

# Analisi e applicazioni dell'algoritmo LSQR

Gianluca Iacullo

29 agosto 2025

## Sommario

In questo report viene analizzato l'algoritmo LSQR, un metodo iterativo per la risoluzione di problemi ai minimi quadrati, particolarmente adatto al trattamento di matrici di grandi dimensioni e sparse. Dopo un'introduzione teorica e un richiamo al lavoro di C. C. Paige e M. A. Saunders [1], viene presentata la formulazione dell'algoritmo e la sua applicazione al problema del deblurring di immagini sfocate, tipicamente mal condizionato. In tale contesto LSQR si dimostra numericamente stabile e affidabile. Viene inoltre proposta una variante del metodo in grado di affrontare il problema dei minimi quadrati regolarizzato, consentendo la scelta del parametro di regolarizzazione  $\lambda$ . Infine, vengono discusse le condizioni in cui l'introduzione di un parametro  $\lambda$  non nullo permette di migliorare i risultati rispetto al caso non regolarizzato.

## Indice

<b>1</b>	<b>Introduzione</b>	<b>1</b>
<b>2</b>	<b>L'Algoritmo LSQR</b>	<b>2</b>
2.1	Riformulazione del problema dei minimi quadrati regolarizzato . . . . .	2
2.2	Algoritmo LSQR per il problema non regolarizzato . . . . .	3
2.3	Algoritmo LSQR per il problema regolarizzato . . . . .	5
<b>3</b>	<b>Implementazione</b>	<b>6</b>
<b>4</b>	<b>Conclusione</b>	<b>8</b>

## 1 Introduzione

Il metodo LSQR è un algoritmo iterativo per la risoluzione di sistemi lineari e di problemi ai minimi quadrati.

Alla base dell'algoritmo vi è la procedura di bidiagonalizzazione di Golub-Kahan, grazie alla quale viene generata una sequenza di approssimazioni  $\{x_k\}$ . In tale sequenza, la norma del residuo

$$r_k = b - Ax_k,$$

decresce monotonamente in norma euclidea, ovvero  $\|r_k\|_2$  tende a ridursi a ogni iterazione.

Il problema ai minimi quadrati che consideriamo ha la forma

$$\min_x \|Ax - b\|_2,$$

dove la matrice  $A$  non è disponibile in forma esplicita, ma solo attraverso un operatore che ne simula l'azione. In questo scenario l'algoritmo LSQR risulta particolarmente adatto, poiché richiede esclusivamente il calcolo di prodotti del tipo  $Av$  e  $A^T v$ , senza la necessità di memorizzare l'intera matrice  $A$ .

Il contesto applicativo che analizzeremo riguarda il *deblurring* di immagini sfocate, un problema tipicamente mal condizionato. In tale situazione, l'operatore  $A$  rappresenta l'effetto di sfocatura, modellato come una convoluzione bidimensionale con un kernel generato casualmente. L'implementazione pratica di questa operazione è affidata alla funzione `matvec.m`, che calcola il prodotto matrice-vettore  $Ax$ , simulando così l'effetto di sfocatura sull'immagine originale  $I$ .

## 2 L'Algoritmo LSQR

### 2.1 Riformulazione del problema dei minimi quadrati regolarizzato

Ci poniamo come obiettivo la risoluzione del problema dei minimi quadrati regolarizzato, che formuliamo come:

$$\min_x \left\| \begin{bmatrix} A \\ \lambda I \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_2 = \min_x \{ \|Ax - b\|_2^2 + \lambda^2 \|x\|_2^2 \}, \quad (1)$$

dove  $A$  è una matrice  $m \times n$ , con  $m \geq n$ , di rango massimo,  $b \in \mathbb{R}^m$ , e il parametro  $\lambda$  modula una penalizzazione sulla norma della soluzione  $x$ .

Il problema posto in questi termini può essere trasformato nel sistema (2). Possiamo dedurre l'equivalenza dei due problemi in due modi, da un lato tramite le equazioni normali, e dall'altro attraverso il tipico utilizzo della pseudoinversa di Moore-Penrose nella risoluzione del problema dei minimi quadrati, nel nostro caso della matrice  $\begin{bmatrix} A \\ \lambda I \end{bmatrix}$ .

$$\begin{bmatrix} I & A \\ A^T & -\lambda^2 I \end{bmatrix} \begin{bmatrix} r \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}, \quad (2)$$

in cui  $r = b - Ax$  rappresenta il vettore residuo.

Applicare il processo di Lanczos, tramite  $2k + 1$  iterazioni della procedura di bidiagonalizzazione di Golub-Kahan, così come descritto in [1], ci conduce ai sistemi:

$$\begin{bmatrix} I & B_k \\ B_k^T & -\lambda^2 I \end{bmatrix} \begin{bmatrix} t_{k+1} \\ y_k \end{bmatrix} = \begin{bmatrix} \beta_1 e_1 \\ 0 \end{bmatrix} \quad (3)$$

$$\begin{bmatrix} r_k \\ x_k \end{bmatrix} = \begin{bmatrix} U_{k+1} & 0 \\ 0 & V_k \end{bmatrix} \begin{bmatrix} t_{k+1} \\ y_k \end{bmatrix}$$

dove

$$U_{k+1} = [u_1, u_2, \dots, u_{k+1}], \quad V_k = [v_1, v_2, \dots, v_k], \quad B_k = \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \ddots & \alpha_k \\ & & & \beta_{k+1} \end{bmatrix}$$

sono calcolati mediante la procedura di bidiagonalizzazione, e si hanno le identità

$$t_{k+1} = \begin{bmatrix} B_k \\ \lambda I \end{bmatrix} y_k - \begin{bmatrix} \beta_1 e_1 \\ 0 \end{bmatrix}, \quad x_k = V_k y_k, \quad r_k = b - Ax_k.$$

Poiché  $r_{k+1} = U_{k+1} t_{k+1}$ , con  $U_{k+1}$  ortogonale, e il nostro obiettivo è minimizzare il residuo, il problema è equivalente a risolvere

$$\min \left\| \begin{bmatrix} B_k \\ \lambda I \end{bmatrix} y_k - \begin{bmatrix} \beta_1 e_1 \\ 0 \end{bmatrix} \right\|_2 \quad (4)$$

## 2.2 Algoritmo LSQR per il problema non regolarizzato

Per il momento consideriamo il problema senza introdurre il termine di regolarizzazione  $\lambda$  e introduciamo l'algoritmo LSQR per il problema dei minimi quadrati. Nel paragrafo successivo sarà descritta la versione generale.

Una volta riformulato il problema (1) in (4), il cuore del metodo LSQR consiste nella fattorizzazione QR. Notiamo infatti che essa è molto opportuna nel nostro contesto, dove  $B_k$  è bidiagonale superiore e quindi in forma di Hessenberg. Possiamo quindi realizzare la fattorizzazione QR di  $B_k$  adoperando rotazioni di Givens, molto convenienti dal punto di vista computazionale.

Sia  $B_k = Q_k^* \begin{bmatrix} R_k \\ 0 \end{bmatrix}$  la fattorizzazione QR di  $B_k$ . Si mostra in [1] che  $R_k$  è la matrice  $k \times k$  bidiagonale superiore prodotta da una seconda procedura di bidiagonalizzazione, ed ha la forma:

$$R_k = \begin{bmatrix} \rho_1 & \theta_2 & & 0 \\ & \rho_2 & \theta_3 & \\ & & \ddots & \ddots \\ 0 & & & \rho_k \end{bmatrix}$$

Considerando l'azione di  $Q_k$  anche sul termine noto  $\beta_1 e_1$  possiamo scrivere

$$Q_k = \begin{bmatrix} R_k & f_k \\ & \phi_{k+1} \end{bmatrix} = \left[ \begin{array}{cccccc|c} \rho_1 & \theta_2 & & & & & | & \phi_1 \\ \rho_2 & \theta_3 & & & & & | & \phi_2 \\ & \ddots & \ddots & & & & | & \vdots \\ & & \rho_{k-1} & \theta_k & & & | & \phi_{k-1} \\ \hline & & & & \rho_k & & | & \phi_k \\ & & & & & & | & \phi_{k+1} \end{array} \right]$$

Segue che ponendo  $D_k := V_k R_k^{-1} = [d_1 \dots d_k]$   
si ha

$$x_k = V_k R_k^{-1} f_k = D_k d_k = x_{k-1} + \phi_k d_k$$

e i  $d_k$  sono ottenibili da  $R_k^t D_k^t = V_k^t$  per sostituzione in avanti.  
Si hanno dunque

$$d_k = \frac{1}{\rho_k} (v_k - \theta_k d_{k-1}) \quad (5)$$

$$x_k = V_k R_k^{-1} f_k = D_k d_k = x_{k-1} + \phi_k d_k \quad (6)$$

Da queste relazioni segue l'algoritmo LSQR (1)

---

**Algorithm 1** Algoritmo LSQR per il problema dei minimi quadrati

---

```
1: Inizializzazione
2:  $\beta_1 u_1 = b$ 
3:  $\alpha_1 v_1 = A^T u_1$ 
4:  $w_1 = v_1$ 
5:  $x_0 = 0$ 
6:  $\bar{\phi}_1 = \beta_1$ 
7:  $\bar{\rho}_1 = \alpha_1$ 

8: for  $i = 1, 2, 3, \dots$  do ▷ Ripeti gli step 3–6
    9: Continua la bidiagonalizzazione
    10:  $\beta_{i+1} u_{i+1} = Av_i - \alpha_i u_i$ 
    11:  $\alpha_{i+1} v_{i+1} = A^T u_{i+1} - \beta_{i+1} v_i$ 

    12: Cotruisci e applica la prossima trasformazione ortogonale
    13:  $\rho_i = (\bar{\rho}_i^2 + \beta_{i+1}^2)^{1/2}$ 
    14:  $c_i = \bar{\rho}_i / \rho_i$ 
    15:  $s_i = \beta_{i+1} / \rho_i$ 
    16:  $\theta_{i+1} = s_i \alpha_{i+1}$ 
    17:  $\bar{\rho}_{i+1} = -c_i \alpha_{i+1}$ 
    18:  $\phi_i = c_i \bar{\phi}_i$ 
    19:  $\bar{\phi}_{i+1} = s_i \bar{\phi}_i$ 

    20: Aggiorna  $x, w$ .
    21:  $x_i = x_{i-1} + (\phi_i / \rho_i) w_i$ 
    22:  $w_{i+1} = v_{i+1} - (\theta_{i+1} / \rho_i) w_i$ 
```

---

### 2.3 Algoritmo LSQR per il problema regolarizzato

Esaminiamo adesso il problema nel caso  $\lambda \neq 0$ . Siamo ora interessati alla fattorizzazione QR della matrice

$$\begin{bmatrix} B_k \\ \lambda I \end{bmatrix}.$$

Riorganizziamo quindi l'algoritmo in modo da includere le rotazioni di Givens necessarie a eliminare le entrate  $\lambda$ . A tal fine introduciamo i nuovi scalari intermedi  $\tilde{\rho}$  e  $\tilde{\phi}$ , grazie ai quali arriviamo all'algoritmo nella sua versione (2).

---

**Algorithm 2** Algoritmo LSQR per il problema dei minimi quadrati regolarizzato

---

```
1: Inizializzazione
2:  $\beta_1 u_1 = b$ 
3:  $\alpha_1 v_1 = A^T u_1$ 
4:  $w_1 = v_1$ 
5:  $x_0 = 0$ 
6:  $\phi_1 = \beta_1$ 
7:  $\bar{\rho}_1 = \alpha_1$ 

8: for  $i = 1, 2, 3, \dots$  do ▷ Ripeti gli step 3–6
    9:     Continua la bidiagonalizzazione
    10:     $\beta_{i+1} u_{i+1} = Av_i - \alpha_i u_i$ 
    11:     $\alpha_{i+1} v_{i+1} = A^T u_{i+1} - \beta_{i+1} v_i$ 

    12:    Step di eliminazione di  $\lambda$ 
    13:     $\tilde{\rho} = \sqrt{\bar{\rho}^2 + \lambda^2}$ 
    14:     $\tilde{\phi} = \frac{\bar{\rho}}{\rho} \bar{\phi}$ 

    15:    Cotruisci e applica la prossima trasformazione ortogonale
    16:     $\rho_i = (\tilde{\rho}_i^2 + \beta_{i+1}^2)^{1/2}$ 
    17:     $c_i = \tilde{\rho}_i / \rho_i$ 
    18:     $s_i = \beta_{i+1} / \rho_i$ 
    19:     $\theta_{i+1} = s_i \alpha_{i+1}$ 
    20:     $\bar{\rho}_{i+1} = -c_i \alpha_{i+1}$ 
    21:     $\underline{\phi}_i = c_i \tilde{\phi}_i$ 
    22:     $\bar{\phi}_{i+1} = s_i \tilde{\phi}_i$ 

    23:    Aggiorna  $x, w$ .
    24:     $x_i = x_{i-1} + (\phi_i / \rho_i) w_i$ 
    25:     $w_{i+1} = v_{i+1} - (\theta_{i+1} / \rho_i) w_i$ 
```

---

### 3 Implementazione

In questa sezione presentiamo un'applicazione del metodo LSQR nel contesto del recupero di un'immagine sfocata (*deblurring*). In particolare, una fotografia del telescopio spaziale Hubble viene sfocata mediante una convoluzione 2D con un kernel generato casualmente. La risoluzione del problema ai minimi quadrati associato a tale operatore, avendo come termine noto l'immagine sfocata, permette di ottenere un'approssimazione quanto più vicina possibile all'immagine originale.

Il risultato fornito dall'algoritmo (1) risulta simile a quello calcolato dalla funzione predefinita di MATLAB. La figura 1 riporta, nell'ordine, l'immagine

originale, l’immagine sfocata e la versione restaurata.

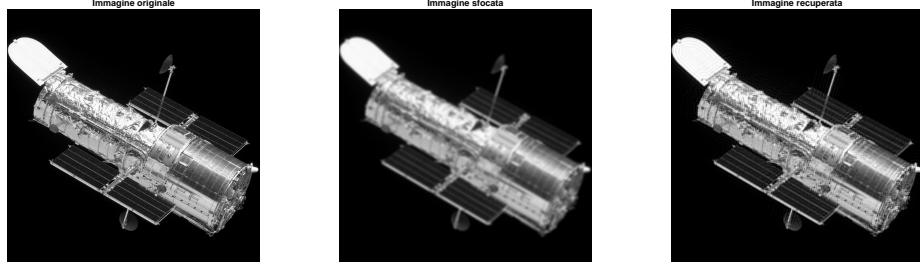


Figura 1: Recupero dell’immagine tramite algoritmo LSQR: originale, sfocata e restaurata.

Passiamo ora a utilizzare l’algoritmo (2) per risolvere il problema in forma regolarizzata, funzionalità non prevista dal comando predefinito di MATLAB. La regolarizzazione è utile per prevenire l’*overfitting*, ovvero l’adattamento eccessivo al rumore presente nei dati. Consideriamo quindi due scenari: applicazione diretta all’immagine sfocata e applicazione dopo l’aggiunta di rumore gaussiano.

Per valutare l’efficacia al variare del parametro di regolarizzazione  $\lambda$ , abbiamo effettuato una *grid search* su scala logaritmica, misurando la differenza in norma di Frobenius con l’immagine originale. Nel caso senza rumore aggiuntivo, il risultato ottimale si ottiene per  $\lambda = 0$ , come mostrato dall’andamento dell’errore in figura 2. Ciò indica che, in assenza di disturbi significativi, l’introduzione di regolarizzazione non porta benefici apprezzabili.

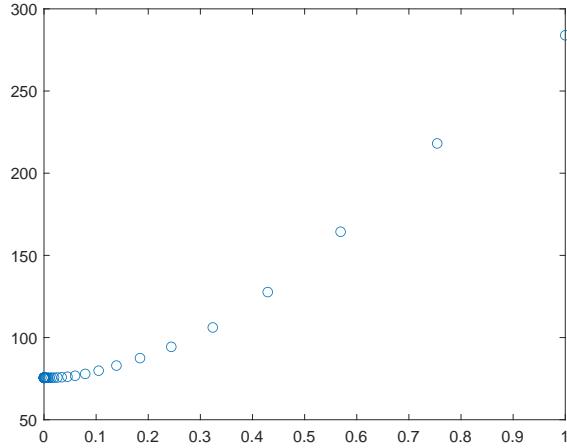


Figura 2: Errore in norma di Frobenius al variare di  $\lambda$  senza rumore aggiuntivo.

La situazione cambia se aggiungiamo rumore all’immagine già sfocata. La figura 3 mostra l’andamento dell’errore in norma di Frobenius per 50 valori di  $\lambda$ , scelti in modo logaritmico uniforme nell’intervallo  $[10^{-6}, 10^2]$ . Il rumore è

stato generato come matrice di valori gaussiani a media nulla e deviazione standard pari alla metà della deviazione standard empirica dei pixel dell'immagine. In questo caso, il valore ottimale di  $\lambda$  non è più nullo ma si attesta intorno a 0.35. Si potrebbe ulteriormente affinare la ricerca nei dintorni di tale valore per ottenere un risultato ancora migliore.

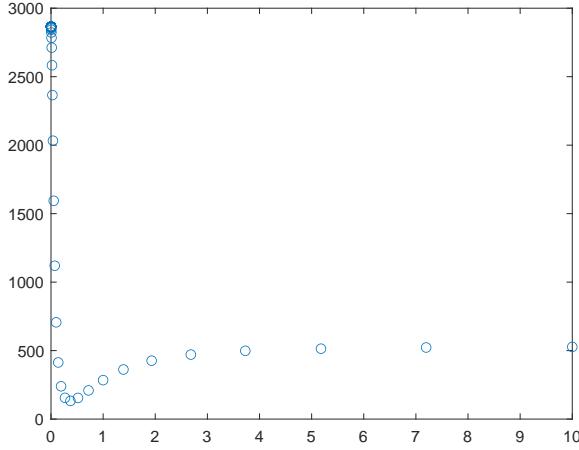


Figura 3: Errore in norma di Frobenius al variare di  $\lambda$  con rumore gaussiano aggiunto.

## 4 Conclusione

In questo report abbiamo analizzato l'algoritmo LSQR, e derivato la formulazione sia per il problema standard dei minimi quadrati sia per la sua controparte regolarizzata. Abbiamo condotto un'implementazione pratica su un problema di deblurring di immagini, la quale ha confermato l'efficacia del metodo. Inoltre è emerso un punto cruciale: l'utilità della regolarizzazione è strettamente legata alla presenza di rumore nei dati. In condizioni ideali, LSQR senza regolarizzazione riesce a restituire risultati eccellenti. Tuttavia, quando i dati sono soggetti a rumore, un parametro  $\lambda$  scelto opportunamente è fondamentale per prevenire l'overfitting e ottenere una soluzione accurata. Questo lavoro dimostra come LSQR non sia solo un potente metodo numerico, ma anche uno strumento flessibile che, nella sua versione regolarizzata, può essere adattato a risolvere problemi mal condizionati tipici delle applicazioni scientifiche e ingegneristiche.

## Riferimenti bibliografici

- [1] Paige, Saunders. *LSQR: An Algorithm for Sparse Linear Equations and Sparse Least Squares.*