A.D. 1308
unipg
DIPARTIMENTO
DI INGEGNERIA

Progetto di

Signal Processing and Optimization for Big Data

Corso di Laurea in Ingegneria Informatica e Robotica

Curriculum Data Science

Dipartimento di Ingegneria

docente

Prof. Paolo Banelli

Confronto tra Algoritmi di Regressione Lasso Sviluppati da Zero

363433 **Gian Marco Ferri** gianmarco.ferri@studenti.unipg.it

Indice

1 Introduzione **2**
1.1 Contesto Generale 2

2 Analisi Teorica **3**
2.1 LASSO 3
2.2 Soft-Thresholding 3
2.3 ADMM 4
2.4 ADMM Distribuito 5

3 Implementazione **7**
3.1 Dataset 7

4 Comparazioni **8**

1 Introduzione

Questo progetto affronta la risoluzione del problema della regressione con regolarizzazione Lasso tramite l'implementazione e il confronto di diversi algoritmi. In particolare, sono stati sviluppati in Matlab tre approcci distinti: ISTA, ADMM e una versione simulata di ADMM distribuito su più agenti.

Nella parte finale della documentazione vengono presentati dei confronti tra i tre algoritmi in termini di tempi di calcolo, numero di iterazioni e grafici che illustrano l'andamento della convergenza durante le iterazioni.

1.1 Contesto Generale

La regressione LASSO (Least Absolute Shrinkage and Selection Operator) rappresenta una metodologia fondamentale nell'ambito della modellazione statistica e del machine learning. Essa consente la gestione dell'overfitting e la selezione delle variabili, fornendo un approccio efficace per ottenere modelli robusti.

Con l'avvento di algoritmi sempre più complessi, è emersa la sfida dell'overfitting, che si verifica quando un modello si adatta eccessivamente ai dati di addestramento, catturando rumore anziché relazioni reali. La regressione LASSO si presenta come una soluzione a questo problema, introducendo una penalità sulla somma degli apporti delle variabili nel modello.

La sua peculiarità è dunque quella di applicare una regolarizzazione di tipo L1, promuovendo la sparsità del modello. Ciò significa che la LASSO favorisce la presenza di coefficienti nulli, consentendo di selezionare in modo automatico le variabili più rilevanti per la predizione.

2 Analisi Teorica

In questo capitolo viene svolta un'analisi approfondita del problema dal punto di vista teorico e matematico, allo scopo di comprendere gli algoritmi che saranno successivamente implementati.

Si fornisce una panoramica dettagliata della regressione LASSO, introducendo il concetto di regolarizzazione L1 e esaminando le soluzioni teoriche per il problema di ottimizzazione associato.

Inoltre, vengono esplorate le varianti dell'algoritmo, fornendo le basi concettuali e matematiche necessarie per una comprensione completa delle implementazioni pratiche che saranno esaminate nei capitoli successivi.

2.1 LASSO

Il LASSO (Least Absolute Shrinkage and Selection Operator) è una tecnica di regressione che introduce una regolarizzazione L1 al problema di ottimizzazione. Ciò consiste nell'aggiunta di un termine di penalità proporzionale alla somma dei valori assoluti dei coefficienti del modello, favorendo la presenza di coefficienti nulli, semplificando il modello e prevenendo l'overfitting.

Il problema di ottimizzazione associato alla regressione LASSO coinvolge la minimizzazione di una funzione di costo, che comprende sia il termine di regressione che la penalità L1.

$$\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y} - A\mathbf{x}\|_2^2 + \alpha\|\mathbf{x}\|_1$$

L'equazione sopra rappresenta il problema di ottimizzazione associato alla regressione LASSO. Il termine $\mathbf{x}$ rappresenta il vettore dei coefficienti del modello, $\mathbf{y}$ è il vettore delle risposte osservate, A è la matrice delle features e α è il parametro di regolarizzazione.

2.2 Soft-Thresholding

Nell'algoritmo ISTA (Iterative Shrinkage Thresholding Algorithm) la regressione LASSO viene risolta attraverso iterazioni di gradient descent fino a convergenza.

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \gamma\nabla\left[\frac{1}{2}\|\mathbf{y} - A\mathbf{x}^{(k)}\|_2^2 + \alpha\|\mathbf{x}^{(k)}\|_1\right]$$

Poiché la norma L_1 non è differenziabile, viene fatto ricorso al concetto di sub-gradiente attraverso l'uso dell'operatore di *soft-thresholding*.

$$[\text{sub}\nabla f(x_0)]_i = \begin{cases} \frac{\partial f(x_0)}{\partial x_i}, & \text{se differenziabile in } x_0 \\ \left[\frac{\partial f(x_0^-)}{\partial x_i}, \frac{\partial f(x_0^+)}{\partial x_i}\right], & \text{se non differenziabile in } x_0 \end{cases}$$

Che applicato al problema in questione risulta:

$$\text{sub}\nabla\Phi = \begin{cases} \gamma\alpha + (u_j - x_j) & \text{se } u_j > 0 \\ -\gamma\alpha + (u_j - x_j) & \text{se } u_j < 0 \\ \left[-\gamma\alpha + (u_j - x_j), \gamma\alpha + (u_j - x_j)\right] & \text{se } u_j = 0 \end{cases}$$

$$\Rightarrow \begin{cases} u_j = x_j - \gamma\alpha & \text{se } x_j > \gamma\alpha \\ u_j = x_j + \gamma\alpha & \text{se } x_j < -\gamma\alpha \\ u_j = 0 & \text{se } -\gamma\alpha < x_j < \gamma\alpha \end{cases}$$

Questo prende il nome di *Soft-Thresholding* e permette di calcolare il gradiente della funzione costo in esame. La convergenza viene raggiunta quando la variazione è inferiore ad una quantità che indica la tolleranza.

2.3 ADMM

L'Alternating Direction Method of Multipliers (ADMM) è un approccio di ottimizzazione che si basa sulla decomposizione del problema in sottoproblemi più gestibili. Appartiene alla famiglia degli algoritmi di ottimizzazione primal-dual.

Per applicarlo al Lasso è necessario riformulare il problema, andando ad introdurre la *slack variable* $\mathbf{z}$ per separare l'ottimizzazione della funzione originale da quella della regolarizzazione.

La formulazione generale è dunque:

$$\min_{\mathbf{x},\mathbf{z}} \frac{1}{2}\|A\mathbf{x} - \mathbf{y}\|_2^2 + \alpha\|\mathbf{z}\|_1$$

$$\text{s.t.} \quad \mathbf{x} - \mathbf{z} = 0$$

ADMM suddivide il problema in 3 sotto-problemi, le cui soluzioni verranno ricercate in maniera alternata fino a convergenza.

Viene mostrata la formulazione standard scalata:

$$\mathbf{x}^{(k+1)} = \underset{\mathbf{x}}{\text{argmin}} \left\{ \frac{1}{2}\|A\mathbf{x} - \mathbf{y}\|_2^2 + \frac{\rho}{2}\|\mathbf{x} - \mathbf{z}^{(k)} + \mathbf{u}^{(k)}\|_2^2 \right\}$$

$$\mathbf{z}^{(k+1)} = \underset{\mathbf{z}}{\text{argmin}} \left\{ \alpha\|\mathbf{z}\|_1 + \frac{\rho}{2}\|\mathbf{x}^{(k+1)} - \mathbf{z} + \mathbf{u}^{(k)}\|_2^2 \right\}$$

$$\mathbf{u}^{(k+1)} = \mathbf{u}^{(0)} + \sum_{i=1}^{k+1} \|\mathbf{x}^{(i)} - \mathbf{z}^{(i)}\|_2^2$$

Il primo step si può risolvere in forma chiusa eguagliando la derivata a 0.

$$\nabla_{\mathbf{x}} \left(\frac{1}{2} \|A\mathbf{x} - \mathbf{y}\|_2^2 + \frac{\rho}{2} \|\mathbf{x} - \mathbf{z}^{(k)} + \mathbf{u}^{(k)}\|_2^2 \right) = 0$$

$$A^T(A\mathbf{x} - \mathbf{y}) + \rho(\mathbf{x} - \mathbf{z}^{(k)} + \mathbf{u}^{(k)}) = 0$$

$$(A^T A + \rho I)\mathbf{x} = A^T\mathbf{y} + \rho(\mathbf{z}^{(k)} - \mathbf{u}^{(k)})$$

$$\rightarrow \mathbf{x}^{(k+1)} = (A^T A + \rho I)^{-1}(A^T\mathbf{y} + \rho(\mathbf{z}^{(k)} - \mathbf{u}^{(k)}))$$

Per il secondo step invece, non esiste il gradiente quindi, come nell'algoritmo precedente, si utilizza il sub-gradiente che in questo caso corrisponde al soft thresholding operator.

Concludendo, Lasso con ADMM scalato assume la seguente formulazione:

$$\mathbf{x}^{(k+1)} = (A^T A + \rho I)^{-1}(A^T\mathbf{y} + \rho(\mathbf{z}^{(k)} - \mathbf{u}^{(k)}))$$

$$\mathbf{z}^{(k+1)} = S_{\frac{\alpha}{\rho}}(\mathbf{x}^{(k+1)} + \mathbf{u}^{(k)})$$

$$\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \mathbf{x}^{(k+1)} - \mathbf{z}^{(k+1)}$$

I parametri ρ e α rivestono un ruolo critico e la loro scelta richiede attente valutazioni per bilanciare la velocità di convergenza e la robustezza.

L'utilizzo dell'ADMM porta svariati vantaggi, come la suddivisione in sotto-problemi di più facile gestione, buona stabilità e convergenza a valori ottimali in tempi generalmente più brevi ed inoltre apre la strada a possibili implementazioni distribuite.

2.4 ADMM Distribuito

La versione distribuita di ADMM estende l'ADMM classico per affrontare problemi di dimensioni più grandi distribuendo il lavoro tra diversi agenti. L'ottimizzazione al consenso è implementata suddividendo gli esempi tra gli agenti, ognuno con copie delle stesse variabili di ottimizzazione. Il problema è quindi diviso tra i dati e risolto in modo collaborativo.

Questo tipo di approccio è possibile quando la funzione obiettivo è additiva e si può quindi scomporre l'ottimizzazione.

La formulazione standard di un'ottimizzazione al consenso è:

$$\min_{(x_1,\ldots,x_N),\mathbf{z}} \sum_{i=1}^{N} f_i(\mathbf{x}) + g(\mathbf{z})$$

$$\text{s.t.} \quad \mathbf{x}_i - \mathbf{z} = \mathbf{0}, \quad i = 1, \ldots, N$$

Il problema Lasso originale appartiene alla classe di problemi distribuibili in quanto il vettore delle osservazioni $\mathbf{y}$ è separabile in N osservazioni e, di conseguenza, anche il prodotto scalare $A\mathbf{x}$.

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_N \end{bmatrix}, \quad A\mathbf{x} = \begin{bmatrix} A_1 \\ \vdots \\ A_N \end{bmatrix} \mathbf{x} = \begin{bmatrix} A_1\mathbf{x} \\ \vdots \\ A_N\mathbf{x} \end{bmatrix}$$

Andando a formulare al consenso il problema trattato si ottiene:

$$\min_{(x_1,\dots,x_N),\mathbf{z}} \sum_{i=1}^{N} \|\mathbf{y}_i - A_i\mathbf{x}_i\|_2^2 + \lambda\|\mathbf{z}\|_1$$
$$\text{s.t.} \quad \mathbf{x}_i - \mathbf{z} = \mathbf{0}, \quad i = 1,\dots,N$$

Il problema viene risolto mediante la versione scalata dell'ADMM:

$$\mathbf{x}_i^{(k+1)} = \underset{\mathbf{x}_i}{\operatorname{argmin}} \left\{ \|\mathbf{y}_i - A_i\mathbf{x}_i\|_2^2 + \frac{\rho}{2}\|\mathbf{x}_i - \mathbf{z}^{(k)} + \mathbf{u}_i^{(k)}\|_2^2 \right\}$$

$$\mathbf{z}^{(k+1)} = \underset{\mathbf{z}}{\operatorname{argmin}} \left\{ g(\mathbf{z}) + \frac{\rho}{2} \sum_{i=1}^{N} \|\mathbf{x}_i^{(k+1)} - \mathbf{z} + \mathbf{u}_i^{(k)}\|_2^2 \right\} \quad i = 1,\dots,N$$
$$\mathbf{u}_i^{(k+1)} = \mathbf{u}_i^{(k)} + (\mathbf{x}_i^{(k+1)} - \mathbf{z}^{(k+1)}) \quad i = 1,\dots,N$$

Nel primo step si hanno N problemi distribuibili su vari agenti con una soluzione in forma chiusa, come nel caso centralizzato.

Nel secondo step servono tutte le variabili primali e duali per calcolare il valore globale $\mathbf{z}$, quindi verrà eseguito in un fusion center.

L'aggiornamento del terzo step viene invece eseguito localmente in ogni agente.

L'algoritmo può quindi essere scritto così:

$$\mathbf{x}_i^{(k+1)} = (A_i^T A_i + \frac{\rho}{2} I)^{-1} (A_i^T \mathbf{y_i} + \frac{\rho}{2}(\mathbf{z}_i^{(k)} - \mathbf{u}_i^{(k)}))$$

$$\mathbf{z}^{(k+1)} = \operatorname{Prox}_{\frac{\lambda}{N\rho}\|\cdot\|_1} \left\{ \hat{\mathbf{x}}^{(k+1)} + \hat{\mathbf{u}}^{(k)} \right\} = S_{\frac{\lambda}{N\rho}} (\hat{\mathbf{x}}^{(k+1)} + \hat{\mathbf{u}}^{(k)})$$

$$\mathbf{u}_i^{(k+1)} = \mathbf{u}_i^{(k)} + \mathbf{x}_i^{(k+1)} - \mathbf{z}^{(k+1)} \quad i = 1,\dots,N$$

dove $\hat{\mathbf{x}}^{(k+1)}$ e $\hat{\mathbf{u}}^{(k)}$ sono le medie delle variabili primarie e duali al passo $k+1$ e k rispettivamente.

Questa tecnica si presta bene a dataset di grandi dimensioni e inoltre consente di gestire situazioni in cui i dati sono distribuiti.

3 Implementazione

Gli algoritmi descritti di seguito sono stati implementati all'interno della classe `LassoReg`. In base ai parametri passati all'oggetto istanziato dalla classe, è possibile scegliere l'algoritmo con cui verrà eseguita la fase di training, lo step-size, la tolleranza per la convergenza, il numero massimo di iterazioni e la penalità da applicare.

3.1 Dataset

Il dataset utilizzato è il California Housing Dataset, un insieme di dati pubblici che raccoglie informazioni relative alle abitazioni in California. Ogni riga rappresenta un blocco residenziale e le colonne includono le seguenti variabili:

- **longitude**: longitudine del blocco;
- **latitude**: latitudine del blocco;
- **housing_median_age**: età mediana delle abitazioni nel blocco;
- **total_rooms**: numero totale di stanze nel blocco;
- **total_bedrooms**: numero totale di camere da letto nel blocco;
- **population**: popolazione nel blocco;
- **households**: numero di nuclei familiari nel blocco;
- **median_income**: reddito mediano delle famiglie nel blocco;
- **median_house_value**: valore mediano delle abitazioni nel blocco;

Durante la fase di preprocessing, i valori mancanti nella variabile *total_bedrooms* sono stati sostituiti con la mediana. Tutte le feature numeriche sono state normalizzate nell'intervallo $[0, 1]$. La variabile categorica *ocean_proximity* è stata esclusa dall'analisi.

4 Comparazioni

I parametri con cui sono stati eseguiti i tre algoritmi sono i seguenti:

- iterazioni massime = 50000;
- step-size = 0.01;
- l1-penalty = 1;
- tolerance = 1e-4;
- agenti = 8 (ADMM distribuito);

Il confronto delle prestazioni ottenute è mostrato nella Tabella 1.

Tabella 1: Comparazione algoritmi

	R2	time (s)	iterazioni
ISTA	0.5588	12.876	50000
ADMM	0.5839	0.0010	4
ADMM-Dist	0.5726	0.0376	170

È importante sottolineare che i valori riportati sono stati ottenuti effettuando una suddivisione casuale del dataset in training set e test set: esecuzioni diverse potrebbero generare risultati leggermente differenti.

A seguire vengono illustrate le variazioni dei criteri di convergenza durante le iterazioni degli algoritmi, insieme ai grafici delle predizioni effettuate rispetto ai valori reali.

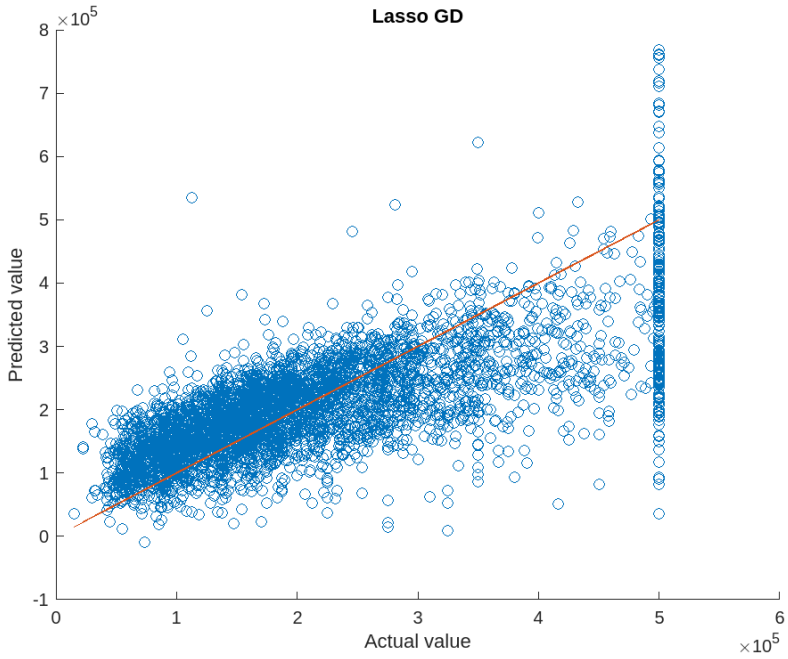

Figura 1: Predizioni ottenute con Soft-Thresholding.

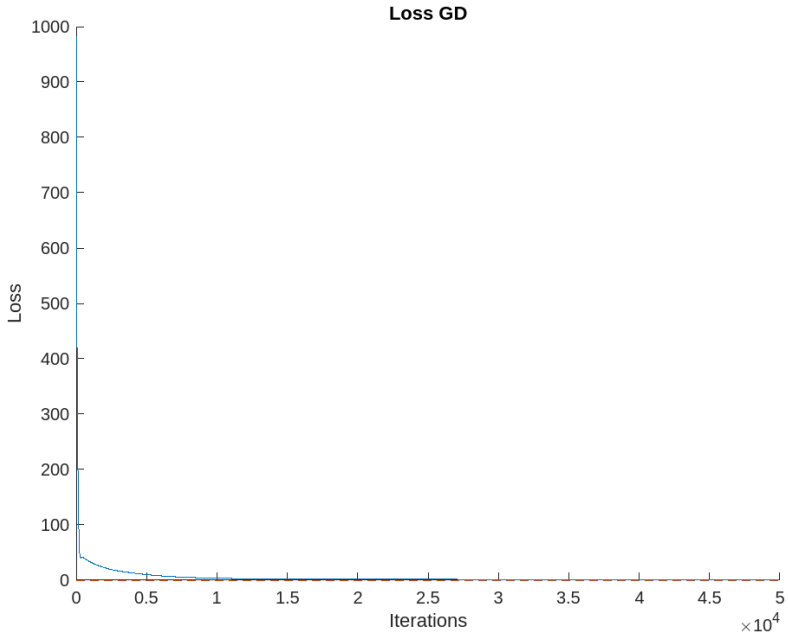

Figura 2: Convergenza del Soft-Thresholding.

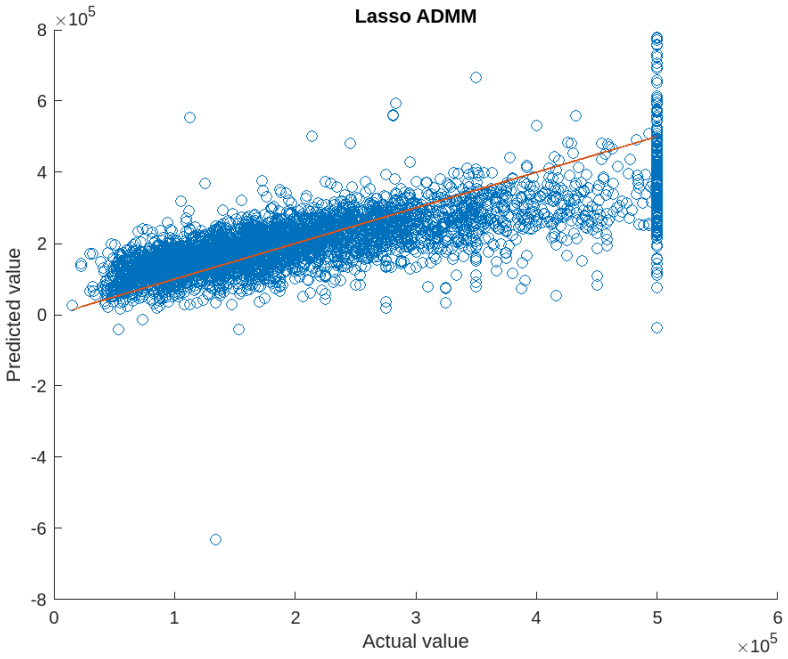

Figura 3: Predizioni ottenute con ADMM.

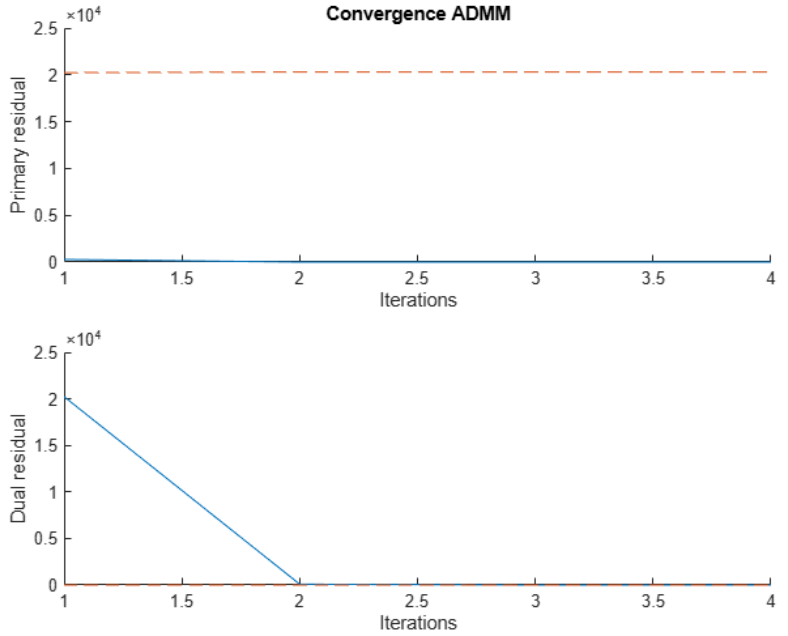

Figura 4: Convergenza di ADMM.

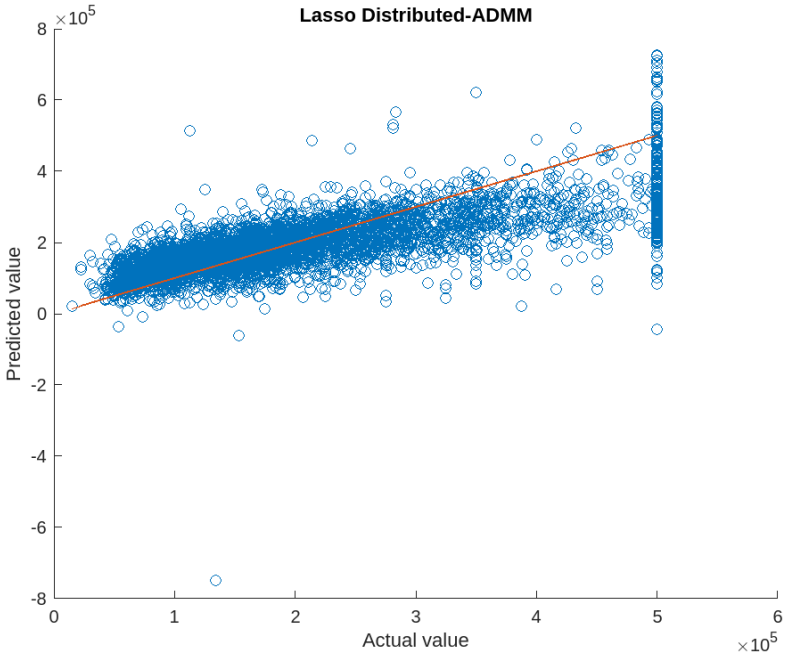

Figura 5: Predizioni ottenute con ADMM distribuito.

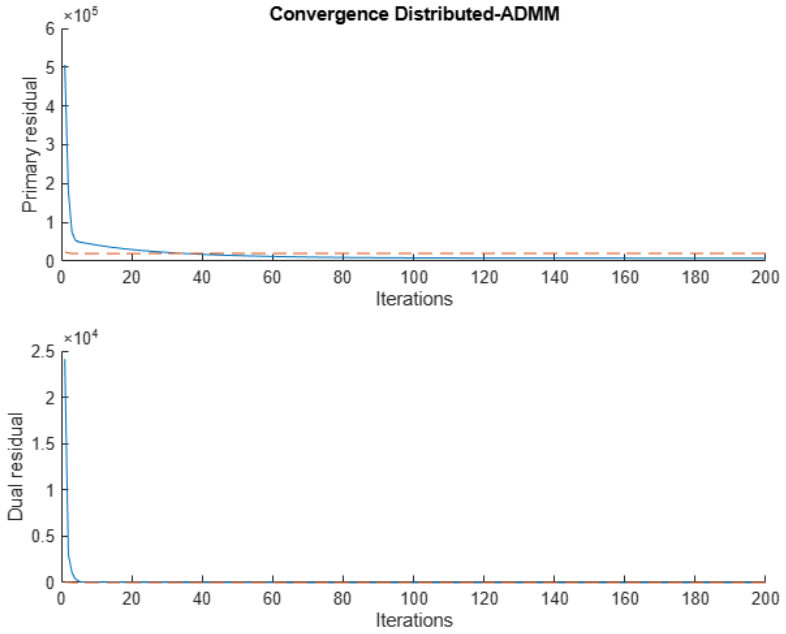

Figura 6: Convergenza di ADMM distribuito.