitz in cui ogni riga (colonna) è lo shift della riga (colonna) precedente, come in (2.7), è detta **matrice circolante**.

- Reflexive Boundary conditions. In questo caso si assume che $w_1 = x_2$, $w_2 = x_1$, $y_1 = x_5$ e $y_2 = x_4$ e la (2.5) puó essere riscritta come:

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} = \left(\begin{bmatrix} p_3 & p_2 & p_1 & & \\ p_4 & p_3 & p_2 & p_1 & \\ p_5 & p_4 & p_3 & p_2 & p_1 \\ & p_5 & p_4 & p_3 & p_2 \\ & & p_5 & p_4 & p_3 \end{bmatrix} + \begin{bmatrix} p_4 & p_5 & & & \\ p_5 & & & & \\ & & & & \\ & & & & p_1 \\ & & & p_1 & p_2 \end{bmatrix} \right) \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}$$

$$= \begin{bmatrix} p_3 + p_4 & p_2 + p_5 & p_1 & & \\ p_4 + p_5 & p_3 & p_2 & p_1 & \\ p_5 & p_4 & p_3 & p_2 & p_1 \\ & p_5 & p_4 & p_3 & p_2 + p_1 \\ & & p_5 & p_4 + p_1 & p_3 + p_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}. \tag{2.8}$$

  Una matrice i cui elementi siano costanti su ogni antidiagonale è detta **matrice di Hankel**, cosí la matrice in (2.8) è detta *matrice di Toeplitz+Hankel*.

2.6.2 Caso Bidimensionale

Il prodotto di convoluzione di matrici bidimensionali è molto simile al caso unidimensionale.
Nel caso di condizioni al bordo nulle, il vettore b è dato da:

$$\begin{bmatrix} b_{11} \\ b_{21} \\ b_{31} \\ b_{12} \\ b_{22} \\ b_{32} \\ b_{13} \\ b_{23} \\ b_{33} \end{bmatrix} = \left[\begin{array}{ccc|ccc|ccc} p_{22} & p_{12} & & p_{21} & p_{11} & & & & \\ p_{32} & p_{22} & p_{12} & p_{31} & p_{21} & p_{11} & & & \\ & p_{32} & p_{22} & & p_{31} & p_{21} & & & \\ \hline p_{23} & p_{13} & & p_{22} & p_{12} & & p_{21} & p_{11} & \\ p_{33} & p_{23} & p_{13} & p_{32} & p_{22} & p_{12} & p_{31} & p_{21} & p_{11} \\ & p_{33} & p_{23} & & p_{32} & p_{22} & & p_{31} & p_{21} \\ \hline & & & p_{23} & p_{13} & & p_{22} & p_{12} & \\ & & & p_{33} & p_{23} & p_{13} & p_{32} & p_{22} & p_{12} \\ & & & & p_{33} & p_{23} & & p_{32} & p_{22} \end{array}\right] \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{12} \\ x_{22} \\ x_{32} \\ x_{13} \\ x_{23} \\ x_{33} \end{bmatrix} \tag{2.9}$$

La matrice dell'equazione precedente ha una struttura a blocchi di Toeplitz in cui ogni blocco è anch'esso una matrice di Toeplitz ed è detta **matrice di Toeplitz a blocchi di Toeplitz (BTTB)**.

Nel caso di condizioni al bordo periodiche si ha:

$$\begin{bmatrix} b_{11} \\ b_{21} \\ b_{31} \\ b_{12} \\ b_{22} \\ b_{32} \\ b_{13} \\ b_{23} \\ b_{33} \end{bmatrix} = \left[\begin{array}{ccc|ccc|ccc} p_{22} & p_{12} & p_{32} & p_{21} & p_{11} & p_{31} & p_{23} & p_{13} & p_{33} \\ p_{32} & p_{22} & p_{12} & p_{31} & p_{21} & p_{11} & p_{33} & p_{23} & p_{13} \\ p_{12} & p_{32} & p_{22} & p_{11} & p_{31} & p_{21} & p_{13} & p_{33} & p_{23} \\ \hline p_{23} & p_{13} & p_{33} & p_{22} & p_{12} & p_{32} & p_{21} & p_{11} & p_{31} \\ p_{33} & p_{23} & p_{13} & p_{32} & p_{22} & p_{12} & p_{31} & p_{21} & p_{11} \\ p_{13} & p_{33} & p_{23} & p_{12} & p_{32} & p_{22} & p_{11} & p_{31} & p_{21} \\ \hline p_{21} & p_{11} & p_{31} & p_{23} & p_{13} & p_{33} & p_{22} & p_{12} & p_{32} \\ p_{31} & p_{21} & p_{11} & p_{33} & p_{23} & p_{13} & p_{32} & p_{22} & p_{12} \\ p_{11} & p_{31} & p_{21} & p_{13} & p_{33} & p_{23} & p_{12} & p_{32} & p_{22} \end{array}\right] \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{12} \\ x_{22} \\ x_{32} \\ x_{13} \\ x_{23} \\ x_{33} \end{bmatrix} \tag{2.10}$$

Una matrice si dice **circolante a blocchi** se ogni riga (colonna) formata dai blocchi è uno shift periodico della precedente riga (colonna). Una matrice si dice **circolante a blocchi circolanti (BCCB)** se è circolante a blocchi

e ogni blocco è circolante. La matrice dell'equazione precedente è di questo tipo.

Nel caso di condizioni al bordo riflessive, la matrice A è somma di quattro matrici: una matrice a blocchi di Toeplitz con blocchi di Hankel **(BTHB)**, una matrice a blocchi di Hankel con blocchi di Toeplitz **(BHTB)**, una matrice a blocchi di Hankel con blocchi di Hankel **(BHHB)** e una matrice a blocchi di Toeplitz con blocchi di Toeplitz **(BTTB)**.

Nella ricostruzione di immagini si ha a che fare con problemi di grandi dimensioni. Sorgono cosí delle difficoltá dovute all'elevato costo computazionale, dato che, come si è visto, il modello per la formazione di un'immagine digitale prevede un prodotto matrice-vettore Ax. Ad esempio, se l'immagine x ha dimensioni 256×256, il prodotto matrice-vettore implica la moltiplicazione di una matrice A di dimensioni $256^2 \times 256^2 = 65536 \times 65536$ per un vettore x di dimensioni $256^2 \times 1 = 65536 \times 1$. Nel prossimo paragrafo si mostra che le matrici coinvolte in questo modello godono di particolari proprietá che permettono una notevole diminuzione del costo computazionale.

2.6.3 Proprietá delle matrici BCCB

Non è difficile verificare che le matrici BCCB sono normali, cioè $A^*A = AA^*$. Esiste dunque la decomposizione spettrale $A = VDV^{-1}$. In particolare una matrice BCCB si decompone in questo modo

$$A = F^*\Gamma F,$$

dove F è la matrice della trasformata discreta di Fourier[3], Γ è la matrice diagonale contenente gli autovalori e F^* è la matrice aggiunta di F.

La matrice F gode di un'importante proprietà: è possibile effettuare i prodotti di F e F^* con un vettore senza costruire esplicitamente F, ma usando la trasformata di Fourier discreta. In MATLAB, esistono le funzioni `fft` e `ifft`[4] che calcolano il prodotto matrice-vettore rispettivamente con F e F^*.

Come calcolare gli autovalori di A? Si puó verificare che la prima colonna di F è un vettore unitario, moltiplicato per l'inverso della radice quadrata della dimensione, $\sqrt{N}$. Denotando con a_1 e f_1 rispettivamente la prima colonna di A e F, si ha:

$$A = F^*\Lambda F \quad \Rightarrow \quad FA = \Lambda F \quad \Rightarrow \quad Fa_1 = \Lambda f_1 = \frac{1}{\sqrt{N}}\lambda,$$

dove λ è il vettore contenente gli autovalori di A. Per trovare gli autovalori di A basta quindi moltiplicare a_1 per la matrice $\sqrt{N}F$, e quindi applicare la funzione `fft2` a una matrice contenente la prima colonna di A.

Si è visto che esiste una maniera efficiente in termini di tempo e di memoria per calcolare la decomposizione spettrale di una matrice BCCB.

Una volta ottenuta la decomposizione spettrale si possono eseguire efficientemente diversi calcoli matriciali con matrici BCCB.

---

[3]Se x è un vettore di lunghezza n di componenti x_j, allora la **trasformata discreta di Fourier (DFT)** di x è il vettore $\hat{x}$ la cui k-esima componente è

$$\hat{x}_k = \frac{1}{\sqrt{n}} \sum_{j=1}^{n} x_j e^{-2\pi\hat{i}(j-1)(k-1)/n}.$$

La trasformata inversa di $\hat{x}$ è il vettore x, dove

$$x_j = \frac{1}{\sqrt{n}} \sum_{k=1}^{n} x_k e^{2\pi\hat{i}(j-1)(k-1)/n}.$$

[4]In MATLAB le funzioni `fft` e `ifft` agiscono sulle matrici. Per calcolare il prodotto Fx bisogna recuperare la matrice X dal vettore $x = \mathrm{vec}(X)$ ed eseguire `fft(X)` e `ifft(X)`.

Per esempio

$$b = Ax = F^*\Lambda Fx$$

puó essere calcolata in MATLAB con i seguenti comandi:

```
S=fft2(circshift(P, 1 - center));
B=ifft2(S .* fft2(X));
B=real(B);
```

La funzione `circshift` esegue uno shift sugli elementi della matrice e serve a selezionare la prima colonna di A trasformandola in matrice; `real` viene usato perchè le funzioni `fft2` e `ifft2` coinvolgono numeri complessi e, a causa di errori di rounding, B potrebbe contenere numeri complessi.

Ora si vuole risolvere il sistema lineare $b = Ax$, assumendo l'esistenza di A^{-1}. Il problema

$$x = A^{-1}b = F^*\Lambda^{-1}Fb$$

si risolve usando i comandi

```
S=fft2(circshift(P, 1 - center));
X=ifft2(fft2(B) ./ S);
X=real(X).
```

Capitolo 3

Metodi di regolarizzazione per la ricostruzione di immagini

Il problema della ricostruzione di immagini è un problema malposto, è un problema cioè per il quale non esiste un'unica soluzione per ogni dato iniziale o per il quale la soluzione non dipende in modo continuo dai dati.

Si è visto nel capitolo precedente che il processo di acquisizione e registrazione di un'immagine digitale è descritto dal modello lineare

$$y = Ax + e. \tag{3.1}$$

Data l'immagine registrata y, il rumore e e la matrice A, si cerca una buona approssimazione dell'immagine reale x, tenendo presente che i valori di x sono non negativi. Tuttavia la matrice A è mal condizionata, infatti i valori singolari di A decadono gradualmente a zero.

Il fatto che il problema sia malposto non implica che non si possa trovare una buona soluzione. Si puó ricorrere infatti alle tecniche di *regolarizzazione*, che consistono nell'utilizzo di metodi particolari per il calcolo di una soluzione significativa. Esse rinunciano a trovare la soluzione esatta del problema (3.1) e calcolano la soluzione di un problema leggermente diverso ma meglio condizionato.

Il sistema (3.1) viene cosí sostituito dal problema di minimo

$$\begin{aligned} \min \quad & J(x) = J_0(x) + \lambda J_R(x) \\ & x \geq 0, \end{aligned} \tag{3.2}$$

dove λ è il *parametro di regolarizzazione* che controlla il peso da dare all'uno o all'altro termine.

Il termine di data fitting $J_0(x)$ dipende dal rumore sui dati: se è di tipo gaussiano, si utilizza il metodo dei minimi quadrati, cioè si cerca di minimizzare il residuo

$$||Ax - b||^2.$$

Se il rumore è di tipo poissoniano è conveniente usare la divergenza di Kullback-Leibler[1].

Il piú famoso metodo di regolarizzazione è quello di Tikhonov, in cui la soluzione regolarizzata x_λ è il minimo di una combinazione pesata tra la norma del residuo e il termine $\Omega(x) = ||L(x - x^*)||_2^2$

$$x_\lambda = \min\{||Ax - b||_2^2 + \lambda||L(x - x^*)||_2^2\},$$

dove x^* è una stima iniziale della soluzione.

Un secondo metodo è quello della Variazione Totale, in cui il termine di regolarizzazione J_R è

$$J_R(x) = \int \sqrt{|\nabla x|^2} dx.$$

$J_R(x)$ cosí definito non è differenziabile. Per questo viene aggiunto un secondo parametro β, tipicamente piccolo, e il termine di regolarizzazione diventa

$$J_R(x) = \int \sqrt{|\nabla x|^2 + \beta^2} dx.$$

[1] In teoria della probabilitá la divergenza di Kullback-Leibler è una misura non simmetrica della differenza tra due distribuzioni di probabilitá P e Q. Nel caso di distribuzioni di probabilitá discrete P e Q la divergenza di K-L è definita come

$$D_{KL}(P||Q) = \sum_i P(i) log \frac{P(i)}{Q(i)}.$$

Tipicamente P rappresenta la vera distribuzione dei dati osservati, o una precisa formulazione teorica, e Q rappresenta una teoria, un modello che descrive P.

Sono stati proposti diversi algoritmi per la risoluzione di (3.2). Nelle prossime sezioni si studieranno il metodo Fast Projected Quasi-Newton (QNP) e il metodo Projected Newton Coniugate Gradient (PNCG), che rientrano nella classe dei metodi di proiezione a due metriche. Per ulteriori approfondimenti si rimanda rispettivamente a [5] e [6].

3.1 Il metodo QNP

Si consideri il modello lineare (3.1) che descrive il processo di acquisizione di un'immagine, in cui e è il rumore di tipo Gaussiano e A è una matrice BTTB. Essendo un problema mal condizionato, si passa allo studio del problema di regolarizzazione di Tikhonov

$$\begin{aligned} \min \quad & J(x) = \frac{1}{2}||Ax - y||^2 + \frac{\lambda}{2}||x||^2 \\ & x \geq 0, \end{aligned}$$

dove λ è il parametro di regolarizzazione.

La chiave del metodo proposto, che appartiene alla classe dei metodi di Newton proiettato, sta nel fatto che la matrice hessiana di $J(x)$ è approssimata da una matrice circolante. Grazie a questa approssimazione ogni iterazione puó essere facilmente calcolata con la trasformata veloce di Fourier.

Si consideri la matrice hessiana di $J(x)$

$$\nabla^2 J(x) = \lambda I + A^T \cdot A.$$

Sia C una matrice BCCB tale che $C^T \cdot C$ sia una matrice BCCB che approssima $A^T \cdot A$. Allora la matrice

$$Q = \lambda I + C^T \cdot C$$

è ancora una matrice BCCB ed è un'approssimazione di $\nabla^2 J(x)$. In particolare, l'approssimazione dell'hessiana Q è simmetrica e definita positiva e puó essere facilmente invertita usando la trasformata veloce di Fourier FFT.

In seguito si indicherá con g_k il gradiente di $J(x)$ calcolato all'iterato x_k e $N = n^2$. Per ogni iterato $x_k \geq 0$ si definisce

$$I(x_k) = \left\{ i \mid 0 \leq x_{ki} \leq \epsilon_k \ \text{ e } \ \frac{\partial J(x_k)}{x_i} > 0 \right\},$$

dove $\epsilon_k = \min\{\epsilon,\ \omega_k\}$, con $\omega_k = |x_k - [x_k - \nabla J(x_k)]^+|$ e ϵ indica un parametro positivo. Si dice infine che una matrice simmetrica D è *diagonale rispetto all'insieme di indici* $L \subset \{1, 2, ...N\}$ se

$$D_{ij} = 0, \quad \forall i \in L, \quad j = 1, 2, ..., N, \quad i \neq j.$$

Il metodo proposto per la risoluzione del problema di regolarizzazione di Tikhonov vincolato ha la forma generale

$$x_{k+1} = [x_k - \alpha_k p_k]^+,$$

dove α_k è il passo e p_k la direzione di discesa. Alla k-esima iterazione, p_k è calcolata come

$$p_k = D_k g_k,$$

dove D_k è diagonale rispetto a $I(x_k)$ ed è cosí definita:

$$D_{kij} = \begin{cases} \delta_{ij} & \text{se } i \in I(x_k) \text{ o } j \in I(x_k) \\ Q_{ij}^{-1} & \text{altrimenti} \end{cases}$$

Q è la matrice circolante che approssima $\nabla^2 J(x)$.

Considerando $v \in R^n$ e il corrispondente $I(v)$, si chiama `reduce` l'operatore tale che

$$\{\texttt{reduce}(v)\}_i = \begin{cases} v_i & \text{se } i \notin I(v) \\ 0, & \text{altrimenti} \end{cases}$$

con $i = 1, ..., N$.

Allora

$$p_k = (g_k - reduce(g_k)) + reduce(Q^{-1}(reduce(g_k))).$$

Infatti, se $i \in I_k$ $p_{ki} = g_{ki}$, mentre se $i \notin I_k$ si ha

$$p_{ki} = \sum_{j \notin I_k} \{D_k\}_{ij} g_{kj} = \sum_{j=1}^{n} \{Q^{-1}\}_{ij} \{reduce(g_k)\}_j.$$

Il passo α_k è invece calcolato con la regola di Armijo vista nel capitolo precedente.

Per dimostare che l'algoritmo presentato è ben definito e convergente occorre dimostrare la seguente proposizione:

Proposizione 3.1. *Le matrici D_k sono definite positive e soddisfano*

$$\mu_1||z||^2 \leq z^t D_k z \leq \mu_2||z||^2,$$

$\forall z \in R^n$, $k = 0, 1, 2, ...,$ *per alcuni scalari positivi* μ_1 *e* μ_2.

Si riassume ora l'algorimo **Fast Projected Quasi-Newton Method (QNP)**:

- Si sceglie $x_0 \geq 0$
- Si calcolano I_0, g_0 e D_0
- `for` $k = 0, 1, 2, ...$
  - si calcola la direzione p_k
  - si calcola il passo α_k
  - si aggiorna $x_{k+1} = [x_k - \alpha_k p_k]^+$ e si calcolano I_{k+1}, g_{k+1} e D_{k+1}; infine si imposta $k = k + 1$.

`end`

Le iterazioni terminano quando è soddisfatto uno dei seguenti criteri d'arresto:

- la distanza relativa tra due iterazioni successive

$$\frac{|J(x_k) - J(x_{k-1})|}{|J(x_k)|}$$

  è minore di una data tolleranza τ;
- si raggiunge il massimo numero di iterazioni consentito, $K_{\max}$.

3.2 Il metodo PNCG

Si vuole ora restaurare un'immagine digitale affetta da un errore di tipo Poissoniano. La miglior funzione obiettivo che minimizza questo tipo di rumore è costituita dal logaritmo negativo della funzione di verosimiglianza di Poisson, la divergenza di Kullback-Leibler.

Il problema di minimizzazione della divergenza di Kullback-Leibler è un problema mal condizionato, pertanto è necessario adottare una tecnica di regolarizzazione. In questo caso si sceglie un funzionale di regolarizzazione di variazione totale, che ben si adatta, ad esempio, all'edge detection di immagini mediche.

Si presenta ora il metodo PNCG, una versione speciale del metodo di Newton proiettato di Bertsekas, che si applica nel caso in cui J_0 è la divergenza di Kullback-Leibler e J_R è il funzionale di variazione totale. Per migliorare l'efficienza computazionale del metodo si vedrá anche una strategia di preconditioning.

Si consideri il problema di minimo (3.2). L'incognita x non solo deve essere strettamente positiva, ma deve anche essere piú grande di una costante γ che è piú piccola del rumore di background: $x \geq \gamma$. Cambiando la variabile da x a $x - \gamma$ il problema (3.2) è sostituito da

$$\begin{aligned} \min \quad & J(x,\gamma) = J_0(x,\gamma) + \lambda J_R(x) \\ & x \geq 0, \end{aligned} \tag{3.3}$$

dove J_0 è la divergenza di K-L

$$J_0(x,\gamma) = \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\left\{ y_{i,j} \ln \frac{y_{i,j}}{x_{i,j}+\gamma} + x_{i,j} + \gamma - y_{i,j} \right\}$$

e $J_R(x)$ è il funzionale discreto di variazione totale

$$J_R(x) = \frac{1}{N_x N_y} \sum_{i=1}^{N_x}\sum_{j=1}^{N_y} \sqrt{|\nabla x_{i,j}|^2 + \beta},$$

con β piccolo parametro positivo.

$J_0(x,\gamma)$ e $J_R(x)$ sono entrambe convesse. In particolare $J_0(x,\gamma)$ è strettamente convessa se e solo se $y_{i,j}$ è positivo per ogni i, j mentre $J_R(x)$ è strettamente convessa se y non è un vettore costante. Allora $J(x,\gamma)$ è strettamente convessa e quindi esiste una soluzione a (3.3).

Data la stima iniziale x_0, l'iterazione generale del metodo PNCG è

$$x_{k+1} = [x_k - \alpha_k p_k]^+,$$

dove α_k è il passo e p_k la direzione di discesa. Si indica con g_k il gradiente di $J(x)$ calcolato all'iterato x_k. Per ogni iterato $x_k \geq 0$ si definisce

$$I(x_k) = \left\{ i \mid 0 \leq (x_k)_i \leq \epsilon_k \ \text{ e } \ g_{k\,i,j} > 0 \right\},$$

dove $\epsilon_k = \min\{\epsilon,\ \omega_k\}$, con $\omega_k = ||x_k - [x_k - g]^+||$ e ϵ indica un parametro positivo.

Sia g_k^I il *gradiente ridotto* definito come

$$\{g_k^I\}_{i,j} = \begin{cases} g_{kij} & \text{se } (i,j) \notin I_k \\ 0, & \text{altrimenti.} \end{cases}$$

Sia H_k l'approssimazione simmetrica definita positiva della matrice hessiana di J:

$$H_k := D_k + \lambda L(x_k), \qquad \forall k = 0, 1, 2, ...$$

dove

$$D_k = \text{diag}(y./(x_k + \gamma).^2),$$

e l'operatore $L(x)$ è definito da

$$< L(x)v, v >:= \frac{1}{N_x N_y} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \frac{< \nabla v_{i,j}, \nabla v_{i,j} >}{\sqrt{|\nabla x_{i,j}|^2 + \beta}},$$

per $v \in R^{N_x \times N_y}$.

Si consideri ora il sistema lineare

$$H_k d = g_k^I. \tag{3.4}$$

La direzione p_k è data da

$$p_{ki,j} = \begin{cases} d_{kij} & \text{se } i \notin I_k \\ g_{kij} & \text{altrimenti} \end{cases} \qquad i = 1, ..., N_x, \quad j = 1, ..., N_y,$$

dove d_k è la matrice della soluzione approssimata di (3.4) calcolata con il metodo del gradiente coniugato troncato, i cui criteri di arresto sono discussi in seguito.

Il passo α_k è scelto con la regola di Armijo.

Si riassume ora l'algorimo **Projected Newton Coniugate Gradient (PN-CG)**:

- Si sceglie $x_0 \geq 0$
- Si calcolano I_0, g_0^I e H_0
- `for` $k = 0, 1, 2, ...$
  - si calcola la direzione p_k
  - si calcola il passo α_k
  - si aggiorna $x_{k+1} = [x_k - \alpha_k p_k]^+$ e si calcolano I_{k+1}, g_{k+1}^I e H_{k+1}; infine si imposta $k = k + 1$.

`end`

Criteri di arresto In questo caso esistono due diversi criteri d'arresto, interni ed esterni.

Criteri di arresto esterni Le iterazioni terminano quando

- la distanza relativa tra due iterazioni successive è minore di una data tolleranza τ,
$$\frac{|J(x_k) - J(x_{k-1})|}{|J(x_k)|} \leq \tau;$$
- si raggiunge il massimo numero di iterazioni consentito, $K_{\max}$.

Criteri di arresto interni Le iterazioni CG per la risoluzione di (3.4) terminano quando

- la precisione relativa è minore della tolleranza data τ_{CG}, cioè

$$||r_{k,l}|| \leq \tau_{CG}||g_{Ik}||, \quad \tau_{CG} \in (0,1),$$

  dove $r_{k,l}$ è il residuo del CG.

- si raggiunge il massimo numero di iterazioni consentito, K_{CGmax}.

Preconditioning Al fine di rendere l'algoritmo piú efficiente per problemi di grandi dimensioni, si introduce un precondizionamento diagonale per risolvere il sistema (3.4) con il metodo del gradiente coniugato.

Nonostante la sua semplicitá, il precondizionamento diagonale è molto efficace nel ridurre il numero di iterazioni necessarie a raggiungere la tolleranza predefinita.

Capitolo 4

L'interfaccia grafica

L'interfaccia grafica serve ad eseguire gli algoritmi del pacchetto NPtool[1], algoritmi per la risoluzione del problema di minimo

$$\begin{aligned} \min \quad & J_0(x) + \lambda J_R(x) \\ & x \geq 0 \end{aligned} \tag{4.1}$$

che sorge nella ricostruzione di immagini digitali.
Come piattaforma di sviluppo è stato utilizzato MATLAB 7.4

4.1 Guida all'utente

Per far partire l'interfaccia grafica è necessario digitare il nome della stessa nella shell di MATLAB. All'apertura della finestra grafica si osserva che l'interfaccia è suddivisa in quattro parti.

La prima parte riguarda il caricamento dei dati, la seconda la scelta del metodo, la terza è costituita da un piccolo pannello per salvare i dati di output in un MAT-file e la quarta da una finestra con le informazioni relative all'algoritmo scelto.

Una volte cliccato uno dei due pulsanti *Sumbit* si apre una seconda finestra che carica l'immagine reale, quella perturbata, l'immagine ricostruita e l'im-

[1]Disponibile su `www.dm.unibo.it/~piccolom/`

magine errore. Infine vengono stampate in una finestra alcune informazioni relative alla complessitá dell'algoritmo scelto.

4.1.1 Primo passo: caricamento dell'immagine

Per caricare un'immagine ci sono varie possibilitá. Si illustrano ora i passi necessari al caricamento:

1. nel pannello *Data Type* scegliere
   - **Simulated Data** per lavorare su un problema-test, e quindi disporre anche dell'immagine originale;
   - **Real Data** se non si dispone dell'immagine originale e si cerca di ripulire l'immagine perturbata;
2. nel pannello *File Type* scegliere tra le seguenti modalitá di caricamento:
   - **MatFile** se si dispone di tutti i dati in un unico MatFile;
   - **Image** se si preferisce caricare separatamente l'immagine e le funzioni relative al blur e al noise;
3. dal menú a tendina selezionare l'immagine da caricare;
4. cliccando **Submit** i dati vengono effettivamente caricati: si apre una nuova finestra nella quale viene visulizzata l'immagine perturbata ed eventualmente, nel caso del problema test, l'immagine reale.

Si passa ora alla scelta dell'algoritmo.

4.1.2 Secondo passo: la scelta del metodo

Il secondo riquadro dell'interfaccia principale riguarda il metodo per il restauro dell'immagine. L'utente deve seguire questi passi:

1. selezionare innanzitutto il problema su cui lavora, scegliendo tra il denoising e il deblurring e, a seconda del tipo di rumore, poissoniano

o gaussiano, scegliere rispettivamente tra la divergenza di Kullback-Leibler (KL) e il metodo dei minimi quadrati (LS). Le possibilitá dunque sono:

- **KL + denoising**;
- **KL + deblurring**;
- **LS + denoising**;
- **LS + deblurring**;

2. selezionare poi la funzione di regolarizzazione tra:
   - il funzionale di variazione totale (**TV**);
   - il funzionale di Tikhonov (**Tikh**), scelta possibile solo nei problemi di deblurring;
3. scegliere uno tra i seguenti algoritmi:
   - **Fast Projected Quasi-Newton (QNP)**, utilizzabile solo nel problema di deblurring con divergenza di KL e funzionale di regolarizzazione di Tikhonov;
   - **Newton Projection (NP)**, attivo solo nel problema di deblurring con metodo del minimi quadrati (LS) e funzionale di regolarizzazione di Tikhonov;
   - **Projected Newton Coniugate Gradient (PNCG)**, attivo in tutti i casi. In questo caso si puó anche scegliere se usare il precondizionamento o meno, scegliendo'**preconditioning**'.
4. indicare:
   - il parametro di regolarizzazione **Regularization parameter**, impostato per default a 10^{-4};
   - la tolleranza (**tol**), per default 10^{-4}.
5. cliccare **Submit** per eseguire l'algoritmo.

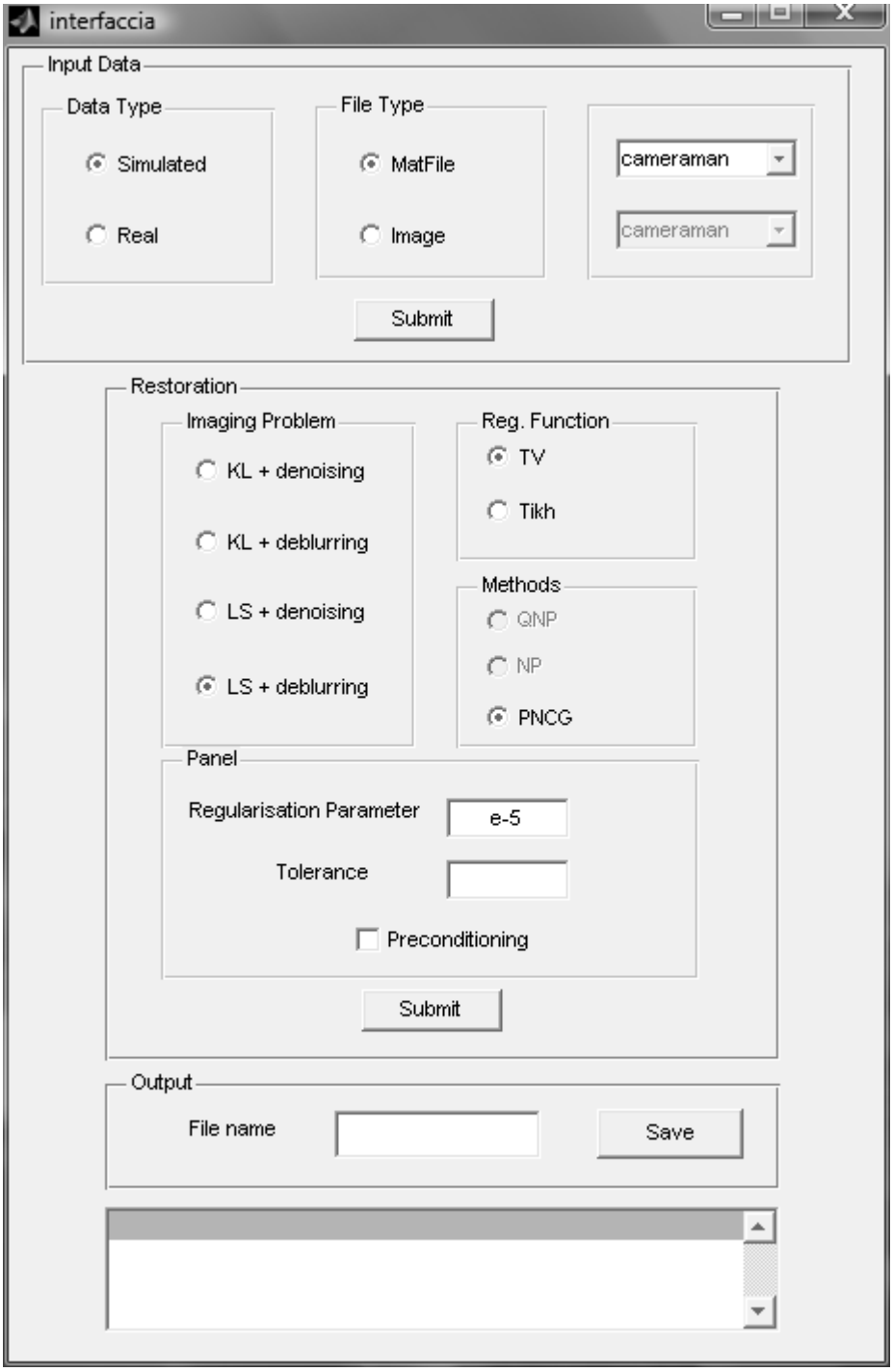


Figura 4.1: Caricamento dell'immagine e scelta del metodo

4.1.3 Output

Una volta cliccato il pushbutton *Submit*, nella seconda finestra appaiono l'immagine ricostruita e, nel caso del problema test, l'immagine errore.

Nel riquadro inferiore vengono stampate le informazioni sulla complessità e sull'efficienza del metodo scelto. Indicando con x_{ij} un elemento dell'immagine reale e con y_{ij} un elemento dell'immagine ricostruita, con $i = 1, ..., n$, $j = 1, ..., m$, si hanno:

- l'errore relativo:

$$\text{err} = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m} \frac{(x_{ij} - y_{ij}).^2}{\sum_{i=1}^{n}\sum_{j=1}^{m} y_{ij}^2}};$$

- il numero di trasformate di Fourier (FFT) eseguite;

- Structural Similarity Index: la funzione `ssim` di MATLAB fornisce un indice per confrontare due immagini. Se una delle due è considerata di qualitá perfetta, l'indice puó essere considerato come una misura della qualitá dell'altra immagine. Se le due immagini sono identiche allora `ssim` vale uno;

- Mean-Squared Error (MSE):

$$\text{MSE} = \sum_{i=1}^{n}\sum_{j=1}^{m} \frac{(x_{ij} - y_{ij})^2}{n \cdot m};$$

- Peak Signal-to-Noise ratio (PSNR): dato L, il massimo valore possibile dei pixel, si calcola

$$\text{PSNR} = 10 \cdot \log_{10} \frac{L^2}{\text{MSE}};$$

- Mean Absolute Error (MAE):

$$\text{MAE} = \sum_{i=1}^{n}\sum_{j=1}^{m} \frac{|x_{ij} - y_{ij}|}{n \cdot m}.$$

Sull'interfaccia principale, l'utente ha la possibilità di salvare su un Mat-file l'output del processo, cioè l'immagine restaurata e altre informazione relative al metodo scelto, digitando il nome del file e cliccando su **Save**. In fondo alla finestra principale viene anche stampato il nome del metodo utilizzato.

4.2 Esempi

In questa sezione si elencano alcuni esempi di utilizzo dell'interfaccia, caricando un problema test. In questo modo è possibile fare un confronto con l'immagine originale e visualizzare l'immagine errore. Il parametro di regolarizzazione sará indicato con λ.

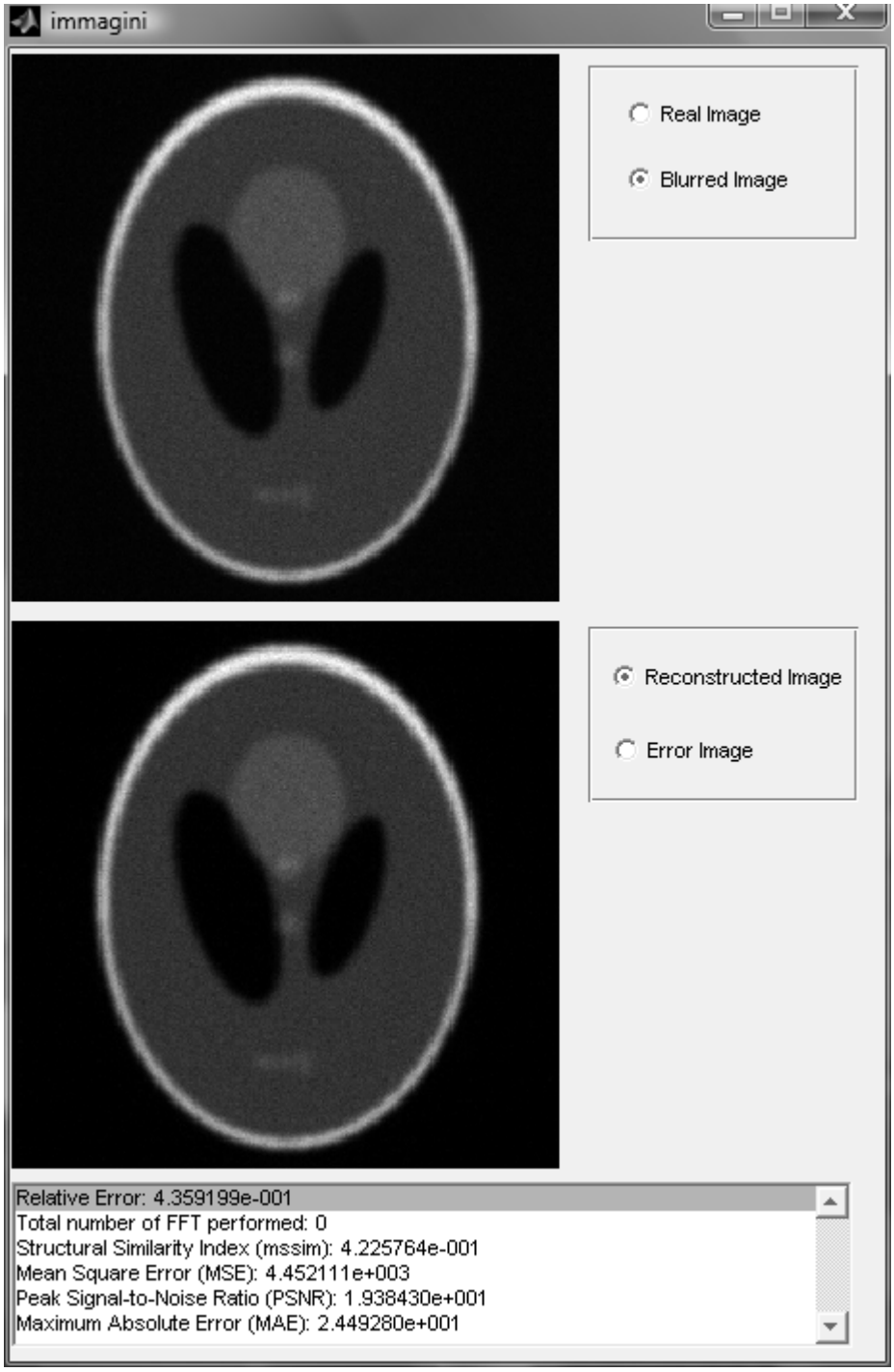


Figura 4.2: KL denoising + TV + PNCG - $\lambda = 10^{-5}$, tol $= 10^{-4}$

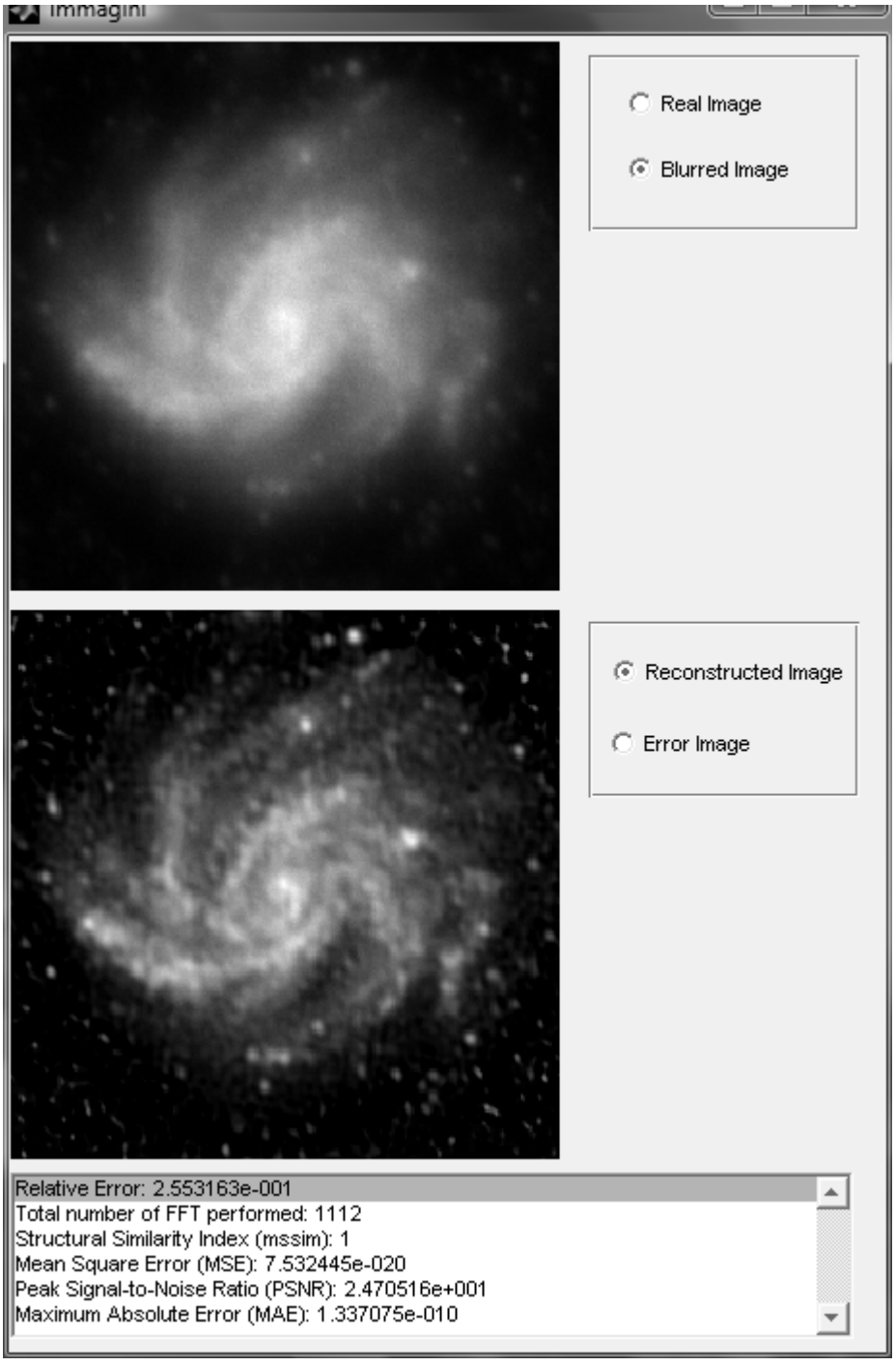


Figura 4.3: KL deblurring + TV + PNCG - $\lambda = 10^{-4}$, tol $= 10^{-3}$

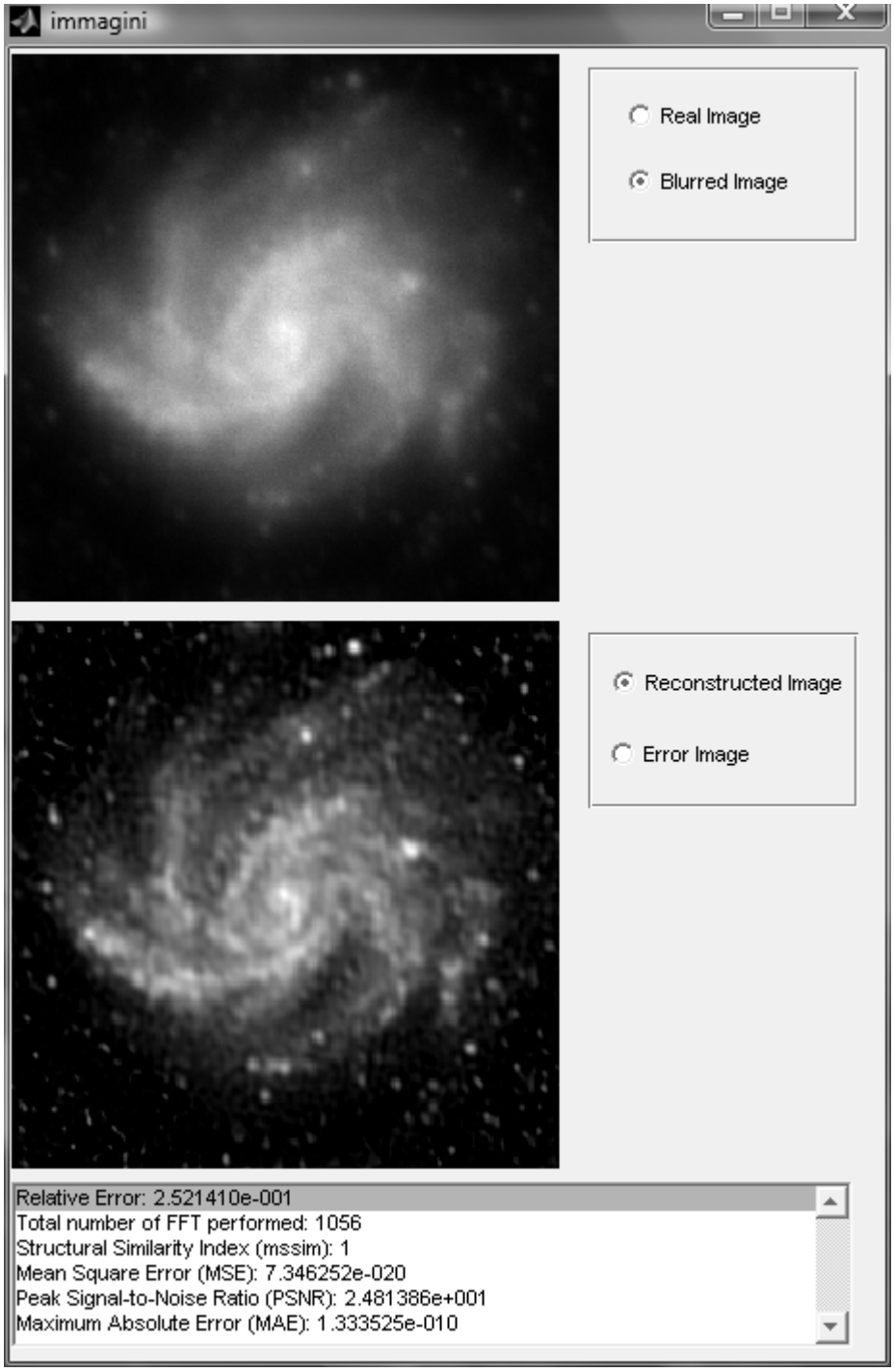


Figura 4.4: KL deblurring + Tikh + PNCG - $\lambda = 10^{-4}$, tol $= 10^{-4}$

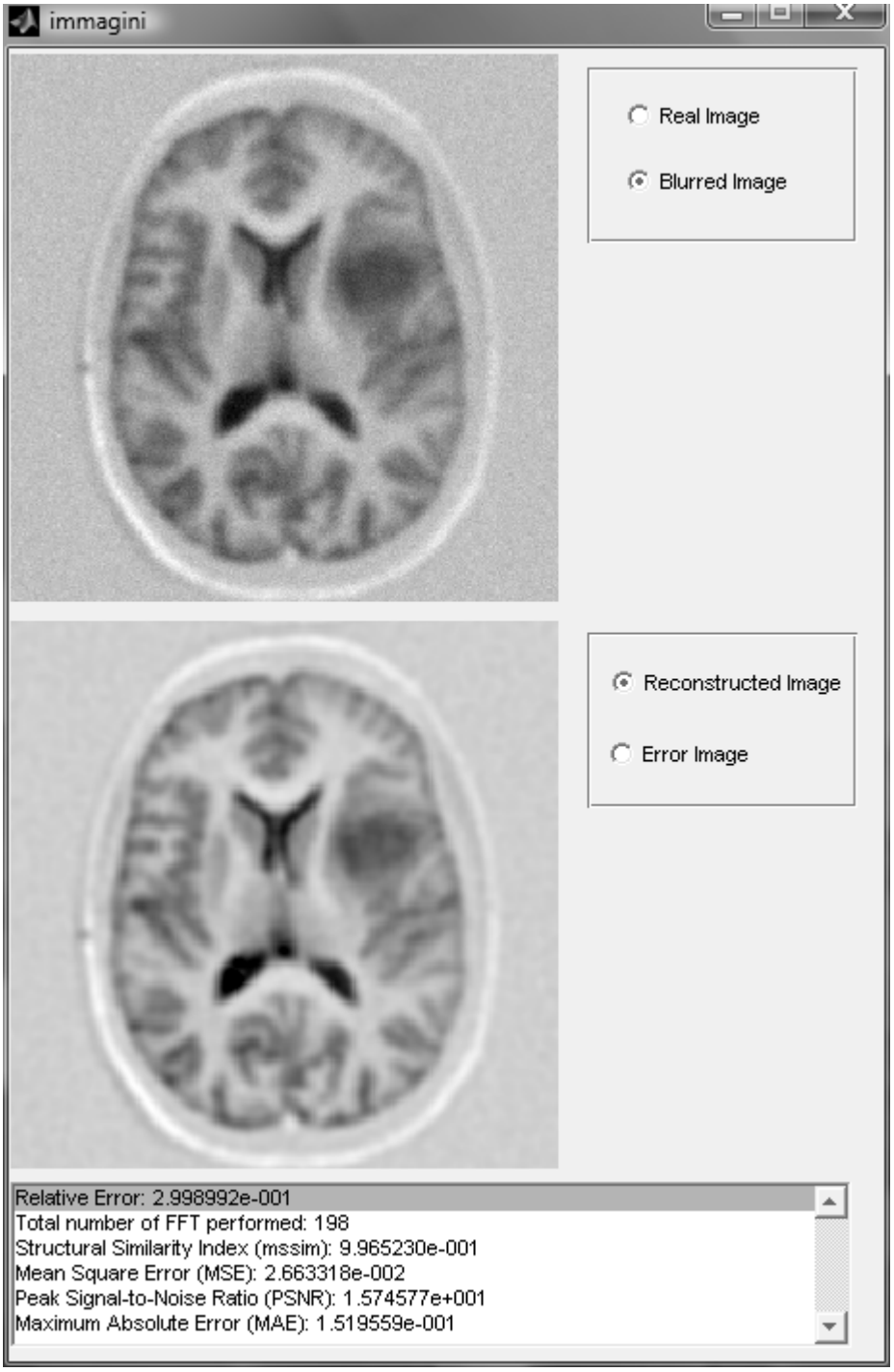


Figura 4.5: KL deblurring + Tikh + QNP - $\lambda = 1$, tol $= 10^{-2}$

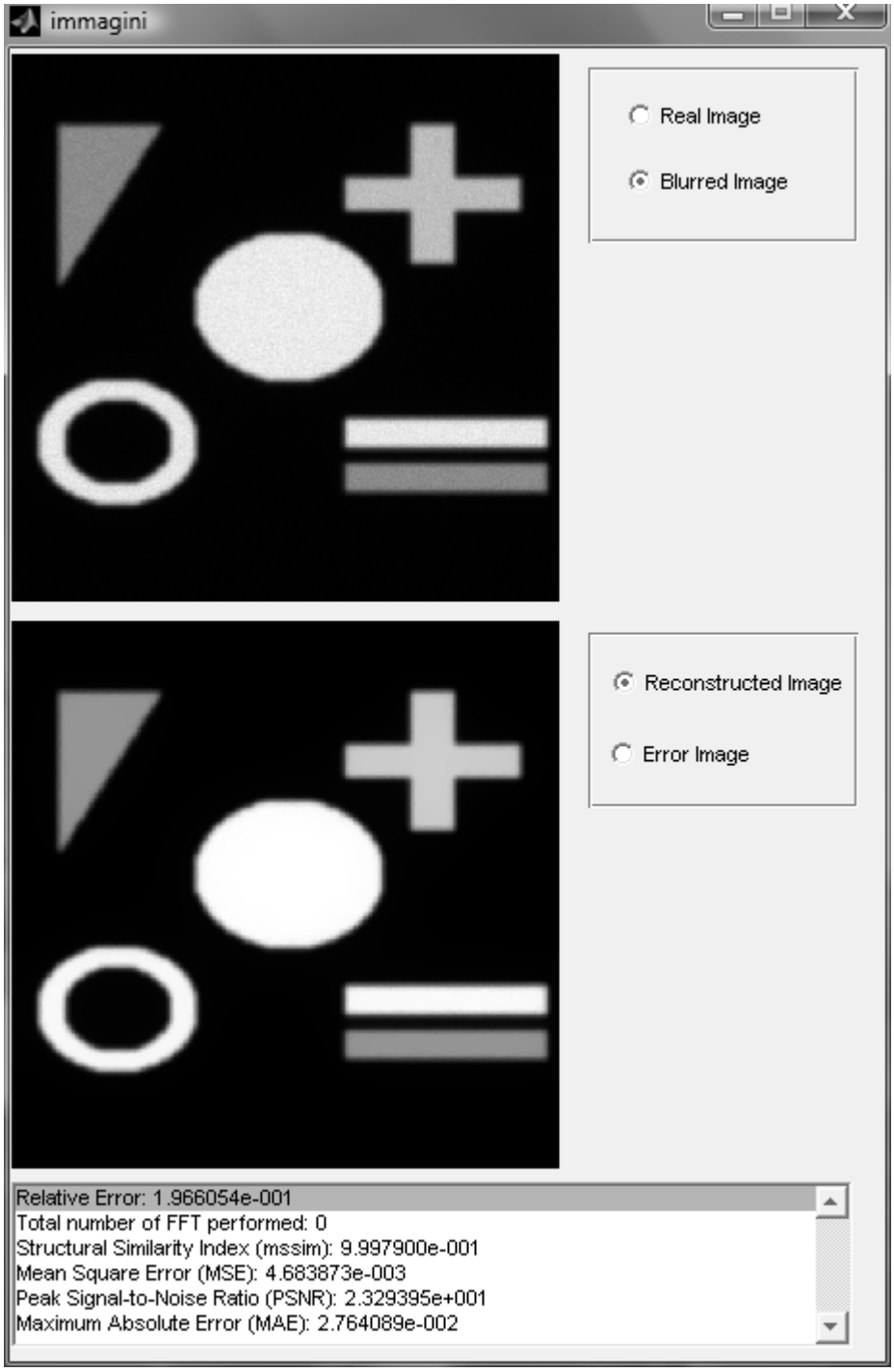


Figura 4.6: LS denoising + TV + PNCG - $\lambda = 10$, tol $= 10^{-1}$

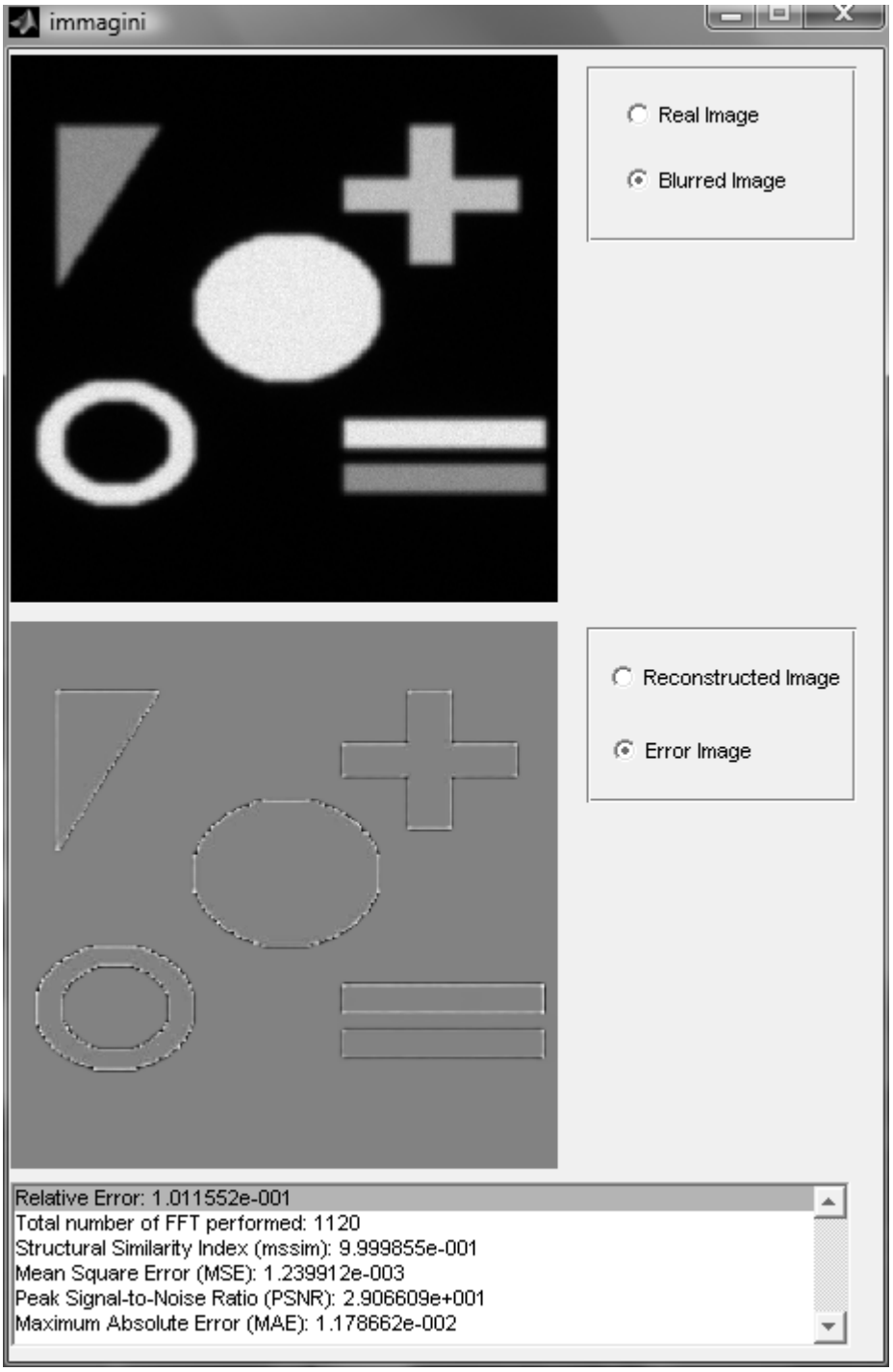


Figura 4.7: Immagine errore: LS deblurring + TV + PNCG - $\lambda = 10$, tol $= 10^{-2}$

Figura 4.8: Immagine errore: LS deblurring + Tikh + PNCG - $\lambda = 10$, tol $= 10^{-1}$

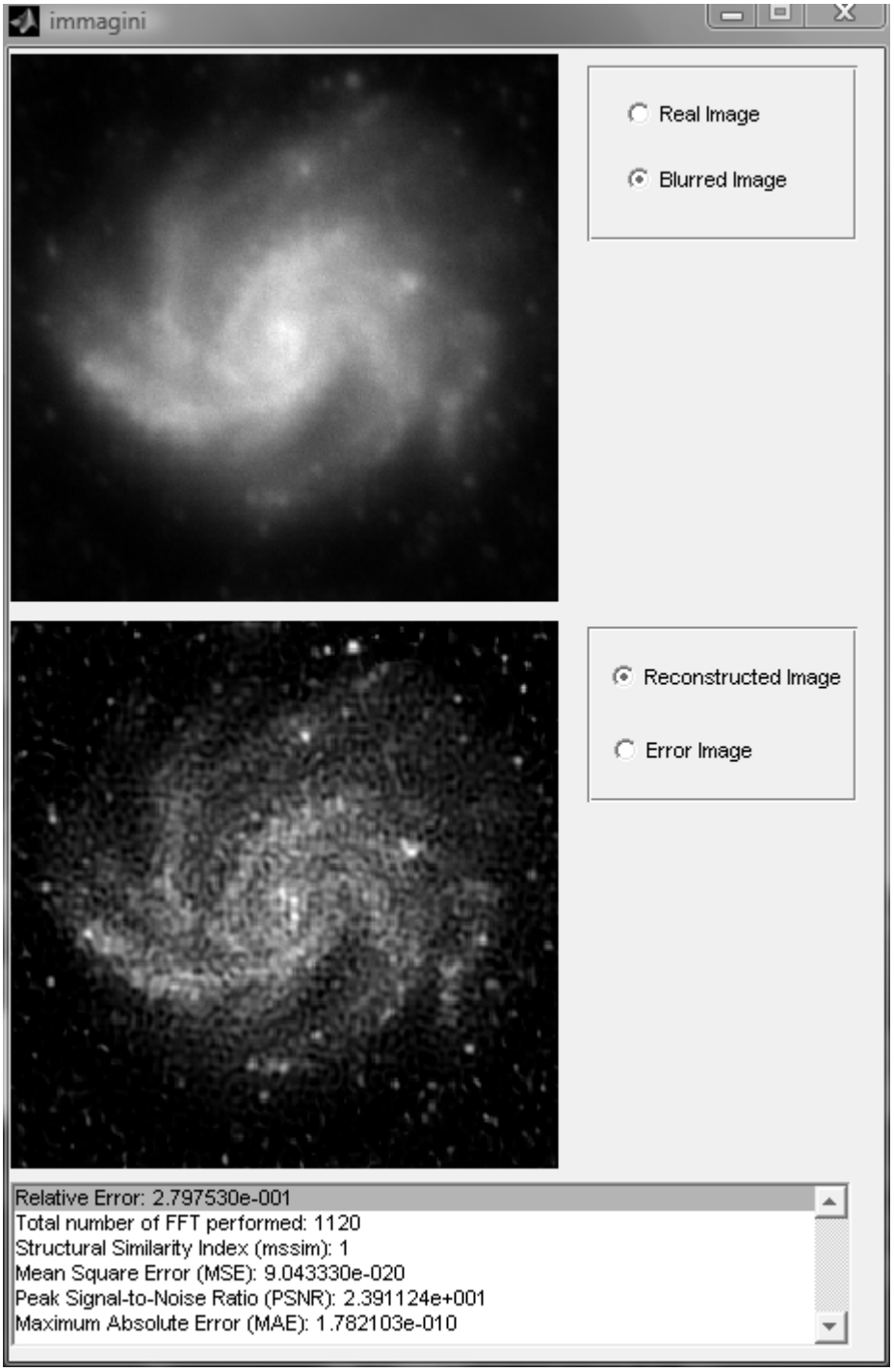


Figura 4.9: LS deblurring + Tikh + NP - $\lambda = 1$, tol $= 10^{-1}$

Conclusioni

In questa tesi si è creata un'interfaccia grafica per l'utilizzo di algoritmi di ricostruzione di immagini digitali.

L'interfaccia permette di usare agevolemente i diversi algoritmi senza dover digitare i comandi nella shell.

Nella finestra principale dell'interfaccia, l'utente carica i dati di input e sceglie il metodo da utilizzare. Potrá scegliere tra un problema test e un'immagine reale, caricare i dati da un unico MatFile o da file separati e infine indicare l'immagine da caricare. L'utente inoltre seleziona il tipo di problema tra deblurring e denoising, il funzionale di regolarizzazione e il tipo di algoritmo, ed ha la possibilitá di digitare il valore del parametro di regolarizzazione e della tolleranza.

Nella finestra secondaria appaiono le immagini caricate, l'immagine ricostruita e l'immagine errore. In questa finestra sono anche stampate alcune informazioni sulla complessitá e sull'efficienza dell'algoritmo scelto. C'è infine la possibilitá di salvare i dati di output in un MatFile.

Bibliografia

[1] Dimitri P. Bertsekas, *Constrained Optimization and Lagrange multiplier methods*, Academic Press Inc., 1982

[2] Dimitri P. Bertsekas, *Nonlinear programming*, second edition - Belmont, MA: Athena scientific, 1999

[3] Micheal T. Heath, *Scientific Computing, an introductory survey*, second edition, McGraw Hill, 2005

[4] Christian Hansen, James G. Nagy, Dianne P. O'Leary, *Deblurring images: matrices, spectra and filtering*, Philadelphia: SIAM, 2006

[5] G. Landi, E. Loli Piccolomini, *A Fast Projected Quasi-Newton Method for Nonnegative Tikhonov Regularization*, International Journal of Mathematics and Computer Science, 2008, no. 3, 201-205

[6] E. Loli Piccolomini, G. Landi, *An efficien method for nonnegatively constrained Total Variation-based denoising of medical images corrupted by Poisson noise*, Submitted

[7] Mario Bertero, Henri Lantéri, Luca Zanni *Iterative image reconstruction: a point of view*, Mathematical methods IMRT, CRM series, 7, 37-63, edizioni della Normale, Pisa, 2008

[8] Duane C. Hanselman, Bruce L. Littlefield, *Mastering Matlab 7*, Upper Saddle River (NJ) : Pearson Education, 2005