Università degli Studi di Padova
Dipartimento di Scienze Statistiche
Corso di Laurea Magistrale in
Scienze Statistiche

Stima dei parametri del modello di regressione multivariato penalizzato mediante ADMM

Relatore Prof. Mauro Bernardi
Dipartimento di Scienze Statistiche
Università degli Studi di Padova

Laureando Pietro Mietto
Matricola 2028828

Anno Accademico 2023/2024

Indice

Introduzione

Con l'emergere dei *big data* e la crescente necessità di gestire volumi di dati sempre più vasti, sono emerse due esigenze principali nel campo dell'analisi statistica e del *machine learning*. La prima riguarda la selezione delle variabili, soprattutto nei casi in cui il numero di variabili esplicative (denotate come p) supera quello delle osservazioni (denotate come n). In queste circostanze, diventa essenziale individuare un sottoinsieme di variabili esplicative che possano spiegare in modo adeguato la variabile risposta, evitando così problemi di sovra-adattamento e migliorando la generalizzazione del modello.
In secondo luogo, specialmente quando il numero di variabili è molto maggiore rispetto alle osservazioni, diventa necessario trovare metodologie che riducano il carico computazionale. Infatti visto l'enorme carico di dati, il costo computazionale può essere proibitivo se non vengono adottate strategie di riduzione della dimensionalità o di ottimizzazione degli algoritmi.
In sintesi, la selezione delle variabili e la riduzione della dimensionalità sono due sfide cruciali quando si affrontano grandi volumi di dati. Affrontare efficacemente queste sfide può portare a modelli più efficienti, interpretabili e generalizzabili, consentendo così di trarre il massimo valore informativo dai dati disponibili.
Proprio per questo in questo lavoro sarà trattato il modello di regressione lineare multivariato, quindi in quei casi in cui la variabile rispsota è una matrice $\boldsymbol{Y}$ formata da n righe e q colonne. Si è deciso di trattare questo tipo di modello prima di tutto per la sua versatilità, infatti molti altri tipi di modelli si possono riscrivere in questo formato. Un esempio di questo tipo è la scrittura in forma compatta o matriciale del modello autoregressivo vettoriale di ordine p, VAR(p), Pesaran and Shin (1998). L'obiettivo, quindi, sarà quello di trovare un metodo efficacie per effettuare una selezione dei parametri, in modo simultaneo, cercando di stimare

al meglio la matrice dei parametri $\boldsymbol{B} \in \mathbb{R}^{p \times q}$. Si introdurrà, quindi, il concetto di modello multivariato penalizzato e con doppia penalizzazione, rifacendosi al lavoro svolto da Rothman et al. (2010). Anche in questo lavoro il tipo di penalizzazione sarà di tipo *LASSO*.
Per la stima dei parametri del modello lineare multivariato penalizzato, e quindi della soluzione di un problema di ottimizzazione vincolato, si è deciso di concentrarsi sull'utilizzo del "Metodo Alternato dei Moltiplicatori" (ADMM), introdotto ds Boyd et al. (2011b). Questo metodo come si vedrà in questo lavoro è molto versatile e permette di trovare delle soluzioni in forma chiusa, impostando il modello secondo una determinata specificazione. Inoltre, si cercherà di applicare questo algoritmo in un contesto con variabile risposta multivariata.
Nel primo capitolo, quindi, sarà presentata la formalizzazione del modello multivariato penalizzato e con doppia penalizzazione. Inoltre, ampio spazio sarà dedicato alla soluzione del problema utilizzando il metodo ADMM.
Nel secondo capitolo, invece, sarà presentato lo studio di simulazione. Per questo studio si è seguito quanto scritto da Rothman et al. (2010) cercando poi di confrontare il modello stimato con il metodo dell'ADMM con altri tipi di modelli o procedure che effettuano una selezione dei parametri e delle variabili.
Infine, nel terzo capitolo è riportata l'applicazione del modello multivariato penalizzato a un *dataset* che riguarda i log-rendimenti di nove *stocks* americani del 2004. Per valutare la stima dei parametri calcolati secondo il metodo ADMM si sono stimati altri modelli e si sono confrontati per la loro capacità predittiva.
Una parte importante del lavoro è stata dedicata anche alla creazione di un pacchetto `R`, poi nominato `MLRS`, che compie la stima dei parametri di un modello lineare multivariato penalizzato utilizzando l'ADMM. Si è deciso di implementare gli algoritmi nell'ambiente C++, i tre algoritmi principali sono stati riportati nella sezione dell'Appendice.

Capitolo 1

Introduzione ADMM

In questo capitolo si presenta la parte di teoria di questo lavoro. La prima parte si descrive il metodo di stima tramite *Alternating Direction Method of Multipliers* (ADMM), quando può essere applicato e i criteri di stop di questo algoritmo. In seguito si formalizza il modello multivariato penalizzato e si mostra come può essere applicato l'algortmo ADMM per la stima dei parametri in questi problemi. Infine, si introduce il modello multivariato con doppia penalizzazione, sui parametri della matrice B e sulla parte di varianza, e come l'algoritmo ADMM può essere utilizzato in questi casi.

1.1 Formulazione del problema di ottimizzazione

Si considera il seguente problema di minimizzazione convesso:

$$\begin{array}{ll} \text{min.} & \ell(\boldsymbol{\beta}) + g(\boldsymbol{\vartheta}) \\ \text{sotto v. :} & \boldsymbol{C}\boldsymbol{\beta} + \boldsymbol{B}\boldsymbol{\vartheta} = \boldsymbol{c} \\ & \boldsymbol{\beta} \in \mathbb{R}^p, \boldsymbol{\vartheta} \in \mathbb{R}^q, \end{array} \tag{1}$$

dove $\ell : \mathbb{R}^p \text{o} \mathbb{R} \cup \{\infty\}$ e $g : \mathbb{R}^q \text{o} \mathbb{R} \cup \{\infty\}$ sono funzioni convesse, $\boldsymbol{C} \in \mathbb{R}^{m \times p}$, $\boldsymbol{B} \in \mathbb{R}^{m \times q}$ and $\boldsymbol{c} \in \mathbb{R}^m$. Per risolvere il problema (1), Gabay and Mercier (1976) e Glowinski and Marroco (1975) hanno proposto *Alternating Direction Method of Multipliers* (ADMM), il quale genera una sequenza iterativa attraverso la seguente

ricorsione

$$\widehat{\boldsymbol{\beta}}^{(k+1)} = \arg\min_{\boldsymbol{\beta}} \mathcal{L}_\rho(\boldsymbol{\beta}, \widehat{\boldsymbol{\vartheta}}^{(k)}, \widehat{\boldsymbol{\lambda}}^{(k)}) \tag{2a}$$

$$\widehat{\boldsymbol{\vartheta}}^{(k+1)} = \arg\min_{\boldsymbol{\vartheta}} \mathcal{L}_\rho(\widehat{\boldsymbol{\beta}}^{(k+1)}, \boldsymbol{\vartheta}, \widehat{\boldsymbol{\lambda}}^{(k)}) \tag{2b}$$

$$\widehat{\boldsymbol{\lambda}}^{(k+1)} = \widehat{\boldsymbol{\lambda}}^{(k)} + \rho(\boldsymbol{C}\widehat{\boldsymbol{\beta}}^{(k+1)} + \boldsymbol{B}\widehat{\boldsymbol{\vartheta}}^{(k+1)} - \boldsymbol{c}), \tag{2c}$$

dove $\mathcal{L}_\rho : \mathbb{R}^p \times \mathbb{R}^q \times \mathbb{R}^m \text{o}\mathbb{R}$ è il *lagrangiano aumentato* per (1) definito come

$$\mathcal{L}_\rho(\boldsymbol{\beta}, \boldsymbol{\vartheta}, \boldsymbol{\lambda}) = \ell(\boldsymbol{\beta}) + g(\boldsymbol{\vartheta}) + \boldsymbol{\lambda}^\intercal(\boldsymbol{C}\boldsymbol{\beta} + \boldsymbol{B}\boldsymbol{\vartheta} - \boldsymbol{c}) + \frac{\rho}{2}\|\boldsymbol{C}\boldsymbol{\beta} + \boldsymbol{B}\boldsymbol{\vartheta} - \boldsymbol{c}\|_2^2. \tag{3}$$

Il parametro $\boldsymbol{\lambda} \in \mathbb{R}^m$ è il moltiplicatore associato al vincolo di uguaglianza e $\rho > 0$ è il parametro di penalità. La convergenza globale dell'ADMM (2a)-(2c) può essere stabilita secondo condizioni lievi non forti (guardare Boyd et al., 2011a). L'ADMM può essere scritto in una forma lievamente differente, ma più conveniente, combinando il termine quadratico con il termine lineare nel *langrangiano aumentato* e scomponendo la variabile duale $\boldsymbol{\lambda}$. Si definisce il residuo $\boldsymbol{r} = \boldsymbol{C}\boldsymbol{\beta} + \boldsymbol{B}\boldsymbol{\vartheta} - \boldsymbol{c}$, si ottiene

$$\boldsymbol{\lambda}^\intercal\boldsymbol{r} + \frac{\rho}{2}\|\boldsymbol{r}\|_2^2 = \frac{\rho}{2}\|\boldsymbol{r} + \boldsymbol{v}\|_2^2 - \frac{\rho}{2}\|\boldsymbol{v}\|_2^2,$$

dove $\boldsymbol{v} = \frac{1}{\rho}\boldsymbol{\lambda}$ è la variabile duale scalata. Usando la variabile duale scalata il *lagrangiano aumentato* scalato diventa

$$\mathcal{L}_\rho^\star(\boldsymbol{\beta}, \boldsymbol{\vartheta}, \boldsymbol{v}) = \ell(\boldsymbol{\beta}) + g(\boldsymbol{\vartheta}) + \frac{\rho}{2}\|\boldsymbol{r} + \boldsymbol{v}\|_2^2 - \frac{\rho}{2}\|\boldsymbol{v}\|_2^2, \qquad \boldsymbol{r} = \boldsymbol{C}\boldsymbol{\beta} + \boldsymbol{B}\boldsymbol{\vartheta} - \boldsymbol{c}. \tag{4}$$

1.1.1 Condizioni ottimali

Prima di passare ai passi di stima si rivedono le condizioni di arresto descritte in Boyd et al. (2011a) per un generico problema di ottimizzazione vincolato definito in equazione (1) che ha il *lagrangiano aumentato* definito in equazione (3). La soluzione del problema di ottimizzazione in equazione (1) è ottenuta iterativamente dalle regole di aggiornamento in equazione (2). Le condizioni necessarie e sufficienti per cui $(\boldsymbol{\beta}^*, \boldsymbol{\vartheta}^*, \boldsymbol{\lambda}^*)$ sono considerate come soluzione ottima per il problema di ottimizzazione in equazioni (1) riguardano la funzione primale e la funzione duale,

che per essere accattabili devono rispettare le seguenti uguaglianze

$$\boldsymbol{C}\boldsymbol{\beta}^* + \boldsymbol{B}\boldsymbol{\vartheta}^* - \boldsymbol{c} = 0 \tag{5}$$

$$0 \in \partial\ell(\boldsymbol{\beta}^*) + \boldsymbol{C}^\intercal\boldsymbol{\lambda}^* \tag{6}$$

$$0 \in \partial g(\boldsymbol{\vartheta}^*) + \boldsymbol{B}^\intercal\boldsymbol{\lambda}^*, \tag{7}$$

dove ∂ denota l'operatore subdifferenziale. Visto che $\widehat{\boldsymbol{\vartheta}}^{(k+1)}$ minimizza $\mathcal{L}_\rho(\widehat{\boldsymbol{\beta}}^{(k+1)}, \boldsymbol{\vartheta}, \widehat{\boldsymbol{\lambda}}^{(k)})$ in equazione (2b), per definizione si ottiene che

$$\begin{aligned} 0 &\in \partial g(\widehat{\boldsymbol{\vartheta}}^{(k+1)}) + \boldsymbol{B}^\intercal\widehat{\boldsymbol{\lambda}}^{(k)} + \rho\boldsymbol{B}^\intercal(\boldsymbol{C}\widehat{\boldsymbol{\beta}}^{(k+1)} + \boldsymbol{B}\widehat{\boldsymbol{\vartheta}}^{(k+1)} - \boldsymbol{c}) \\ &=\in \partial g(\widehat{\boldsymbol{\vartheta}}^{(k+1)}) + \boldsymbol{B}^\intercal\widehat{\boldsymbol{\lambda}}^{(k)} + \rho\boldsymbol{B}^\intercal\widehat{\mathbf{r}}^{(k+1)} \\ &= \partial g(\widehat{\boldsymbol{\vartheta}}^{(k+1)}) + \boldsymbol{B}^\intercal\widehat{\boldsymbol{\lambda}}^{(k+1)}, \end{aligned} \tag{8}$$

dove $\widehat{\mathbf{r}}^{(k+1)} = \boldsymbol{C}\widehat{\boldsymbol{\beta}}^{(k+1)} + \boldsymbol{B}\widehat{\boldsymbol{\vartheta}}^{(k+1)} - \boldsymbol{c}$ è il residuo primale al passo $(k+1)$ e $\widehat{\boldsymbol{\lambda}}^{(k+1)} = \widehat{\boldsymbol{\lambda}}^{(k)} + \rho\widehat{\mathbf{r}}^{(k+1)}$ è l'aggiornamento per $\boldsymbol{\lambda}$ in equazione (2c). L'equazione (8), essenzialmente, assicura che $\widehat{\boldsymbol{\vartheta}}^{(k+1)}$ e $\widehat{\boldsymbol{\lambda}}^{(k+1)}$ soddisfano sempre l'equazione (7) che assicura che la convergenza verso l'ottimalità dell'algoritmo ADMM può essere stimato dalle funzioni in equzioni (5)-(6). Inoltre, visto che $\widehat{\boldsymbol{\beta}}^{(k+1)}$ minimizza $\mathcal{L}_\rho(\boldsymbol{\beta}, \widehat{\boldsymbol{\vartheta}}^{(k)}, \widehat{\boldsymbol{\lambda}}^{(k)})$ in equazione (2a), per definizione si ottiene

$$\begin{aligned} 0 &\in \partial f(\widehat{\boldsymbol{\beta}}^{(k+1)}) + \boldsymbol{C}^\intercal\widehat{\boldsymbol{\lambda}}^{(k)} + \rho\boldsymbol{C}^\intercal(\boldsymbol{C}\widehat{\boldsymbol{\beta}}^{(k+1)} + \boldsymbol{B}\widehat{\boldsymbol{\vartheta}}^{(k)} - \boldsymbol{c}) \\ &= \partial f(\widehat{\boldsymbol{\beta}}^{(k+1)}) + \boldsymbol{C}^\intercal\widehat{\boldsymbol{\lambda}}^{(k)} + \rho\boldsymbol{C}^\intercal(\boldsymbol{C}\widehat{\boldsymbol{\beta}}^{(k+1)} + \boldsymbol{B}\widehat{\boldsymbol{\vartheta}}^{(k)} + \boldsymbol{B}\widehat{\boldsymbol{\vartheta}}^{(k+1)} - \boldsymbol{B}\widehat{\boldsymbol{\vartheta}}^{(k+1)} - \boldsymbol{c}) \\ &= \partial f(\widehat{\boldsymbol{\beta}}^{(k+1)}) + \boldsymbol{C}^\intercal\widehat{\boldsymbol{\lambda}}^{(k)} + \rho\boldsymbol{C}^\intercal(\boldsymbol{C}\widehat{\boldsymbol{\beta}}^{(k+1)} + \boldsymbol{B}\widehat{\boldsymbol{\vartheta}}^{(k+1)} - \boldsymbol{c}) + \rho\boldsymbol{C}^\intercal\boldsymbol{B}(\widehat{\boldsymbol{\vartheta}}^{(k)} - \widehat{\boldsymbol{\vartheta}}^{(k+1)}) \\ &= \partial f(\widehat{\boldsymbol{\beta}}^{(k+1)}) + \boldsymbol{C}^\intercal\widehat{\boldsymbol{\lambda}}^{(k)} + \rho\boldsymbol{C}^\intercal\widehat{\mathbf{r}}^{(k+1)} + \rho\boldsymbol{C}^\intercal\boldsymbol{B}(\widehat{\boldsymbol{\vartheta}}^{(k)} - \widehat{\boldsymbol{\vartheta}}^{(k+1)}) \\ &= \partial f(\widehat{\boldsymbol{\beta}}^{(k+1)}) + \boldsymbol{C}^\intercal(\widehat{\boldsymbol{\lambda}}^{(k)} + \rho\widehat{\mathbf{r}}^{(k+1)}) + \rho\boldsymbol{C}^\intercal\boldsymbol{B}(\widehat{\boldsymbol{\vartheta}}^{(k)} - \widehat{\boldsymbol{\vartheta}}^{(k+1)}) \\ &= \partial f(\widehat{\boldsymbol{\beta}}^{(k+1)}) + \boldsymbol{C}^\intercal\widehat{\boldsymbol{\lambda}}^{(k+1)} + \rho\boldsymbol{C}^\intercal\boldsymbol{B}(\widehat{\boldsymbol{\vartheta}}^{(k)} - \widehat{\boldsymbol{\vartheta}}^{(k+1)}), \end{aligned} \tag{9}$$

dove $\widehat{\mathbf{r}}^{(k+1)} = \boldsymbol{C}\widehat{\boldsymbol{\beta}}^{(k+1)} + \boldsymbol{B}\widehat{\boldsymbol{\vartheta}}^{(k+1)} - \boldsymbol{c}$ è il residuo primale al passo $(k+1)$ e $\widehat{\boldsymbol{\lambda}}^{(k+1)} = \widehat{\boldsymbol{\lambda}}^{(k)} + \rho\widehat{\mathbf{r}}^{(k+1)}$ è l'aggiornamento per $\boldsymbol{\lambda}$ in equazione (2c). L'equazione

(9) può essere scrittia in modo equvalente come segue

$$\widehat{\mathbf{s}}^{(k+1)} = \rho \boldsymbol{C}^{\intercal}\boldsymbol{B}(\widehat{\boldsymbol{\vartheta}}^{(k+1)} - \widehat{\boldsymbol{\vartheta}}^{(k)}) \in \partial f(\widehat{\boldsymbol{\beta}}^{(k+1)}) + \boldsymbol{C}^{\intercal}\widehat{\boldsymbol{\lambda}}^{(k+1)}, \tag{10}$$

dove $\widehat{\mathbf{s}}^{(k+1)} = \rho \boldsymbol{C}^{\intercal}\boldsymbol{B}(\widehat{\boldsymbol{\vartheta}}^{(k+1)} - \widehat{\boldsymbol{\vartheta}}^{(k)})$ denota il residuo per la condizione di verosimiglianza duale al passo $(k+1)$. Riassumendo, le condizioni di ottimo per il problema di ADMM sono tre, (5)-(7). L'ultima condizione (7) vale sempre per $(\widehat{\boldsymbol{\beta}}^{(k+1)}, \widehat{\boldsymbol{\vartheta}}^{(k+1)}, \widehat{\boldsymbol{\lambda}}^{(k+1)})$; i residui per le altre due, (5) e (6), sono rispettivamente i residui primali e duali $\widehat{\mathbf{r}}^{(k+1)}$ e $\widehat{\mathbf{s}}^{(k+1)}$. Per come procede l'ADMM, questi due residui convergono a zero.

1.1.2 Criteri di stop

Boyd et al. (2011a) suggeriscono che dei termini di stop ragionevoli riguardano i residui primali e duali. Questi devono essere "piccoli", per esempio,

$$\|\widehat{\mathbf{r}}^{(k+1)}\|_2 \leq \epsilon^{\text{pri}}, \qquad \|\widehat{\mathbf{s}}^{(k+1)}\|_2 \leq \epsilon^{\text{dual}},$$

dove $\epsilon^{\text{pri}} > 0$ e $\epsilon^{\text{dual}} > 0$ rientrano nelle condizioni di accettabilità per i residui primali e i residui duali in (5) e (6). Questi valori di tolleranza possono essere scelti utilizzando un criterio assoluto e relativo, come per esempio

$$\begin{aligned}
\epsilon^{\text{pri}} &= \sqrt{m}\epsilon^{\text{abs}} + \epsilon^{\text{rel}} \max\{\|\boldsymbol{C}\widehat{\boldsymbol{\beta}}^{(k)}\|_2, \|\boldsymbol{B}\widehat{\boldsymbol{\vartheta}}^{(k)}\|_2, \|\boldsymbol{c}\|_2\} \\
\epsilon^{\text{dual}} &= \sqrt{p}\epsilon^{\text{abs}} + \epsilon^{\text{rel}}\|\boldsymbol{C}^{\intercal}\widehat{\boldsymbol{\lambda}}^{(k)}\|_2,
\end{aligned}$$

dove m è il numero di righe della matrice $\boldsymbol{C}$, p è la dimensione del vettore $\boldsymbol{\beta}$, $\epsilon^{\text{abs}} > 0$ è il valore di tolleranza assoluto e $\epsilon^{\text{rel}} > 0$ è quello relativo. Si nota anche che i fattori $\sqrt{m}$ e $\sqrt{p}$ vengono considerati per il fatto che la norme di ℓ_2 sono di dimensione, rispettivamente, $\mathbb{R}^m$ e $\mathbb{R}^p$. Un valore ragionevole per lo stop relativo, invece, potrebbe essere $\epsilon^{\text{rel}} = 10^{-3}$ o $\epsilon^{\text{rel}} = 10^{-4}$. Solitamente la scelta di un valore rispetto ad un altro dipende dalla applicazione. Mentre, per la scelta del criterio di stop assoluto dipende tipicalemnte dalla scala o dal tipo di valore delle variabili.

1.2 Modello di regressione multivariato penalizzato

Con quanto descritto nella sezione di introduzione, si formalizza in questa sezione il modello di regressione multivariato penalizzato. Per la stima dei parametri si utilizza l'algoritmo ADMM descritto in precedenza.

1.2.1 Formalizzazione del modello

Data la matrice risposta $\mathbf{Y} \in \mathbb{R}^{n\times q}$ e la matrice disegno $\mathbf{X} \in \mathbb{R}^{n\times p}$, il modello di regressione lineare multivariato è definito come

$$\mathbf{Y} = \mathbf{XB} + \mathbf{E},$$

dove $\mathbf{B} \in \mathbb{R}^{p\times q}$ e $\mathbf{E} \sim \mathsf{N}_{n\times q}(0, \mathbf{\Sigma}, \mathbf{I}_n)$, che indica una distribuzione Normale matriciale, con media zero, funzione di varianza e covarianza $\mathbf{\Sigma} \in \mathbb{S}^q_{++}$ per le colonne di $\mathbf{Y}$ e $\mathbf{I}_n$, matrice identità di dimensione n, per le righe di $\mathbf{Y}$. La funzione di log-verosmiglianza del modello appena definito senza costanti Rothman et al. (2010) si definisce come segue

$$\ell(\mathbf{B}, \mathbf{\Sigma}) = -\frac{1}{2n}\text{trace}\left[\mathbf{\Sigma}^{-1}(\mathbf{Y} - \mathbf{XB})^\top(\mathbf{Y} - \mathbf{XB})\right] - \frac{1}{2}\log|\mathbf{\Sigma}|\,. \tag{11}$$

Per ridurre il numero dei parametri da stimare si decide di applicare una funzione di penalizzazione di tipo *lasso*. Il problema di minimo per la stima della matrice di parametri di regressione $\mathbf{B}$ risulta

$$\widehat{\mathbf{B}} = \arg\min_{\mathbf{B}} \ell(\mathbf{B}, \mathbf{\Sigma}), \qquad \text{sotto il vincolo:} \quad C(\mathbf{B}) \leq t, \tag{12}$$

dove $C(\mathbf{B}) = \|\boldsymbol{B}\|_1 = \sum_{j=1}^{q}\sum_{i=1}^{p}|b_{ij}|$ dove b_{ij} è un sigolo elemento di $\mathbf{B}$. Il problema di ottimizzazione definito in equazione (12) può essere riscritto come un modello di regressione vincolata nel modo seguente:

$$\widehat{\mathbf{B}} = \arg\min_{\mathbf{B}} \ell(\mathbf{B}, \mathbf{\Sigma}) + \lambda\|\boldsymbol{B}\|_1. \tag{13}$$

Si nota che il problema di ottimizzazione vincolata definito in equazione (13) riporta la verosimiglianza del modello penalizzata per $C(\mathbf{B})$, e $\lambda > 0$ rappresenta il

parametro di regolazione.

1.2.2 Algoritmo ADMM

Per la soluzione del problema di ottimizzazione vincolata definito in equazione (13) si applica l'algoritmo ADMM introdotto da Boyd et al. (2011b). Il problema di minimo quindi si può scrivere utilizzando il concetto di *lagrangiano aumentato* proposto da Boyd et al. (2011b) nel modo seguente:

$$\mathcal{L}_\rho(\mathbf{B}, \mathbf{\Sigma}, \mathbf{Z}, \tilde{\mathbf{U}}) = \ell(\mathbf{B}, \mathbf{\Sigma}) + \frac{\rho}{2}\|\mathbf{B} - \mathbf{Z} + \mathbf{U})\|_F + \lambda\|\mathbf{Z}\|_1, \tag{14}$$

dove $\boldsymbol{U} = (1/\rho)\widetilde{\boldsymbol{U}}$ è il valore duale scalato e $\|\cdot\|_F$ denota la norma di Frobenius di una matrice. Da quanto riportato in (Boyd et al., 2011a, pag. 47), La soluzione del problema in (14) risulta

$$\widehat{\boldsymbol{Z}}^{(k+1)} = \arg\min_{\boldsymbol{Z}} \left(\lambda\|\boldsymbol{Z}\|_1 + \frac{\rho}{2}\|\widehat{\boldsymbol{B}}^{(k)} - \boldsymbol{Z} + \widehat{\boldsymbol{U}}^{(k)}\|_2^2 \right) \tag{15}$$

$$\widehat{\boldsymbol{B}}^{(k+1)} = \arg\min_{\boldsymbol{B}} \left(\ell(\mathbf{B}, \mathbf{\Sigma}) + \frac{\rho}{2}\|\boldsymbol{B} - \widehat{\boldsymbol{Z}}^{(k+1)} + \widehat{\boldsymbol{U}}^{(k)}\|_F \right) \tag{16}$$

$$\widehat{\boldsymbol{U}}^{(k+1)} = \widehat{\boldsymbol{U}}^{(k)} + \widehat{\boldsymbol{B}}^{(k+1)} - \widehat{\boldsymbol{Z}}^{(k+1)}, \tag{17}$$

per $k = 0, 1, \ldots$

Dimostrazione. Prima di procedere, si riscrive la funzione di verosimiglianza in equazione (11) utilizzando le proprietà dell'operatore vec e quelle dell'operatore traccia discusse nelle Proposizioni (A.0.1) e (A.0.2), rispettivamente.

$$\begin{aligned}
\text{trace}\left[\mathbf{\Sigma}^{-1}(\mathbf{Y} - \mathbf{XB})^\top(\mathbf{Y} - \mathbf{XB})\right] &= \text{vec}\left[(\mathbf{Y} - \mathbf{XB})\mathbf{\Sigma}^{-1}\right]^\top \text{vec}\left[(\mathbf{Y} - \mathbf{XB})\right] \\
&= \left[(\mathbf{\Sigma}^{-1} \otimes \boldsymbol{I}_n)\text{vec}(\boldsymbol{Y} - \boldsymbol{XB})\right]^\top \text{vec}\left[(\mathbf{Y} - \mathbf{XB})\right] \\
&= (\boldsymbol{y} - (\boldsymbol{I}_q \otimes \boldsymbol{X})\boldsymbol{\beta})^\top(\mathbf{\Sigma}^{-1} \otimes \boldsymbol{I}_n)(\boldsymbol{y} - (\boldsymbol{I}_q \otimes \boldsymbol{X})\boldsymbol{\beta}) \\
&= \mathcal{Q}(\boldsymbol{\beta}, \mathbf{\Sigma}),
\end{aligned}$$

dove $\mathbf{y} = \text{vec}(\mathbf{Y}) \in \mathbb{R}^{nq}$ e $\boldsymbol{\beta} = \text{vec}(\mathbf{B}) \in \mathbb{R}^{pq}$. La funzione di verosimiglianza

diventa quibdi

$$\begin{aligned}\ell(\mathbf{B},\mathbf{\Sigma}) &= -\frac{1}{2n}\text{trace}\left[\mathbf{\Sigma}^{-1}(\mathbf{Y}-\mathbf{XB})^\top(\mathbf{Y}-\mathbf{XB})\right] - \frac{1}{2}\log|\mathbf{\Sigma}| \\ &= -\frac{1}{2n}\mathcal{Q}(\boldsymbol{\beta},\mathbf{\Sigma}) - \frac{1}{2}\log|\mathbf{\Sigma}| = \ell_1(\boldsymbol{\beta},\mathbf{\Sigma}) \end{aligned} \tag{18}$$

Si scrive ora la derivata parziale per $\boldsymbol{\beta}$, e quindi per $\mathbf{B}$ e si pone uguale a zero. Si procede in due modi, prima derivando (18) nel modo seguente

$$\begin{aligned}\frac{\partial\mathcal{L}_\rho(\boldsymbol{\beta},\mathbf{\Sigma},\boldsymbol{z},\boldsymbol{u})}{\partial\boldsymbol{\beta}} &= \frac{\partial\ell(\mathbf{B},\mathbf{\Sigma})}{\partial\mathbf{B}} \\ &= -\frac{1}{n}(\mathbf{\Sigma}^{-1}\otimes\boldsymbol{X}^\top\boldsymbol{X})\boldsymbol{\beta} + \frac{1}{n}(\mathbf{\Sigma}^{-1}\otimes\boldsymbol{X}^\top)\boldsymbol{y} + \rho\boldsymbol{\beta} - \rho(\boldsymbol{z}-\boldsymbol{u}) = 0\end{aligned}$$

da cui si ottiene

$$\begin{aligned}\widehat{\boldsymbol{\beta}} &= \big(\mathbf{\Sigma}^{-1}\otimes\boldsymbol{X}^\top\boldsymbol{X} + \boldsymbol{I}_{pq}n\rho\big)^{-1}\big((\mathbf{\Sigma}^{-1}\otimes\boldsymbol{X}^\top)\boldsymbol{y} - n\rho(\boldsymbol{z}-\boldsymbol{u})\big) \\ &= \big(n\rho\mathbf{\Sigma}^{-1}\otimes\text{prec}(\widehat{\boldsymbol{B}}_{n\rho})\big)^{-1}\big((\mathbf{\Sigma}^{-1}\otimes\boldsymbol{X}^\top)\boldsymbol{y} - n\rho(\boldsymbol{z}-\boldsymbol{u})\big) \\ &= \Big(\frac{1}{n\rho}\mathbf{\Sigma}\otimes\text{var}(\widehat{\boldsymbol{B}}_{n\rho})\Big)\big((\mathbf{\Sigma}^{-1}\otimes\boldsymbol{X}^\top)\boldsymbol{y} - n\rho(\boldsymbol{z}-\boldsymbol{u})\big) \\ &= \Big(\frac{1}{n\rho}\boldsymbol{I}_q\otimes\text{var}(\widehat{\boldsymbol{B}}_{n\rho})\boldsymbol{X}^\top\boldsymbol{y}\Big) - \Big(\mathbf{\Sigma}\otimes\text{var}(\widehat{\boldsymbol{B}}_{n\rho})\Big)(\boldsymbol{z}-\boldsymbol{u})\end{aligned}$$

dove $\text{prec}(\widehat{\boldsymbol{B}}_\lambda) = \boldsymbol{X}^\top\boldsymbol{X} + \lambda\boldsymbol{I}_p$ e $\text{var}(\widehat{\boldsymbol{B}}_\lambda) = \Big(\text{prec}(\widehat{\boldsymbol{B}}_\lambda)\Big)^{-1}$, che è equivalente alla prima parte della soluzione di una regressione *ridge*. Questa soluzione matricialmente diventa

$$\begin{aligned}\widehat{\boldsymbol{B}} &= \frac{1}{n\rho}\text{var}(\widehat{\boldsymbol{B}}_\lambda)\boldsymbol{X}^\top\boldsymbol{Y} - \text{var}(\widehat{\boldsymbol{B}}_\lambda)(\boldsymbol{Z}-\boldsymbol{U})\mathbf{\Sigma} \\ &= \text{var}(\widehat{\boldsymbol{B}}_\lambda)\Big[\frac{1}{n\rho}\boldsymbol{X}^\top\boldsymbol{Y} - (\boldsymbol{Z}-\boldsymbol{U})\mathbf{\Sigma}\Big].\end{aligned}$$

Nel secondo metodo invece si deriva per $\mathbf{B}$ e si pone uguale a zero

$$\begin{aligned}\frac{\partial\mathcal{L}_\rho(\mathbf{B},\mathbf{\Sigma},\mathbf{Z},\mathbf{U})}{\partial\mathbf{B}} &= \frac{\partial\ell(\mathbf{B},\mathbf{\Sigma})}{\partial\mathbf{B}} \\ &= \mathbf{X}\mathbf{\Sigma}^{-1}\mathbf{X}^\top\mathbf{B} - \mathbf{X}\mathbf{\Sigma}^{-1}\mathbf{Y}^\top + \rho\mathbf{B} - \rho(\mathbf{Z}-\boldsymbol{U}) = 0 \\ &= (\mathbf{X}^\top\mathbf{\Sigma}^{-1}\mathbf{X} + \rho\mathbf{I})\mathbf{B} = \mathbf{X}^\top\mathbf{\Sigma}^{-1}\mathbf{Y} + \rho(\mathbf{Z}-\tilde{\mathbf{U}})\end{aligned}$$

Si ottiene quindi

$$\widehat{\mathbf{B}} = (\mathbf{X}^\top \boldsymbol{\Sigma}^{-1}\mathbf{X} + \rho\mathbf{I})^{-1}\left(\mathbf{X}^\top \boldsymbol{\Sigma}^{-1}\mathbf{Y} + \rho(\mathbf{Z} - \tilde{\mathbf{U}})\right). \tag{19}$$

Si nota che lo stimatore dei parametri di regressione $\widehat{\mathbf{B}}$, calcolato nei due metodi, dipende dalla matrice di varianze-covarianze $\boldsymbol{\Sigma}$. Si calcola, quindi, lo stimatore per $\boldsymbol{\Sigma}$

$$\begin{aligned}\frac{\partial \mathcal{L}_\rho(\mathbf{B}, \boldsymbol{\Sigma}, \mathbf{Z}, \boldsymbol{U})}{\partial \boldsymbol{\Sigma}} &= -\frac{1}{2}\boldsymbol{\Sigma}^{-1} + \frac{1}{2n}\boldsymbol{\Sigma}^{-2}(\mathbf{Y} - \mathbf{XB})^\top(\mathbf{Y} - \mathbf{XB}) = 0 \\ &= \frac{1}{n}\boldsymbol{\Sigma}^{-2}(\mathbf{Y} - \mathbf{XB})^\top(\mathbf{Y} - \mathbf{XB}) = \boldsymbol{\Sigma}^{-1}\end{aligned}$$

ottenendo così

$$\widehat{\boldsymbol{\Sigma}}_{\boldsymbol{B}} = \frac{1}{n}(\mathbf{Y} - \mathbf{XB})^\top(\mathbf{Y} - \mathbf{XB}) = \frac{1}{n}\mathbf{E}^\top\mathbf{E} \tag{20}$$

Dall'equazioni (19) e (20) lo stimatore della matrice dei parametri risulta

$$\hat{\mathbf{B}} = (\mathbf{X}^\top \hat{\boldsymbol{\Sigma}}^{-1}\mathbf{X} + \rho\mathbf{I})^{-1}\left(\mathbf{X}^\top \hat{\boldsymbol{\Sigma}}^{-1}\mathbf{Y} + \rho(\mathbf{Z} - \tilde{\mathbf{U}})\right). \tag{21}$$

Il secondo passo di aggiornamento riguarda la componente $\mathbf{Z}$. Per la stima di questa componente si deve vettorizzare il *lagrangiano aumentato* definito in (14) tramite l'operatore vec. Si definiscano le seguenti quantità

$$\begin{aligned}\mathbf{y} &= \text{vec}(\mathbf{Y}) \in \mathbb{R}^{nq} \\ \boldsymbol{\beta} &= \text{vec}(\mathbf{B}) \in \mathbb{R}^{pq} \\ \mathbf{z} &= \text{vec}(\mathbf{Z}) \in \mathbb{R}^{pq} \\ \boldsymbol{u} &= \text{vec}(\boldsymbol{U}) \in \mathbb{R}^{pq}.\end{aligned}$$

Da queste quantità si ricava la forma vettorizzata

$$\begin{aligned}\mathcal{L}_\rho(\boldsymbol{\beta}, \widehat{\boldsymbol{\Sigma}}, \mathbf{z}, \boldsymbol{u}) = &-\frac{1}{2n}(\mathbf{y} - (\mathbf{X} \otimes \mathbf{I}_q))^\top(\mathbf{I}_q \otimes \widehat{\boldsymbol{\Sigma}}^{-1})(\mathbf{y} - (\mathbf{X} \otimes \mathbf{I}_q)) \\ &+ \frac{\rho}{2}\|\boldsymbol{\beta} - \mathbf{z} + \boldsymbol{u}\|_2^2 + \lambda\|\mathbf{z}\|_1.\end{aligned} \tag{22}$$

Si calcola quindi la derivata di (22) per $\mathbf{z}$ e si pone uguale a zero. Si nota che la

derivata è scomponibile in due parti

$$\frac{\partial \mathcal{L}_\rho(\boldsymbol{\beta}, \widehat{\boldsymbol{\Sigma}}, \mathbf{z}, \boldsymbol{u})}{\partial \mathbf{z}} + \frac{\partial \lambda \|\mathbf{z}\|_1}{\partial \mathbf{z}} = 0.$$

La prima parte è differenziabile e risulta

$$\frac{\partial \mathcal{L}_\rho(\boldsymbol{\beta}, \widehat{\boldsymbol{\Sigma}}, \mathbf{z}, \boldsymbol{u})}{\partial \mathbf{z}} = \rho \mathbf{z} - \rho \boldsymbol{\beta} - \rho \boldsymbol{u} = \rho \left(\mathbf{z} - \boldsymbol{\beta} - \boldsymbol{u} \right).$$

La seconda parte, non differenziabile, si risolve tramite il concetto di subdifferenziale. Si ottiene così che la soluzione per la componente $\mathbf{z}$ è l'operatore *soft-threshold* definito come

$$S_{\frac{\lambda}{\rho}}(\boldsymbol{\beta} + \boldsymbol{u}) = \begin{cases} (\boldsymbol{\beta} + \boldsymbol{u}) - \frac{\lambda}{\rho}, & \text{se} \quad (\boldsymbol{\beta} + \boldsymbol{u}) > \frac{\lambda}{\rho} \\ 0, & \text{se} \quad |(\boldsymbol{\beta} + \boldsymbol{u})| \leq \frac{\lambda}{\rho} \\ (\boldsymbol{\beta} + \boldsymbol{u}) - \frac{\lambda}{\rho}, & \text{se} \quad (\boldsymbol{\beta} + \boldsymbol{u}) < \frac{\lambda}{\rho} \end{cases} \tag{23}$$

Aggiungendo il passo duale si ottiene quindi la soluzione per il problema con *lasso penalty*

$$\begin{aligned} \boldsymbol{\beta}^{k+1} =& \left((\mathbf{X} \otimes \mathbf{I}_q)^\top (\mathbf{I}_q \otimes \hat{\boldsymbol{\Sigma}}^{-1})(\mathbf{X} \otimes \mathbf{I}_q) + \rho \mathbf{I} \right)^{-1} \left((\mathbf{X} \otimes \mathbf{I}_q)^\top (\mathbf{I}_q \otimes \hat{\boldsymbol{\Sigma}}^{-1}) \mathbf{y} + \rho (\mathbf{z}^k - \tilde{\mathbf{u}}^k) \right) \\ \mathbf{z}^{k+1} =& S_{\frac{\lambda}{\rho}}(\boldsymbol{\beta}^{k+1} + \tilde{\mathbf{u}}^k) \\ \tilde{\mathbf{u}}^{k+1} =& \tilde{\mathbf{u}}^k + \boldsymbol{\beta}^{k+1} - \mathbf{z}^{k+1} \end{aligned}$$

□

1.2.3 Versione alternativa dello stimatore di $\boldsymbol{\Sigma}$

In precedenza si è visto che la stima della matrice dei parametri B dipende dalla funzione di varianza-covarianza $\boldsymbol{\Sigma}$. In Equazione 20 si è presentato uno stimatore per $\boldsymbol{\Sigma}$, però questo, dipende dal vero valore dei parametri B, quantità ignota che si vuole stimare. Nella pratica quindi viene utilizzato un altro stimatore che si stima mediante le quantità note.
Per la stima della funzione di varianza-covarianza, sfruttando la distribuzione del

Gaussian matrix-variate, si osserva che

$$\begin{aligned} p(\boldsymbol{Y}, \boldsymbol{B}, \boldsymbol{\Sigma}) &\propto |\boldsymbol{\Sigma}|^{-n/2} \exp\left\{-\frac{1}{2}\text{trace}\left[\boldsymbol{\Sigma}^{-1}(\boldsymbol{Y}-\boldsymbol{X}\boldsymbol{B})^\top(\boldsymbol{Y}-\boldsymbol{X}\boldsymbol{B})\right]\right\} \\ &\propto |\boldsymbol{\Sigma}|^{-n/2} \exp\left\{-\frac{1}{2}\text{trace}\left[\boldsymbol{\Sigma}^{-1}\big(\boldsymbol{S}+(\boldsymbol{B}-\widehat{\boldsymbol{B}})^\top\boldsymbol{X}^\top\boldsymbol{X}(\boldsymbol{B}-\widehat{\boldsymbol{B}})\big)\right]\right\} \\ &\propto \phi_{p\times q}(\boldsymbol{B}|\widehat{\boldsymbol{B}}, \boldsymbol{\Sigma}, (\boldsymbol{X}^\top\boldsymbol{X})^{-1}) \times |\boldsymbol{\Sigma}|^{-(n-q)/2} \exp\left\{-\frac{1}{2}\text{trace}(\boldsymbol{\Sigma}^{-1}\boldsymbol{S})\right\}, \end{aligned} \tag{24}$$

dove $\widehat{B} = (\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top\boldsymbol{Y}$ e $\boldsymbol{S} = (\boldsymbol{Y}-\boldsymbol{X}\widehat{\boldsymbol{B}})^\top(\boldsymbol{Y}-\boldsymbol{X}\widehat{\boldsymbol{B}}) = \boldsymbol{Y}^\top\boldsymbol{P}\boldsymbol{Y}$ con $\boldsymbol{P} = \boldsymbol{I}_n - \boldsymbol{X}(\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}$. Si osserva inoltre, che nello sviluppo della forma quadratica in equazione (24), si ottiene il seguente risultato notevole:

$$(\boldsymbol{Y}-\boldsymbol{X}\widehat{\boldsymbol{B}})^\top\boldsymbol{X}(\boldsymbol{B}-\widehat{\boldsymbol{B}}) = \big(\boldsymbol{Y}^\top\boldsymbol{X} - \widehat{\boldsymbol{B}}^\top\boldsymbol{X}^\top\boldsymbol{X}\big)(\boldsymbol{B}-\widehat{\boldsymbol{B}}) = 0. \tag{25}$$

Integrando $\boldsymbol{B}$ in equazione (24) si ottiene

$$\int_{\mathbb{R}^{p\times q}} p(\boldsymbol{Y}, \boldsymbol{B}, \boldsymbol{\Sigma}) d\boldsymbol{B} = |\boldsymbol{\Sigma}|^{-(n-q)/2} \exp\left\{-\frac{1}{2}\text{trace}(\boldsymbol{\Sigma}^{-1}\boldsymbol{S})\right\},$$

e quindi

$$\begin{aligned} \widehat{\boldsymbol{\Sigma}} &= \arg\max_{\boldsymbol{\Sigma}} |\boldsymbol{\Sigma}|^{-(n-q)/2} \exp\left\{-\frac{1}{2}\text{trace}(\boldsymbol{\Sigma}^{-1}\boldsymbol{S})\right\} \\ &= \arg\max_{\boldsymbol{\Sigma}} -\frac{1}{2}(n-q)\log|\boldsymbol{\Sigma}| - \frac{1}{2}\text{trace}(\boldsymbol{\Sigma}^{-1}\boldsymbol{S}) \end{aligned}$$

La FOC risulta

$$\frac{\partial}{\partial\boldsymbol{\Sigma}} - \frac{1}{2}(n-q)\log|\boldsymbol{\Sigma}| - \frac{1}{2}\text{trace}(\boldsymbol{\Sigma}^{-1}\boldsymbol{S}) = -\frac{1}{2}(n-q)\boldsymbol{\Sigma}^{-1} - \frac{1}{2}\boldsymbol{S}\boldsymbol{\Sigma}^{-2} = 0 \tag{26}$$

da cui si ottiene $\widehat{\boldsymbol{\Sigma}} = \frac{1}{n-q}\boldsymbol{S}$ che dipende da $\widehat{B}$ ma non dal parametro $\boldsymbol{B}$.

1.3 Modello di regressione multivariato con doppia penalizzazione

In precedenza si è presentato il modello multivariato con penalizzazione sulla matrice dei parametri B. In questa sezione si introduce invece una doppia penalizzazione: una, come in precedenza sulla matride dei parametri B, e l'altra sulla funzione di varianza-covarianza $\mathbf{\Sigma}$. La seconda penalizzazione porta principalmente due vantaggi. Il primo è l'effetto di ridurre i parametri da stimare nella matrice di varianza-covarianza, importante quando si considerano casi in cui il numero di colonne della variabile rispsota (q) è elevato. Il secondo vantaggio, invece, è quello di garantire un numero finito alla funzione obiettivo, anche nel caso in cui il numero di colonne della matrice Y (q) è maggiore rispetto al numero di righe (n). Di seguito si formalizza il modello e si mostra come avviene la stima mediante ADMM.

1.3.1 Formalizzazione del modello

Il problema di ottimizzazione può essere riscritto come un modello di regressione vincolata nel modo seguente, (Rothman et al., 2010):

$$(\widehat{\boldsymbol{B}}, \widehat{\mathbf{\Sigma}}) = \arg\min_{\mathbf{B}} \ell(\mathbf{B}, \mathbf{\Sigma}) + \lambda_1 \|\boldsymbol{B}\|_1 + \lambda_2 \|\mathbf{\Sigma}\|_1. \tag{27}$$

Si nota che, oltre alla penalizzazione della matrice dei parametri $\boldsymbol{B}$, si è aggiunta anche la penalizzazione alla componente $\mathbf{\Sigma}$. Per trovare la soluzione si decide di procedere in due step iterativi: il primo, per la stima della componente $\boldsymbol{B}$ è uguale a quello che abbiamo descritto in precedenza, con $\mathbf{\Sigma}$ fissato; nel secondo step, invece, si applica nuovamente l'ADMM per la componente $\mathbf{\Sigma}$ con $\boldsymbol{B}$ fissato. Si vedono ora i passaggi per la soluzione del secondo step. Si definisce la funzione di verosimiglianza

$$\ell_2(\mathbf{B}, \mathbf{\Sigma}) = -\frac{1}{2n} \text{trace}\left[\mathbf{\Sigma}^{-1}\boldsymbol{S}\right] - \frac{1}{2}\log|\mathbf{\Sigma}|\,. \tag{28}$$

con $\boldsymbol{S} = (\mathbf{Y} - \mathbf{XB})^\top(\mathbf{Y} - \mathbf{XB})$ e

$$\mathcal{L}_2(\boldsymbol{B}, \boldsymbol{\Sigma}) = \ell_2(\mathbf{B}, \boldsymbol{\Sigma}) + \lambda_2 \|\boldsymbol{\Sigma}\|_1, \tag{29}$$

con λ_2 parametro di regolazione per $\boldsymbol{\Sigma}$. Per risolvere il problema di minimo dell'equazione in (29) per la stima di $\boldsymbol{\Sigma}$ si decide di utilizzare l'approccio suggerito da Fang et al. (2020), utilizzando una scomposizione della funzione della matrice di varianza e covarianza e poi tramite l'algoritmo ADMM stimare tutte le componenti.
Sia $\boldsymbol{\Sigma}$ definita positiva per definizione, utilizzando la decomposizione di Cholseky modificata a blocchi (vedi Yin and Li, 2012) si può scrivere $\boldsymbol{T\Sigma T}^\top = \boldsymbol{D}$ dove $\boldsymbol{D}$ è una matrice diagonale a blocchi $\boldsymbol{D} = diag(\boldsymbol{D}_1, \ldots, \boldsymbol{D}_m)$, e $\boldsymbol{T}$ è una matrice triangolare inferirore a blocchi

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{I} & \mathbf{0} & \ldots & \mathbf{0} \\ -\boldsymbol{\Phi}_{21} & \boldsymbol{I} & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \mathbf{0} \\ -\boldsymbol{\Phi}_{m1} & -\boldsymbol{\Phi}_{m2} & \ldots & \boldsymbol{I} \end{bmatrix}$$

da cui si ottiene $\boldsymbol{\Sigma}^{-1} = \boldsymbol{T}^\top \boldsymbol{\Omega T}$ dove $\boldsymbol{\Omega} = \boldsymbol{D}^{-1}$. Da questo risultato si riscrive l'equazione (29) come

$$\mathcal{L}_2(\boldsymbol{B}, \boldsymbol{T}, \boldsymbol{\Omega}) = \text{trace}\left[\boldsymbol{T}^\top \boldsymbol{\Omega T S}\right] - \log|\boldsymbol{\Omega}| + \lambda_1 P_1(\boldsymbol{T}) + \lambda_2 P_2(\boldsymbol{\Omega}), \tag{30}$$

dove $\boldsymbol{\Omega} = diag(\boldsymbol{\Omega}_1, \ldots, \boldsymbol{\Omega}_m)$ con $\boldsymbol{\Omega}_j = \boldsymbol{D}_j^{-1}$. Per le due funzioni di penalizzazioni, invece, si propongono le seguenti funzioni (Fang et al., 2020)

$$P_1(\boldsymbol{T}) = \sum_{j=1}^{m-1} ||\boldsymbol{T}_{g\ell}||_F \tag{31}$$

$$P_2(\boldsymbol{\Omega}) = |\boldsymbol{\Omega}|_1 = \sum_{j=1}^{m-1} |\boldsymbol{\Omega}|_1. \tag{32}$$

Si osserva che l'equazione definita (30) non è convessa, ma si può definire *biconvessa* per $(\boldsymbol{T}, \boldsymbol{\Omega})$. Quindi, fissando $\boldsymbol{T}$ e risolvendo per Ω o viceversa, si ottiene un problema di minimo convesso. Questa procedura di stima viene chiamata *BiConvex*

Blockwise Regularization (BCBR).

1.3.2 Soluzione del BCBR

In questa sezione viene descritto l'algoritmo di ottimizzazione per risolvere l'equazione in (30). Si decide di rsolvere questo problema con il metodo proposto da Gorski et al. (2007) chiamato *Alternate Convex Search* (ACS). Si procede, quindi, in due step ricorsivi. Nel primo si considera $\boldsymbol{T}$ fissato stimato allo step d, per semplicità si scrive $\hat{\boldsymbol{T}} = \hat{\boldsymbol{T}}^{(d)}$. Il problema di minimizzazione diventa

$$\hat{\boldsymbol{\Omega}}^{(d+1)} = \arg\min_{\boldsymbol{\Omega}>0} \left[\text{trace}(\boldsymbol{\Omega}\hat{\boldsymbol{T}}\boldsymbol{S}\hat{\boldsymbol{T}}^{\top}) - \log|\boldsymbol{\Omega}| + \lambda_2|\Omega|_1 \right]. \tag{33}$$

Per la soluzione del problema in (33) si ricorda che i blocchi della matrice $\boldsymbol{T}$ possono essere interpretati come i coefficienti negativi matriciali di un modello VAR (Fang et al., 2020). Il problema si riduce in

$$\hat{\boldsymbol{\Omega}}_j^{(d+1)} = \arg\min_{\boldsymbol{\Omega}_j>0} \left[\text{trace}(\boldsymbol{\Omega}_j\tilde{\boldsymbol{S}}_j(\hat{\boldsymbol{\Phi}}_j)) - \log|\boldsymbol{\Omega}_j| + \lambda_2|\boldsymbol{\Omega}_j|_1 \right], \tag{34}$$

che corrisponde al problema di stima della matrice di varianza-covarianza ℓ_1-penalizzata (Yuan and Lin, 2006, pag. 49-67). La stima di $\hat{\boldsymbol{\Omega}}_j$ si ottiene utilizzando l'algoritmo *graphical lasso* (Hastie et al., 2015, pag. 248-251), e corrisponde a

$$\hat{\boldsymbol{\Omega}}_j = \left(\tilde{\boldsymbol{S}}_j(\hat{\boldsymbol{\Phi}}_j) + \lambda_2\boldsymbol{\Theta}_j \right)^{-1} \tag{35}$$

Nel secondo step, invece, si fissa $\boldsymbol{\Omega}$, utilizzando la stima in (35) e si trova la soluzione di $\boldsymbol{T}$. Sia $\hat{\boldsymbol{\Omega}} = \hat{\boldsymbol{\Omega}}^{(d+1)}$, la soluzione per $\boldsymbol{T}$ si ottiene

$$\hat{\boldsymbol{T}}^{(d+1)} = \arg\min_{\boldsymbol{T}} \left[\text{trace}(\boldsymbol{T}^{\top}\hat{\boldsymbol{\Omega}}\boldsymbol{T}\boldsymbol{S}) + \lambda_1 \sum_{l=1}^{m-1} \|\boldsymbol{T}_{g_l}\|_F \right]. \tag{36}$$

Per la soluzione del problema in (36) si può utilizzare nuovamente il metodo ADMM. Prima si riscrive il problema in (36) come

$$\hat{\boldsymbol{T}}^{(d+1)} = \min_{\boldsymbol{T},\boldsymbol{\Gamma}} \left[\operatorname{trace}(\boldsymbol{T}^\top \hat{\boldsymbol{\Omega}} \boldsymbol{T} \boldsymbol{S}) + \lambda_1 \sum_{l=1}^{m-1} \|\boldsymbol{T}_{g_l}\|_F \right] \quad \text{sotto il vincolo : } \quad \boldsymbol{T} = \boldsymbol{\Gamma}. \tag{37}$$

Prima si definisce il *lagrangiano aumentato* come

$$\mathcal{L}_\rho(\boldsymbol{T}, \boldsymbol{\Gamma}, \boldsymbol{U}) = \operatorname{trace}(\boldsymbol{T}^\top \hat{\boldsymbol{\Omega}} \boldsymbol{T}) + \langle \boldsymbol{U}, \boldsymbol{T} - \boldsymbol{\Gamma} \rangle + \frac{\rho}{2} \|\boldsymbol{T} - \boldsymbol{\Gamma}\|_F. \tag{38}$$

La sequenza iterativa per la soluzione di del problema in (37) si riduce quindi in

$$\hat{\boldsymbol{T}}^{(k+1)} = \arg\min_{\boldsymbol{T}} \mathcal{L}_\rho(\boldsymbol{T}, \hat{\boldsymbol{\Gamma}}^{(k)}, \hat{\boldsymbol{U}}^{(k)}) \tag{39a}$$

$$\hat{\boldsymbol{\Gamma}}^{(k+1)} = \arg\min_{\boldsymbol{\Gamma}} \mathcal{L}_\rho(\hat{\boldsymbol{T}}^{(k+1)}, \boldsymbol{\Gamma}, \hat{\boldsymbol{U}}^{(k)}) \tag{39b}$$

$$\hat{\boldsymbol{U}}^{(k+1)} = \hat{\boldsymbol{U}}^{(k)} + \rho(\hat{\boldsymbol{T}}^{(k+1)} - \hat{\boldsymbol{\Gamma}}^{(k+1)}). \tag{39c}$$

Si definisce ora la soluzione chiusa per $\boldsymbol{T}$, considerando che i termini a blocchi di questa matrice possono essere interpretati come i parametri matriciali di un modello autoregressivo vettoriale

$$\operatorname{vec}(\boldsymbol{\Phi}_j^{(k+1)}) = \left(2\boldsymbol{S}_{1:(j-1),1:(j-1)} \otimes \hat{\boldsymbol{\Omega}}_j + \rho \boldsymbol{I}_{q\times q(j-1)}\right)^{-1} \operatorname{vec}\left(2\hat{\boldsymbol{\Omega}}_j \boldsymbol{S}_{j,1:(j-1)} + \rho \boldsymbol{\Gamma}_j^{(k)} - \boldsymbol{U}_j^{(k)}\right). \tag{40}$$

Per la componente $\boldsymbol{\Gamma}$, invece, si considera il problema di minimizzazione seguente

$$\min_{\boldsymbol{A}^{(l)}} \left[\|\boldsymbol{Z}^{(t)} - \frac{\lambda_1}{\rho} \sum_{l=1}^{m-1} \boldsymbol{A}^{(l)}\|_F^2 \right] \quad \text{sotto il vincolo : } \quad \|\boldsymbol{A}_{g_l}^{(l)}\|_F \leq 1, \quad \|\boldsymbol{A}_{g_l^c}^{(l)}\|_F = 0, \tag{41}$$

dove $\boldsymbol{Z}^{(k+1)} = \boldsymbol{T}^{(k+1)} + \frac{1}{\rho}\boldsymbol{U}^{(k)}$. Date le soluzioni per $(\hat{\boldsymbol{A}^{(1)}}, \ldots, \hat{\boldsymbol{A}^{(m-1)}})$, la soluzione per (39b) è data da

$$\hat{\boldsymbol{\Gamma}}^{(k+1)} = \boldsymbol{Z}^{(k+1)} - \frac{\lambda_1}{\rho} \sum_{l=1}^{m-1} \hat{\boldsymbol{A}^{(l)}}. \tag{42}$$

1.4 Selezione dei parametri di regolazione

In letteratura ci sono vari modi per scegliere i parametri di regolazione (λ_1, λ_2). Fang et al. (2020) consigliano di scegliere i valori di (λ_1, λ_2) che minimizzano il BIC (*Bayesian Information criterion*), il quale garantisce la sparsità e l'acuratezza degli stimatori simultanei. Anche Rothman et al. (2010) consigliano un approccio utilizzando la convalida incrociata. Nello specifico considerando k-fold nella convalida incrociata, per selezionare λ_1 e λ_2 ottimi si può utilizzare la seguente formula

$$\underset{\lambda_1,\lambda_2}{\operatorname{argmin}} \sum_{k=1}^{K} \left\| Y^{(k)} - X^{(k)} B^{(-k)} \right\|_F^2 ,$$

dove $Y^{(k)}$ è la matrice delle risposte con le osservazioni nel k-esimo fold, $X^{(k)}$ è la matrice dei predittori delle osservazioni nel k-esimo fold e $B^{(-k)}$ è la stima dei coefficienti di regressione ottenuta con le osservazioni al di fuori del k-esimo fold, che dipende dai parametri di regolazione.

Capitolo 2

Studio di simulazione

Per valutare l'adeguatezza del metodo di stima presentato nel Capitolo 1, si è proceduto testandolo e confrontandolo con altri metodi in uno studio di simulazione. In questo capitolo si descrive la procedura adottata in questo studio di simulazione. Nella prima sezione vengono descritti i vari modelli confrontati. Nella seconda sezione viene presentato come vengono simulati i dati. In particolare si è fatta attenzione alla matrice dei parametri B, che deve essere sparsa. Inoltre, la matrice degli errori E si sono impostate con diverse correlazioni per studiare l'efficacia di penalizzare simultaneamente i parametri e di includere la stima della funzione di varianza-covarianza nello stimatore dei parametri. Infine, vengono presentati i risultati e i vari confronti su metriche specifiche.

2.1 Valutazione

Per valutare le performance dell'algoritmo implementato, descritto nel primo capitolo e presentato brevemente in appendice, si decide di applicarlo ad uno studio di simulazione fatto da Rothman et al. (2010) e confrontarlo con i vari modelli da loro testati, tenendo in considerazione principalmente quello da loro implementato nell'articolo. I vari modelli presenti nell'articolo fanno riferimento tutti a modelli di penalizzazione lasso che stimano matrici di parametri $\mathbf{B}$ sparse. Di seguito si descrivono i vari modelli

- *Lasso*: vengono stimate q regressioni Lasso, ognuna con lo stesso parametro di regolazione λ;
- *Separate Lasso*: vengono stimati q regressioni Lasso, ma ciascuna con diversi parametri di regolazione λ;
- *MRCE*: il modello implementato da Rothman et al. (2010) presentato nell'algoritmo 2;
- *Approx. MRCE*: il modello implementato da Rothman et al. (2010) presentato nell'Algoritmo 3;
- *MLRS*: codice riportato in Appendice B con matrice di varianza e covarianza fissata;
- *Approx. MLRS*: codice riportato in Appendice B con matrice di varianza e covarianza che si aggiorna ad ogni passo di iterazione.

Oltre a descrivere i vari modelli testati è importante descrivere anche come vengono scelti i vari parametri di regolazione. Nel caso dei primi due modelli, si sceglie il λ che restituisce il valore minimo di devianza in un insieme di verifica anch'esso simulato secondo la precedura di Rothman et al. (2010). Per i vari parametri del modello implementato in questo articolo invece si sono utilizzati i valori di default per i criteri di stop. Mentre, per i parametri di regolazione λ e ρ si è seguito il processo analogo usato da Rothman et al. (2010), nello stesso insieme di convalida simulato per gli altri modelli.
Il modello lineare multivariato penalizzato tramite il metodo MRCE presentato da Rothman et al. (2010) è stato applicato anche per vedere l'adeguatezza del metodo implementato in questo lavoro. Infine, si decide di stimare i parametri della matrice B anche tramite OLS per avere così un modello *threshold* da seguire.

2.2 Simulazioni dei dati

In ogni replica per ogni modello, si genera una matrice predittiva X di dimensioni $n \times p$ con righe estratte indipendentemente da una distribuzione normale multivariata $N_p(0, \mathbf{\Sigma}_X)$, dove $\mathbf{\Sigma}_X = [\sigma_{X_{ij}}]$ è dato da $\sigma_{X_{ij}} = 0.7^{|i-j|}$. Questo modello per i

predittori è stato utilizzato anche da Yuan et al. (2007) e da Peng et al. (2010). Si noti che tutti i predittori sono generati con la stessa varianza marginale unitaria. La matrice degli errori $\boldsymbol{E}$ è generata indipendentemente con righe estratte indipendentemente da $N_q(0, \boldsymbol{\Sigma}_E)$. Per la covarianza degli errori si considerano invece due modelli:

- Un modello autoregressivo di ordine 1 AR(1): $\sigma_{ij} = \rho_E^{|i-j|}$, con $\rho \in (0, 0.9)$
- Errori Gaussiani Frazionali (FGN):

$$\sigma_{E_{ij}} = 0.5((|i-j|+1)^{2H} - 2|i-j|^{2H} + (|i-j|-1)^{2H})$$

  con il valore del parametro di Hurst $H = 0.9, 0.95$

La funzuone di varianza-covarianza inversa degli errori per il modello AR(1) è una matrice sparsa trinagolare diagonale e la sua matrice di covarianza è densa. Il modello FGN è un esempio standard di dipendenza a lungo raggio e sia la covarianza degli errori che la sua inversa sono matrici dense.
Se si varia H si ottiene un diverso grado di dipendenza, se H = 0.5 corrisponde a una sequenza di errori indipendenti e identicamente distribuiti. Se H = 1 il modello degli errori corrisponde a una serie completamente correlata. La dimensione del campione è fissata a n = 50 per tutti i modelli.
Si generano inoltre le matrici di coefficienti sparse B in ogni replica utilizzando il prodotto elemento per elemento tra le matrici,

$$\boldsymbol{B} = WKQ,$$

dove W è generato da colonne indipendenti con distribuzione $N(0, 1)$, K è genrato da elementi indipendenti con distribuzione di Bernoulli e probabilità di successo s_1, e Q ha righe formate da 0 o 1, posizionati secondo distribuzione di Bernoulli indipendente con probabilità di successo s_2 per determinare se la riga del vettore è 1 o è formata da 0. Generando B in questo modo, ci si aspetta che $(1 - s_2)p$ predittori siano irrilevanti per tutte le q risposte, e ci si aspetta che ogni predittore rilevante sia rilevante per $s_1 q$ delle variabili di risposta.

2.3 Metriche per il confronto tra modelli

Per valutare le performance dei vari modelli si segue l'approccio utilizzato da Yuan et al. (2007). Principalmente si definsicono tre metriche. La prima è definita

$$ME(\hat{\boldsymbol{B}}, \boldsymbol{B}) = \text{trace}\left[(\hat{\boldsymbol{B}} - \boldsymbol{B})^T \Sigma_X (\hat{\boldsymbol{B}} - \boldsymbol{B})\right] \tag{1}$$

che considera la matrice dei parametri stimati, la matrice veri parametri e la funzione di varianza e covarianze della matrice X. Le altre due metriche misurano il livello di ricognizione di sparsità della matrice utilizzando i veri positivi (TPR) e i veri negativi (TPR) nel modo seguente

$$TPR(\hat{\boldsymbol{B}}, \boldsymbol{B}) = \frac{\left\{\#(i,j) : \hat{b}_{i,j} \neq 0 \quad \text{and} \quad b_{i,j} \neq 0\right\}}{\#(i,j) : b_{i,j} \neq 0}$$
$$TNR(\hat{\boldsymbol{B}}, \boldsymbol{B}) = \frac{\left\{\#(i,j) : \hat{b}_{i,j} = 0 \quad \text{and} \quad b_{i,j} = 0\right\}}{\{\#(i,j) : b_{i,j} = 0\}}.$$

Si considerano queste metriche in modo simultaneo visto che nel caso della stima OLS i parametri stimati sono tutti diversi da zero.

2.4 Procedimento di stima dei parametri

Le simulazioni sono state eseguite seguendo Rothman et al. (2010). Si sono genrate le varie compoenenti come descritto in precedenza. La matrice dei parametri B presa in considerazione è formata da $p = 15$ righe e $q = 15$ colonne. Il numero di righe considerate per la matrice X invece si è fissato a 50.
Come nell'articolo precedente si è deciso di testare i vari modelli al variare di ρ_E, valore che determina la correlazione autoregressiva degli errori del modello, e al variare della probabilità di sparsita degli zeri nella matrice B (valore s_1). Per ogni test di simulazione sono state effettuate 50 replicazioni.
Si discute ora come è stata effettuata la scelta dei parametri di regolazione per il modello lineare multivariato penalizzato stimato con ADMM e con il metodo MRCE, e le regressioni penalizzate lasso calcolate separatamente. Nel caso del modello lineare multivariato penalizzato stimato secodno ADMM, si sono testati

diversi valori del parametro λ e ρ. La coppia che ha presentato il valore della metrica in equazione (1) minore è stata selezionata. Per il modello lineare multivariato penalizzato con il metodo MRCE, invece, è stata sfruttata la convalida incrociata testanto una griglia di valori per λ_1 e λ_2. In questo modo è stata selezionata la coppia di valori che presentava un errore medio di convalida incrociata minore. Anche nel caso dell'applicazione delle regressioni LASSO separate è stato utilizzato l'errore di convalida per la scelta del parametro di regolazione. Nel primo caso, dove λ è fisso per ogni $q-$regressione LASSO è stata presa la media di ogni λ ottimo singolare.
Oltre ai modelli descritti in precedenza è stata applicata anche la stima OLS, che verrà confrontata con gli altri come modello *threshold*.
Oltre a stimare la matrice dei parametri B di dimensioni $p = 15$ e $q = 15$ si decide di ripetere lo stesso procedimento utilizzando anche matrici sparse di altre dimensioni. Per l'esattezza sono state testate anche dimensioni della matrice B pari a $p = 10$ e $q = 20$ e, solo applicando i modello multivariato penalizzato stimato con ADMM e le regressioni LASSO separato nel caso in cui B ha dimensioni pari a $p = 20$ e $q = 20$.
Si decide poi di procedere allo stesso modo, simulando la matrice E utilizzando gli errori Gaussiani Funzionali. In questo caso si sono stimati i parametri per diverse dimensioni della matrice B. Anche in questo caso si cercherà di vedere se considerando modelli con struttura multivariata e con una selezione simultanea dei parametri portano dei risultati migliori in termini di errore del modello. In questo caso, i parametri di regolazione sono stati selezionati testando più valori e scegliendo quelli che riportavano un errore di convalida incrociata medio inferiore. Si è deciso di stimare i parametri al variare del parametro relativo al parametro ρ_E per due aspetti principali. Il primo, per soddisfare i criteri dell'applicazione del metodo MRCE. Il secondo motivo perché ci si aspetta che, aumentando l'autocorrelazione tra gli errori, valore indicato da ρ_E, i modelli lineari multivariati, quindi quelli stimati secondo ADMM e il metodo MRCE, riportino valori più attendibili visto che la selzione viene eseguita simultaneamente, considerando quindi anche le correlazioni tra le diverse variabili.

2.5 Risultati

Quello che emerge dai risultati delle varie simulazioni conferma ciò che hanno riportato Rothman et al. (2010). Guardando le figure in Appendice che presentano l'andamento dell'errore medio del modello al variare della correlazione si nota che considerando la stima e la penalizzazione simultanea dei parametri della matrice B riporta valori più bassi in termini di media di errore del modello. Quindi quando si stimano i parametri utilizzando modelli penalizzati multivariati si riportano errori medi del modelli più bassi rispetto alla stima utilizzando regressioni di tipo LASSO separate per numero di colonne di Y.
Nello specifico, si nota come in Figura A.1, in cui si sono applicati diversi modelli per la stima della matrice B di dimensioni $p = 15$ e $q = 15$, con valori di s_1 e s_2 rispettivamente pari a 0.1 e 1 si nota che, se si considerano errori non correlati tra loro, il valore d'errore medio del modello risulta essere simile. All'aumentare dell'autocorrelazione la stima dei modelli multivariati penalizzati riportano un errore del modello inferiore rispetto alla stima mediante $q-$regressioni LASSO separate. In Figura A.2, invece, si è provato ad alzare il valore di s_1 ponendolo a 0.5. In questo caso, il modello multivariato penalizzato stimato attraverso MRCE, all'aumentare di ρ_E, nello specfico $\rho_E = 0.9$, riporta nettamente un valore d'errore medio del modello inferiore. Anche il modello multivariato stimato con l'ADMM segue l'andamento decrescente dell'errore medio del modello.
In Figura A.3 è stato riportato l'andamento dell'erroe medio del modello nel caso in cui la matrice dei parametri B ha dimensione $p = q = 20$. I parametri da stimare sono 400. Anche in questo caso, la stima multivariata utilizzando l'ADMM ha riportato dei valori di errore medio del modello decrescenti. All'opposto, invece, la stima mediante *separate LASSO*, nel caso in cui gli errori sono altamente correlati ha riportato un valore d'errore medio del modello simile a quello prodotto dalla stima OLS multivariata.
Infine, in figura A.4, è stato riportato l'ultimo test di simulazione per la stima di una matrice dei parametri B di dimensione $p = 10$ e $q = 20$. I valori prodotti sono più variabili dei casi precedenti. Si può comunque vedere che in corrispondenza di $\rho_E = 0.9$ ancora una volta i modelli che mantengono una struttura multivariata continuano a presentare valori d'errore medio del modello inferiori rispetto agli altri due metodi.

Tabella 2.1: Errori medi del modello considerando $\boldsymbol{E}$ come matrice degli errori Gaussiani Frazionali (FGN). Sono riportate le medie degli errori del modello e i rispettivi *standard error* tra parentesi basate su 50 replicazioni con $n = 50$. I parametri di regolazione sono stati selzionati testando diversi valori mediante convalida incrociata.

p	q	H	s_1,s_2	**ADMM**	**MRCE**	**LASSO**	**OLS**
15	15	0.90	0.1, 1	1.60	0.99	1.62	4.58
				(0.54)	(0.19)	(0.42)	(0.99)
15	15	0.95	0.1, 1	1.29	0.65	1.55	4.45
				(0.44)	(0.15)	(0.66)	(1.31)
10	20	0.90	0.1, 1	1.24	0.86	1.13	4.02
				(0.33)	(0.17)	(0.31)	(0.79)
10	20	0.95	0.1, 1	1.00	0.54	1.02	3.89
				(0.38)	(0.13)	(0.42)	(1.20)
20	20	0.90	0.1, 1	2.37	1.87	3.09	7.96
				(0.54)	(0.29)	(0.75)	(1.34)
20	20	0.95	0.1, 1	2.44	1.16	3.50	7.92
				(0.66)	(0.17)	(1.18)	(2.08)

Per quanto riguarda i modelli applicati considerando come matrice E degli errori Gaussiani Frazionali, si può notare che, anche in questo caso, all'aumentare della correlazione tra gli errori i modelli con una struttura multivariata riportano degli errori medi del modello inferiori rispetto al *separate LASSO*. I risultati delle simulazioni per questo caso sono riportati in Tabella 2.1. Si può notare che all'aumentare dei parametri aumentano anche i valori medi di errore del modello. Tra i due metodi che mantengono una struttura multivariata del modello per la stima e la selezione dei parametri, il modello stimato tramite MRCE riporta dei valori di errore medio inferiore e uno *standard error* inferiore, anche se in questo caso non c'è sparsità nell'inversa della funzione di covarianza dell'errore. Si nota comunque che nel caso della stima tramite ADMM riporta anche questo valori inferiori rispetto al modello che considera delle penalizzazioni separate. Sarebbe interessante testare il valore medio dell'errore del modello considerando diversi valori per le matrici U e Z e verificare se porta dei miglioramenti anche in termini di velocità di convergenza dell'algoritmo.

Capitolo 3

Applicazione con dati reali

In questo capitolo si applica l'algoritmo per la stima dei parametri ad un dataset reale. Si decide di stimare un modello autoregressivo vettoriale di ordine uno (VAR(1)), riscrivendo tale modello in forma multivariata. La prima sezione, quindi, descrive come riscrivere un modello VAR in forma di modello di regressione con risposta multivariata, poi segue una breve spiegazione dei dati e infine l'applicazione dell'algoritmo implementato.

3.1 VAR(1) in forma compatta

Si assume $y_t \in \mathbb{R}^k$ per $t = 0, \pm 1, \pm 2, \dots$ segue un processo autoregressivo vettoriale definito come

$$\boldsymbol{y}_t = \boldsymbol{\nu} + \boldsymbol{\Theta}\boldsymbol{y}_{t-1} + \boldsymbol{\xi}_t \qquad t = 1, 2 \dots, T, \tag{1}$$

con $\boldsymbol{\nu}$ $in\mathbb{R}^k$, $\boldsymbol{y}_{t-1} \in \mathbb{R}^k$, $\boldsymbol{\Theta} \in \mathbb{R}^{k \times k}$ e $\boldsymbol{\xi} \in \mathbb{R}^k$ con distribuzione $\boldsymbol{\xi}_t \sim WN(0, \Sigma_\xi)$, dove Σ_ξ è la matrice di varianza-covarianza del termine d'errore $\boldsymbol{\xi}_t$ ed è assunta non singolare e definita positiva. Si assume inoltre che le radici z dell'equazione caratteristica $|\boldsymbol{\Psi}(z)| = 0$ siano in modulo strettamente maggiori di uno. I parametri $\boldsymbol{\nu}$, $\boldsymbol{\Theta}$ e Σ sono ignoti e vengono stimati.
Per la stima dei parametri si definsce quindi la forma "compatta" o forma "ma-

triciale". Siano

$$Y = \begin{bmatrix} \boldsymbol{y}_1^\top \\ \boldsymbol{y}_2^\top \\ \vdots \\ \boldsymbol{y}_T^\top \end{bmatrix} \in \mathbb{R}^{T\times k} \qquad X = \begin{bmatrix} \boldsymbol{z}_0^\top \\ \boldsymbol{z}_1^\top \\ \vdots \\ \boldsymbol{z}_{T-1}^\top \end{bmatrix} \in \mathbb{R}^{T\times(k+1)}$$

$$\boldsymbol{\Theta} = \begin{bmatrix} \boldsymbol{\nu}^\top \\ \boldsymbol{\Phi}_1^\top \end{bmatrix} \in \mathbb{R}^{(k+1)\times k} \qquad E = \begin{bmatrix} \boldsymbol{\xi}_1^\top \\ \boldsymbol{\xi}_2^\top \\ \vdots \\ \boldsymbol{\xi}_T^\top \end{bmatrix} \in \mathbb{R}^{T\times k}$$

dove Y è la matrice con le colonne che corrispondono ai valori delle serie storiche al tempo t, con $\boldsymbol{y}_t = [y_{1,t}, y_{2,t}, \ldots, y_{k,t}]$. La matrice X, è formata dagli elementi z_j con $j = 0, 1, \ldots, T-1$ ed è definito come

$$z_t = [1 \quad y_t \quad y_{t-1}] \in \mathbb{R}^{1\times(k+1)}.$$

La matrice $\boldsymbol{\Theta}$ invece è la matrice dei parametri composta dalle quantità definite in equazione (1). La matrice E è la matrice dei termini di errore del modello.
Quindi, dato il processo autoregressivo vettoriale per $\boldsymbol{y}_t \in \mathbb{R}^k$, $\boldsymbol{y}_t \sim VAR(1)$ la cui dinamica è definita in equazione (1), con $\xi_t \sim WNN(0, \Sigma)$, osservato per T periodi campionari e per p periodi pre-campionari, si ottiene la seguente rappresentazione matriciale

$$Y = X\boldsymbol{\Theta} + E, \qquad E \sim MN_{T,k}(0, \Sigma, I_t), \tag{2}$$

dove la matrice degli errori E segue una distribuzione *Gaussiana matrix-variate*, con matrice di locazione formata da zeri, e matrici $\Sigma \in \mathbb{R}^{k\times k}$ e $I_t \in \mathbb{R}^{p\times p}$ matrici di varianza simmetriche e definite positive. Si riportano più dettagli della distribuzione *Gaussiana matrix-varia* in A.0.3. Si nota che il modello in equazione (2) corrisponde al modello descritto nel primo capitolo, quindi per la stima dei parametri si può applicare il procedimento descritto sempre nel Capitolo 1.

3.2 Descrizione dei dati

Come dataset di riferimento si decide di utilizzare lo stesso utilizzato da Rothman et al. (2010), relativo nove serie di log-rendimenti settimanali di stocks dal 2004. Si è deciso di utilizzare questi dati perché così si possono confrontare i risultati con i modelli stimati da Rothman et al. (2010). Come descrivono Yuan et al. (2007), i dati sono relativi alle nove compagnie che secondo *Fortune* hanno presentato una *revenue* maggiore nel 2003. Si è deciso, come hanno fatto gli autori, di scartare la serie relativa a Chevron. Gli stocks rilevati sono stati scaricati da "Yahoo Finance" tramite l'apposita libreria R. Nelle Figure A.5 e nelle Figure A.6 sono state riportate rispettivamente le serie storiche settimanali degli stocks e le serie storiche dei log-rendimenti settimanali. Si nota che i log-rendimenti seguono un processo stazionario. Viene riportata anche una tabella con le misure di sintesi dei vari log-rendimenti delle serie.

Tabella 3.1: Statistiche principali dei nove log-rendimenti settimanali del 2004 presi in considerazione

	WAL	**EXN**	**GM**	**FORD**	**GE**	**CP**	**CG**	**IBM**	**AIG**
nobs	52.00	52.00	52.00	52.00	52.00	52.00	52.00	52.00	52.00
NAs	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Minimum	-0.05	-0.09	-0.10	-0.08	-0.06	-0.10	-0.09	-0.06	-0.15
Maximum	0.05	0.06	0.14	0.07	0.06	0.08	0.10	0.09	0.07
1. Quartile	-0.01	-0.02	-0.02	-0.04	-0.01	-0.02	-0.02	-0.01	-0.02
3. Quartile	0.01	0.02	0.03	0.01	0.02	0.02	0.02	0.03	0.02
Mean	-0.00	-0.00	0.00	-0.01	0.00	0.00	0.00	0.01	-0.00
Median	-0.00	0.00	0.00	-0.01	-0.00	0.00	-0.00	0.01	-0.00
Sum	-0.01	-0.03	0.11	-0.53	0.14	0.07	0.15	0.43	-0.04
SE Mean	0.00	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.00
LCL Mean	-0.01	-0.01	-0.01	-0.02	-0.00	-0.01	-0.01	0.00	-0.01
UCL Mean	0.01	0.01	0.02	0.00	0.01	0.01	0.01	0.01	0.01
Variance	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Stdev	0.02	0.03	0.05	0.04	0.02	0.03	0.04	0.02	0.03
Skewness	0.02	-0.60	0.25	0.13	0.00	-0.28	0.39	0.12	-1.41
Kurtosis	-0.03	0.36	0.39	-0.81	-0.21	0.57	-0.02	1.07	5.80

In questa sezione si descrive anche la suddivsione di questo dataset in dataset di stima e dataset di convalida. Per coerenza si decide di seguire ciò che ha fatto

Yuan et al. (2007) e cioè di utilizzare le prime 26 settimane come dataset di stima mentre le restanti come datasete di verifca su cui calcolare le diverse metriche.

3.3 Stima dei parametri

Si consideri il modello in equazione (2) e si stima la matrice $\mathbf{\Theta}$ mediante il modello ADMM descritto nel primo capitolo. Prima di procedere è necessario spiegare come sono stati impostati i vari parametri di *tuning*. Come parametri di stop dell'algoritmo sono stati scelti i valori di default. Per l'esattezza, i valori relativi agli errori primari e agli errori duali sono stati fissati a 10^{-3}. Anche i valori relativi a τ e μ sono stati fissati relativamente a 0.4 e 0.5.
I parametri di regolazione λ e ρ sono stati selezionati testando diversi valori di una griglia di 100 per entrambi i parametri, quindi si sono testati dieci mila modelli, uno per ogni simulazione. In questo modo è stato possibile valutare quale coppia ha fornito l'errore quadratico medio nell'insieme di convalida. In Figura A.7 e in Figura A.8 sono stati riportati gli andamenti dell'errore quadratico medio moltiplicato per mille calcolato nell'insieme di convalida al variare dei valori di λ e ρ. Per quanto riguarda agli andamenti dell'errore quadratico medio ndel dataset di convalida al variare di λ gli andamenti presentano molta variabilità. L'errore quadratico minimo sul dataset di convalida ricade per la maggior parte degli *stocks* su λ pari a 0.51. L'andamento comunqie presenta molti picchi e una forta variabilità. Quindi sono stati testate più simulazioni in un intorno di 0.51. Più interessante è l'andamento dell'errore quadratico medio di convalida al variare del parametro ρ. Si nota che per le diverse variabili si ha un valore di ρ ottimale diverso. Per l'esattezza si ha che per le variabili **WAL**, **FORD** e **IBM** il valore ottimale di ρ risulta 0.3. Mentre negli altri casi si aggira intorno ad un range di 0.4 e di 0.5. Si decide, quindi, di fissare come valori per i parametri di regolazione λ e ρ rispettivamente 0.528 e 0.414.
Si stimano ora i parametri della matrice B applicando il modello all'interno insieme di stima delle 26 settimane. Oltre ad aver settato i parametri di regolazione, si sono impostati come valori di stop rispettivamente 0.0001 e 0.0001 per i residui primali e per i residui duali, per permettere all'algoritmo di fare più iterazioni. Inoltre, per i parametri relativi a μ e τ si sono lasciati dei valori di default rispettivamente 10 e 0.04.

Si sono succesivamente stimati due modelli per confrontare i risultati. Si è stimato prima il modello OLS e poi il modello implementato da Rothman et al. (2010) utilizzando la funzione della libreria `MRCE`. Per il *tuning* dei parametri per il modello MRCE si è seguito quanto scritto nell'analogo articolo, utilizzando quindi convalida incrociata con 10 *folds*.
Si sono poi calcolate le previsioni sulle 26 altre settimane (sull'insieme di verifica) e per confrontare i vari modelli si è calcolato la media degli errori al quadrato. Nella sezione seguente sono presentati i risultati con alcune considerazioni.

3.4 Risultati

Nelle tabelle 3.3 e 3.2 sono presentati i valori dei parametri stimati della matrice B utilizzando i metodi con $\mathbf{\Sigma}$ fissato e con $\mathbf{\Sigma}$ che si aggiorna ad ogni iterazione.

Tabella 3.2: Coefficienti stimati della matrice B utilizzando l'algoritmo ADMM Lasso con ρ fissato

	WAL	**EXX**	**GM**	**FORD**	**GE**	**CP**	**CG**	**IBM**	**AIG**
WAL	0.259	-0.447	0.000	0.404	0.265	-0.646	0.103	-0.310	0.355
EXXON	0.173	0.215	0.190	0.000	0.226	0.124	-0.177	0.000	-0.001
GM	-0.162	-0.442	0.317	0.365	-0.210	-0.219	0.000	-0.216	-0.314
FORD	0.000	0.143	0.000	0.069	-0.234	0.305	0.418	-0.087	0.116
GE	-0.323	0.329	0.559	0.340	0.000	0.330	0.318	0.410	0.036
CP	-0.325	-0.372	-0.048	-0.376	-0.296	-0.028	0.399	0.000	-0.133
CG	0.230	0.320	-0.338	-0.225	0.039	-0.043	-0.242	0.096	0.322
IBM	0.312	0.234	-0.387	0.177	0.369	0.160	-0.223	0.099	0.469
AIG	0.020	-0.169	-0.120	-0.351	-0.462	-0.237	0.000	-0.374	-0.192

Si nota che i valori utilizzado i due diversi metodi non discostano troppo. Si nota anche che l'algoritmo ha effettuato una "compressione" verso lo zero di qualche parametro, e nel caso di stima con l'aggiustamento di ρ ogni iterazione ha fatto una vera e propria selezione.
Sono state calcolate poi 26 previsioni in *rolling-window* in Figura A.9 sono rappresentate le serie storiche dei rendimenti con le previsioni delle ultime 26 settimane. Come accennato in precedenza, per valutare la bontà delle previsioni, si sono applicati anche gli altri modelli e sono stati calcolati i rispettivi errori quadratici medi nell'insieme di verifica. In Tabella 3.4 sono riportati questi valori. Nell'ultima riga

Tabella 3.3: Coefficienti stimati della matrice B utilizzando l'algoritmo ADMM Lasso con $\mathbf{\Sigma}$ aggiornato ad ogni iterazione

	WAL	**EXX**	**GM**	**FORD**	**GE**	**CP**	**CG**	**IBM**	**AIG**
WAL	0.000	0.000	0.000	0.000	0.000	-0.126	0.000	0.000	0.000
EXXON	0.186	0.000	0.000	0.000	0.000	-0.126	0.000	0.000	0.000
GM	0.000	-0.159	0.000	0.084	0.000	-0.077	0.132	0.000	-0.075
FORD	-0.057	0.000	-0.094	0.071	0.000	0.000	0.000	0.000	0.047
GE	-0.107	0.068	0.288	0.157	0.000	0.098	0.000	0.200	0.000
CP	-0.194	-0.096	0.045	0.000	-0.078	0.000	0.131	-0.117	0.000
CG	0.153	0.000	0.000	-0.131	0.000	0.000	-0.223	0.091	0.000
IBM	0.000	0.220	-0.180	0.131	0.000	0.000	-0.066	0.103	0.122
AIG	0.000	-0.080	-0.212	-0.170	0.000	-0.109	0.000	-0.144	0.000

si è riportata anche la media per colonna dei diversi errori per avere una misura complessiva di quanto si è sbagliato.

Tabella 3.4: Media degli errori quadratici medi $\times$ 1000 calcolato sulle previsioni in *rolling-window* sulle ultime 26 settimane.

	OLS	**MRCE**	**ADMM**	**ADMM adj.**
WAL	1.02	0.46	0.47	0.47
EXXON	1.12	0.95	0.95	0.95
GM	4.13	2.90	2.90	2.89
FORD	3.09	2.05	2.05	2.05
GE	0.78	0.40	0.36	0.36
Conoco Philips	1.22	0.94	0.87	0.87
Citgroup	2.13	1.38	1.37	1.37
IBM	1.15	0.82	0.83	0.83
AIG	1.56	1.51	1.52	1.52
Media	1.80	1.27	1.26	1.26

Si nota che i valori dell'errore quadratico medio sull'insieme di verfica più elevato è quello relativo alle previsioni stimate mediante OLS. Infatti, gli errori appaiono maggiori rispetto agli altri due modelli. L'errore quadratico medio relativo al modello stimato tramite ADMM presenta dei risultati simili agli errori calcolati utilizzando l'algoritmo implementato da Rothman et al. (2010), con leggero miglioramento se si considerano alcuni *stocks* e l'errore quadratico medio. Si pensa quindi che, in casi come questo dove c'è la presenza di un vasto numero di variabili

da stimare, in un caso multivariato una selezione delle variabili risulta importante così da ridurre considerevolmente la varianza del modello e di utilizzare solo le variabili più significative. Applicando quindi il modello ADMM lasso, che comprime significativamente i parametri meno significativi verso zero, e il modello MRCE che compie una vera e propria selezione dei parametri in modo simultaneo, riporta risultati migliori in termini predittivi. Inoltre, come scrivono Rothman et al. (2010), in casi in cui i dati sono correlati, una selezione di variabili in modo simultaneo come applicando i modelli ADMM lasso multivariato e MRCE risulta più attendibile rispetto ad operare in modo separato, per esempio stimando q-diverse regressioni Lasso-penalizzate per ogni colonna della matrice Y.

Conclusioni

In questo lavoro abbiamo implementato un metodo per la stima dei paramatri nel caso dei modelli lineari multivariati penalizzati, tramite algoritmo ADMM (*Alternating Direction Method of Multipliers*). Ripercorrendo quanto speigato nei capitolo precedenti, un risultato importante è che l'algoritmo ADMM si può adattare e generalizzare per la soluzione di molteplici problemi di stima. In qiesto lavoro si è mostrato come si adatta alla stima della matrice dei parametri B nel caso di un modello con risposta multivariata penalizzato. Nel primo capitolo, inoltre, si è visto come, oltre ad essere utilizzato per la stima della matrice dei parametri, si può estendere anche per una più corretta stima della funzione di varianza e covarianza.
Nel secondo capitolo si è confrontato il metodo proposto con il metodo MRCE di Rothman et al. (2010) per produrre una stima sparsa della matrice dei coefficienti di regressione multivariata B. Questi due metodi, come si è visto in precedenza, tengono esplicitamente conto della correlazione delle variabili di risposta. Si è visto, inoltre, che in termini di errore del modello, i due metodo performano meglio rispetto alla stima mediante $q-$regressioni LASSO separate, che invece ignorano la correlazione nelle risposte.
Infine, si è valutata anche la capacità predittiva in un caso reale del modello stimato utilizzando ADMM confrontandolo con il metodo MRLS e il metodo classico OLS. I risultati ottenuti hanno riportato un errore simile nei due casi che rigurando la penalizzazione dei modelli multivariati.
Oltre a dei vantaggi riguardo stime più accurate, l'algoritmo ADMM è risultato più efficiente in termini di tempi computazionale.
Si considerano ora alcune limitazioni di questo lavoto. In questo lavoro è stata considerata solo la penalizzazione L1 simultanea di B e di Σ. Visto la potenzia-

lità dell'algoritmo ADMM ad adattarsi a più tipi di problemi di ottimizzazione, si potrebbero utilizzare altre penalità che introducuno per esempio meno distorsione, come per esempio SCAD Lam and Fan (2009). Inoltre, in questo lavoro nella parte di simulazione si è considerata solo la correlazione nella parte d'errore. Una possibile estensione sarebbe quella di introdurre nel modello anche la correlazione della matrice X. Infine, non si è sfruttato del tutto le potenzialità dell'algoritmo ADMM, in questo lavoro infatti si sono cosniderati per le matrici U e Z di partenza matrici con valori tutti nulli. Uno studio futuro potrebbe studiare come migliorare la stima dei parametri e dell'arrivo di convergenza dell'algoritmo inserendo matrici U e Z specifiche per problema.

Appendice A

Risultati utili

Proposition A.0.1 (Proprietà dell'operatore vec)**.** *Sia* $\boldsymbol{A} \in \mathbb{R}^{k\times l}$, $\boldsymbol{B} \in \mathbb{R}^{l\times m}$ *e* $\boldsymbol{C} \in \mathbb{R}^{m\times n}$*, allora*

$$\begin{aligned}\mathrm{vec}(\boldsymbol{AB}) = (\boldsymbol{I}_m \otimes \boldsymbol{A})\mathrm{vec}(\boldsymbol{B}) = (\boldsymbol{B}^\top \otimes \boldsymbol{I}_k)\mathrm{vec}(\boldsymbol{A}) \in \mathbb{R}^{mk}\\ \mathrm{vec}(\boldsymbol{ABC}) = (\boldsymbol{C}^\top \otimes \boldsymbol{A})\mathrm{vec}(\boldsymbol{B}) \in \mathbb{R}^{nk}.\end{aligned}$$

Proposition A.0.2 (Proprietà dell'operatore trace)**.** *Sia* $\boldsymbol{A} \in \mathbb{R}^{p\times q}$, $\boldsymbol{B} \in \mathbb{R}^{p\times q}$, *allora*

$$\mathrm{trace}(\boldsymbol{A}^\top\boldsymbol{B}) = \mathrm{vec}(\boldsymbol{A})^\top\mathrm{vec}(\boldsymbol{B}), \qquad \mathrm{trace}(\boldsymbol{AB}^\top) = \mathrm{trace}(\boldsymbol{B}^\top\boldsymbol{A}),$$

Proposition A.0.3 (Distribuzione Gaussiana Matrix-Variate)**.** *Sia* $Y \in \mathbb{R}^{p\times q}$ *una matrice stocastica di dimensione* $(p\times q)$*. La matrice stocastica* Y *ha una distribuzione Gaussiana matrix-variate, indicata con* $Y \sim \mathcal{MN}_{p\times q}(M, \Xi, \Psi)$*, dove* $M \in \mathbb{R}^{p\times q}$ *è la matrice dei parametri di locazione, e* $\Psi \in \mathbb{S}^{p++}$, $\Xi \in \mathbb{S}^{q++}$ *sono matrici di varianze simmetriche e definite positive di dimensione* p *e* q*, rispettivamente, se la funzione di densità di probabilità di* Y *ha la forma seguente:*

$$f_Y(Y|M, \Xi, \Psi) = \frac{1}{C_N(\Psi, \Xi, p, q)} \exp\left(\frac{1}{2} trace\left(\Xi^{-1}(Y - M)^T\Psi^{-1}(Y - M)\right)\right)$$

A.1 Grafici delle simulazioni

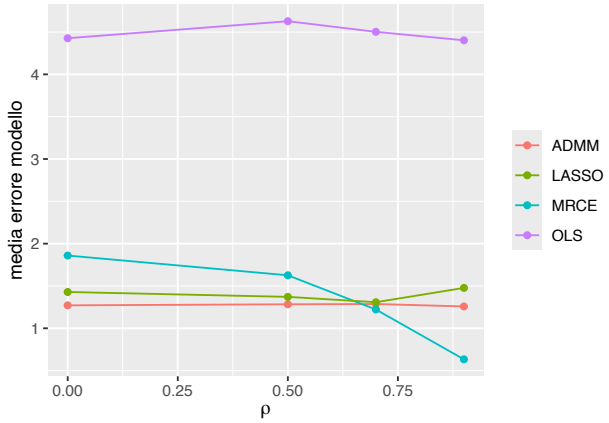

Figura A.1: Andamento della media dell'errore del modello al variare della correlazione autoregressiva di ordine uno (AR(1)) ρ_E basato su 50 replicazioni con $n = 50$, $p = q = 15$, $s_1 = 0.1$, e $s_2 = 1$

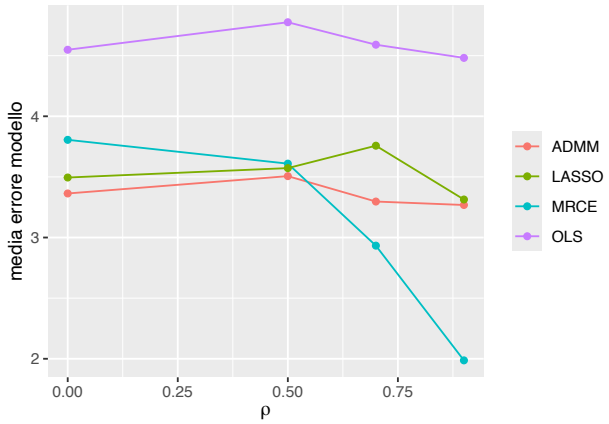

Figura A.2: Andamento della media dell'errore del modello al variare della correlazione autoregressiva di ordine uno (AR(1)) ρ_E basato su 50 replicazioni con $n = 50$, $p = q = 15$, $s_1 = 0.5$, e $s_2 = 1$

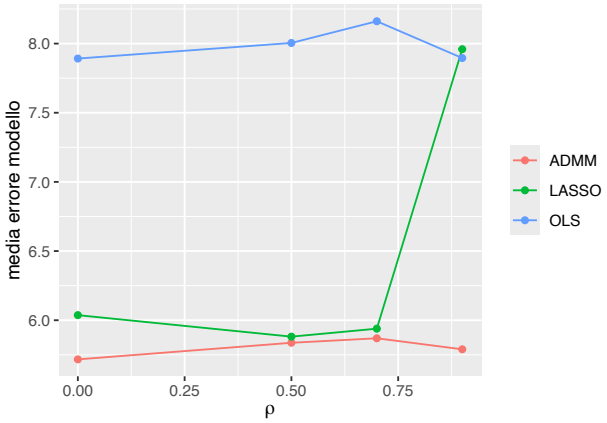

Figura A.3: Andamento della media dell'errore del modello al variare della correlazione autoregressiva di ordine uno (AR(1)) ρ_E basato su 50 replicazioni con $n = 50$, $p = q = 20$, $s_1 = 0.5$, e $s_2 = 1$

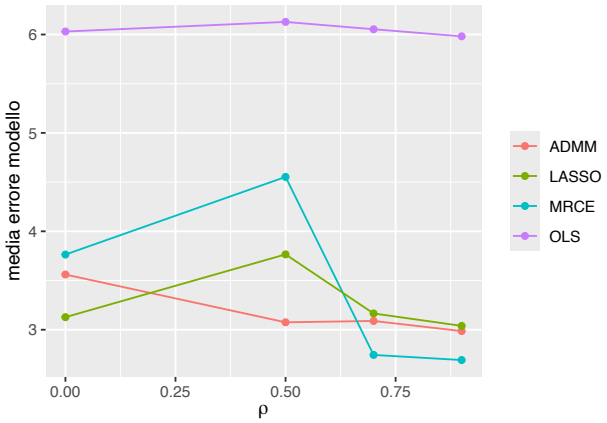

Figura A.4: Andamento della media dell'errore del modello al variare della correlazione autoregressiva di ordine uno (AR(1)) ρ_E basato su 50 replicazioni con $n = 50$, $p = 10$, $q = 20$, $s_1 = 0.5$, e $s_2 = 1$

A.2 Grafici dell'applicazione al *dataset* reale

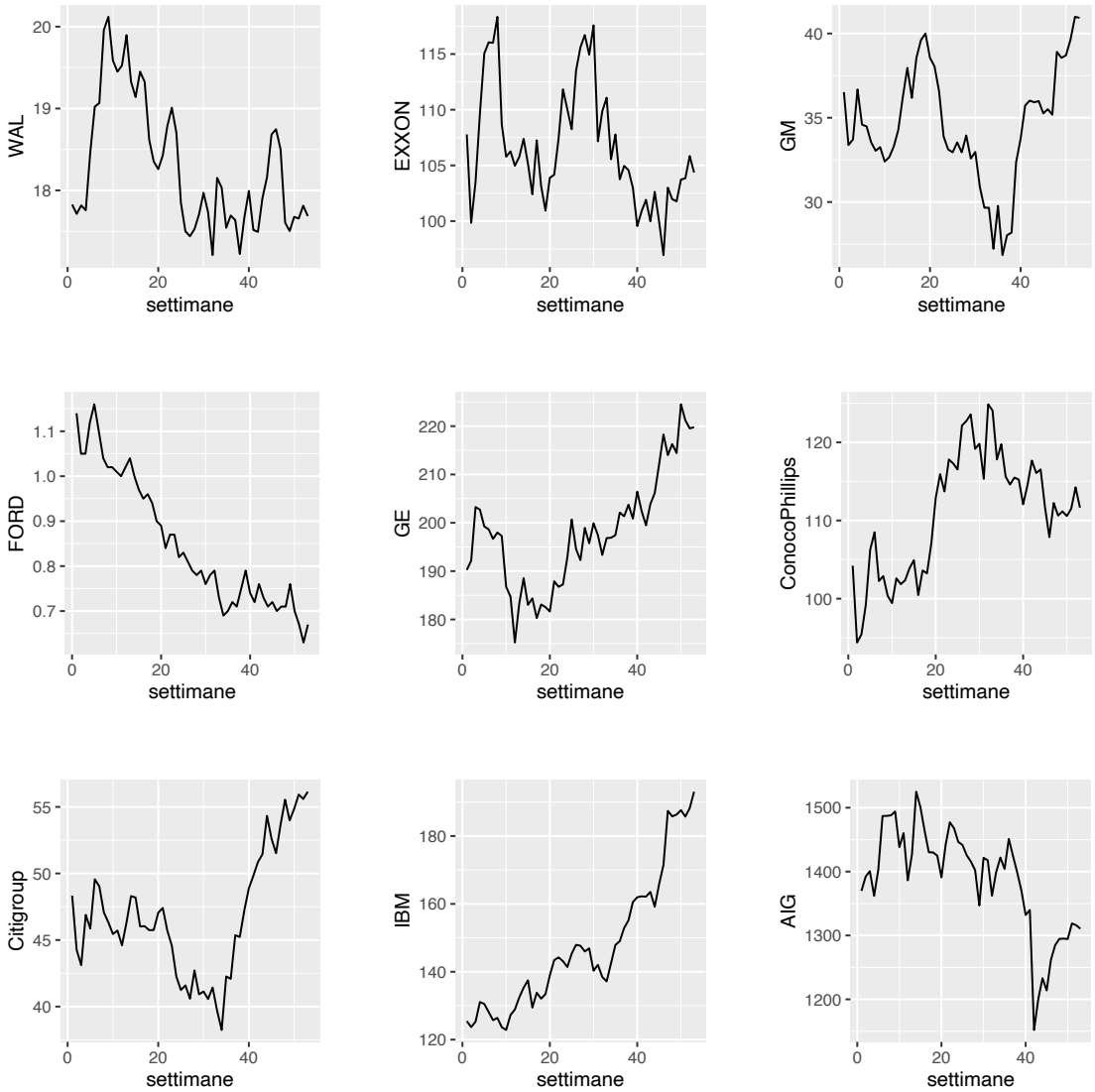

Figura A.5: Grafici delle serie storiche settimanale dei nove stocks market di riferimento

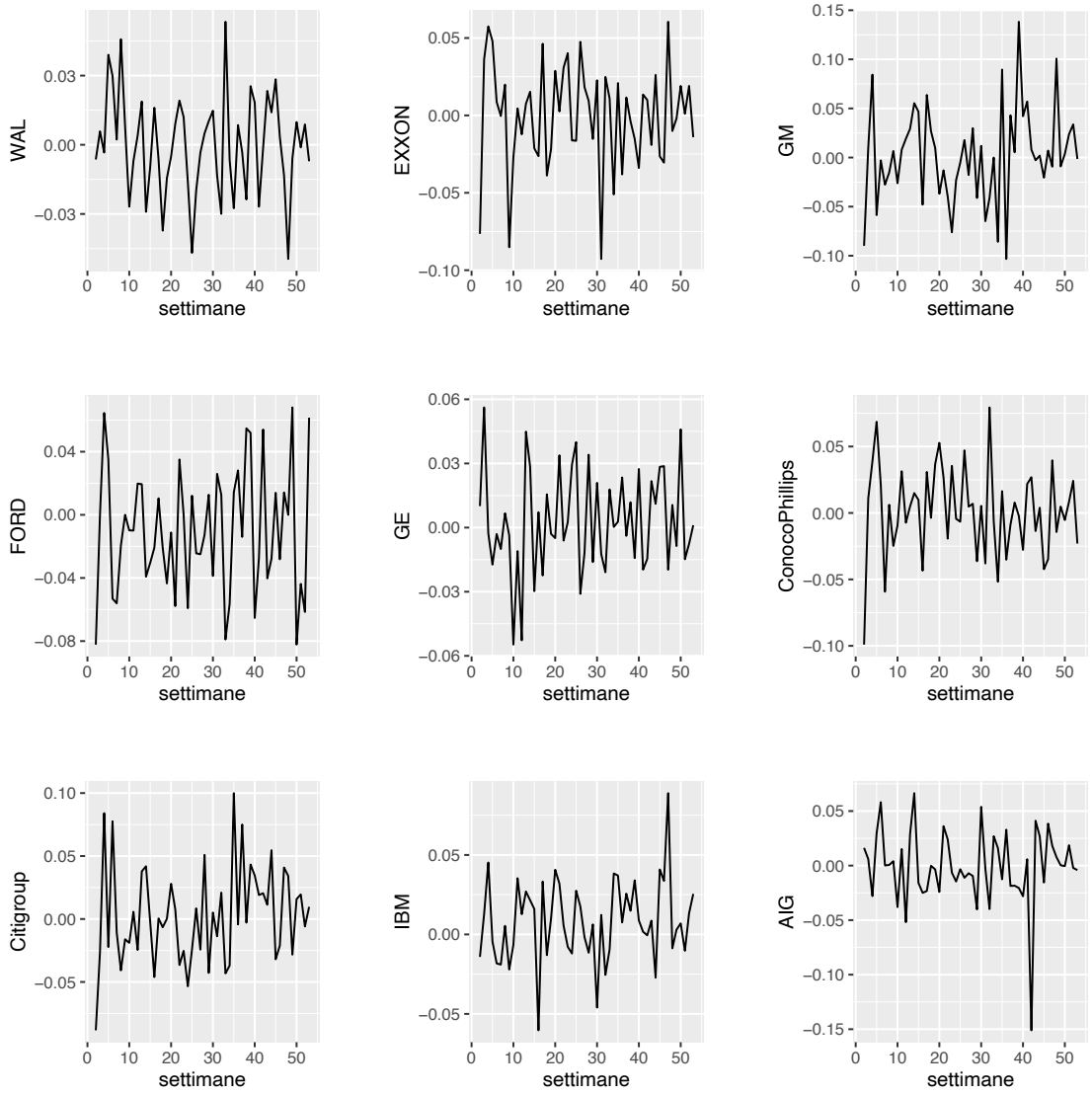

Figura A.6: Grafici delle serie storiche settimanale dei log-rendimenti settimanali degli stocks market di riferimento

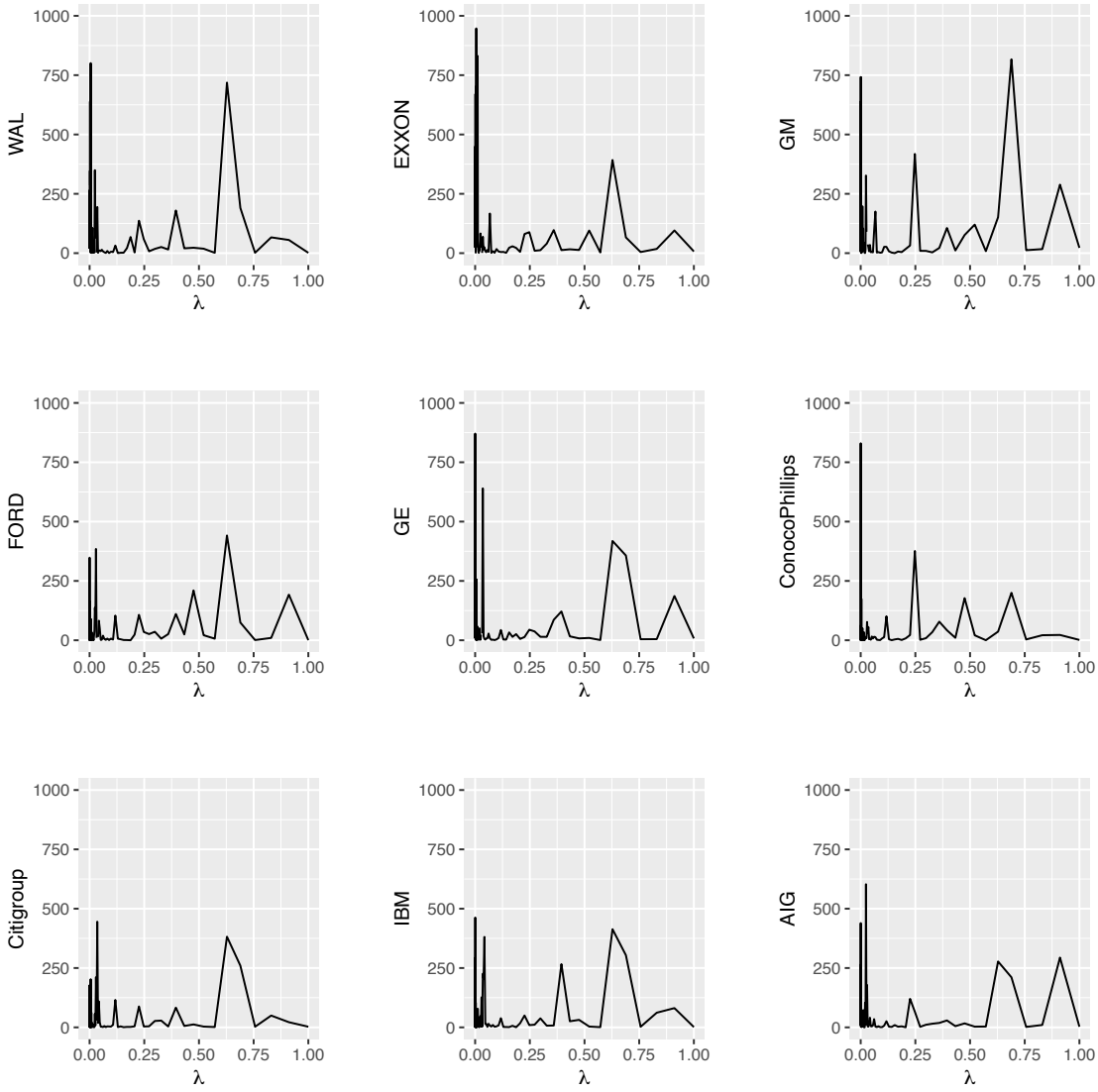

Figura A.7: Grafici relativi all'andamento dell'errore quadratico medio sull'insieme di convalida al variare del parametro di regolazione λ.

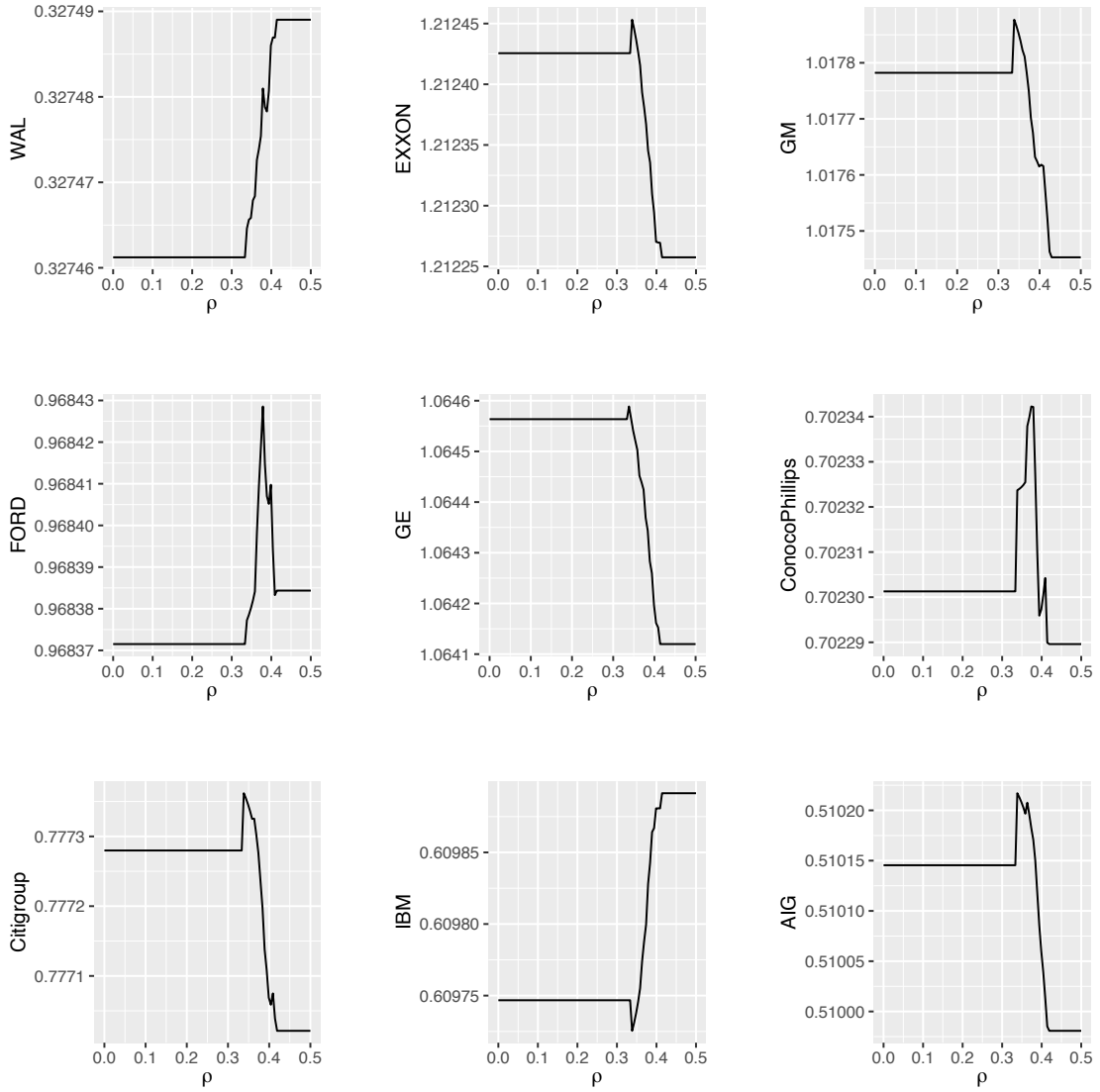

Figura A.8: Grafici relativi all'andamento dell'errore quadratico medio sull'insieme di convalida al variare del parametro di regolazione ρ nel caso della stima tramite ADMM *adjust*.

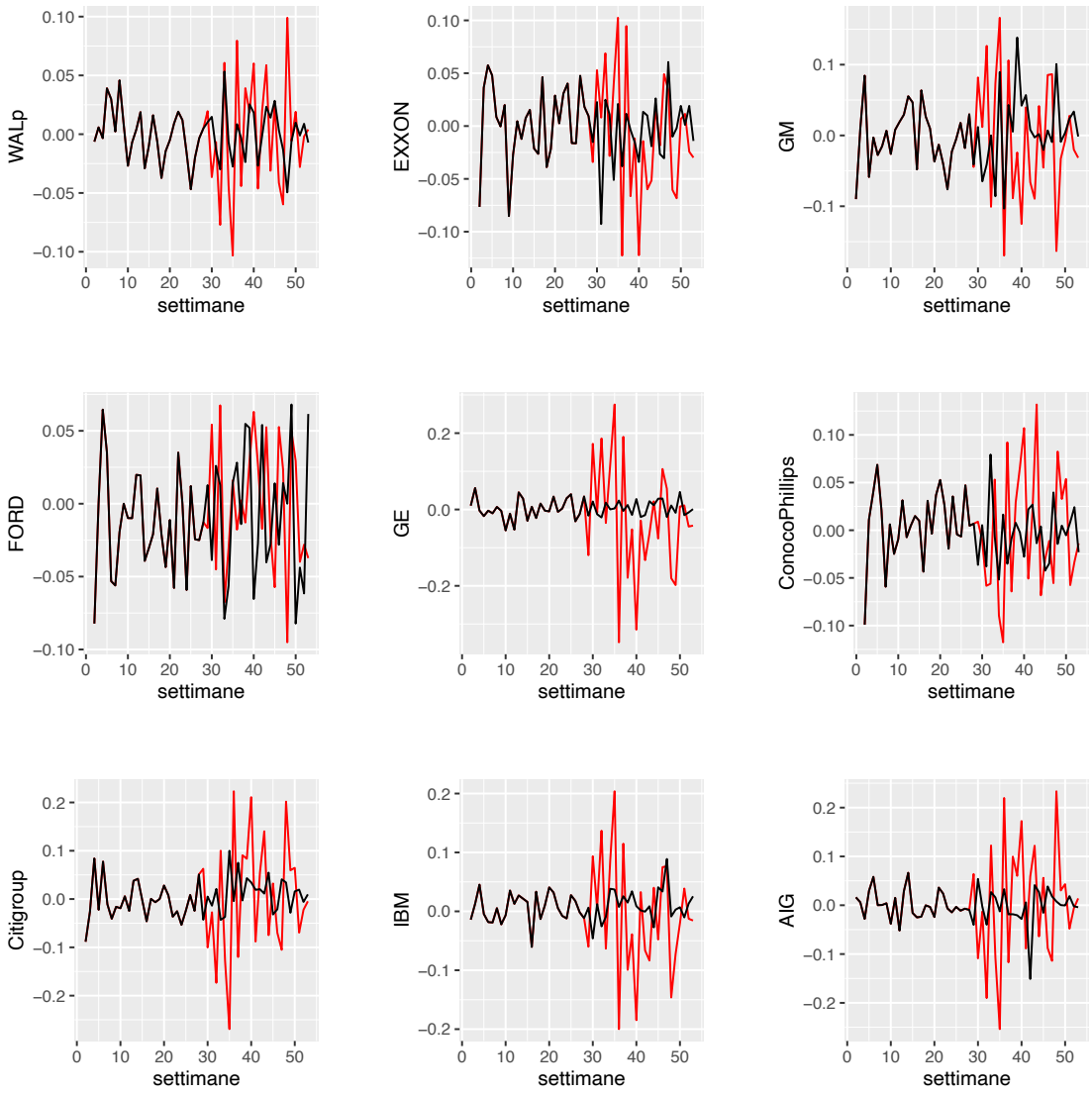

Figura A.9: In nero i grafici delle serie storiche settimanali dei nove *stocks market* di riferimento. In rosso le previsioni in rolling window delle ultime 26 settimane applicando il modello VAR(1) in forma compatta stimato tramite ADMM.

Appendice B

Codice Rcpp

Per l'applicazione del modello si è creato un pacchetto nell'ambiente R. Per poter implementarlo si è utilizzata la libreria `Armadillo`. Tramite questa libreria si sono sviluppate le varie funzioni in `c++` e poi installate in `R` sotto forma di libreria, chiamata `mrls`. In questa sezione vengono presentate le tre funzioni principali che compongono il pacchetto.

Listing B.1: Funzione ADMM lasso univariata

```
// ADMM_Lasso per modelli univariati
// [[Rcpp::export]]
Rcpp::List admm_lasso(const arma::mat& A, const arma::colvec& b,
                      arma::colvec& u, arma::colvec& z,
                      const double lambda,
                      bool rho_adaptation, double rho,
                      const double tau, const double mu,
                      const double reltol, const double abstol,
                      const int maxiter, const int ping) {

  /* Dichiarazione variabili          */
  double sqrtn, rho_old, elTime;
  int k, n1;

  /* Dimensioni              */
  const int m = A.n_rows;
```

```
17    const int n = A.n_cols;
18
19    /* Definizione variabili            */
20    sqrtn = std::sqrt(static_cast<float>(n));
21    if (m >= n) {
22      n1 = n;
23    } else {
24      n1 = m;
25    }
26
27    /* Definizioni vettori e matrici     */
28    arma::colvec x(n, fill::zeros);
29    arma::colvec q(n, fill::zeros);
30    arma::colvec z_old(n, fill::zeros);
31    arma::mat U(n1, n1, fill::zeros);
32    arma::mat L(n1, n1, fill::zeros);
33    arma::vec ATb(n, fill::zeros);
34    arma::vec h_objval(maxiter, fill::zeros);
35    arma::vec h_r_norm(maxiter, fill::zeros);
36    arma::vec h_s_norm(maxiter, fill::zeros);
37    arma::vec h_eps_pri(maxiter, fill::zeros);
38    arma::vec h_eps_dual(maxiter, fill::zeros);
39    arma::vec rho_store(maxiter+1, fill::zeros);
40    arma::mat AA(n1, n1, fill::zeros);
41
42    /* ::::::::::::::::::::::::::::::::::::::::::::
43     Set the computational time
44  */
45    wall_clock timer;
46    timer.tic();
47
48    /* store rho (only for adaptation)         */
49    rho_store(0) = rho;
50    rho_old      = rho;
51
52    /* Precompute relevant quantities         */
```

```
52    ATb = A.t() * b / static_cast<float>(m);
53    squaredmat(AA, A, m, n);
54    lasso_factor_fast(U, AA, rho, m, n); // returns upper
55    L   = U.t();
56
57    /* main loop  */
58    for (k=0; k<maxiter; k++) {
59
60      // 4-1. update 'x'
61      if (rho != rho_old) {
62        lasso_factor_fast(U, AA, rho, m, n); // returns upper
63        L = U.t();
64      }
65      q = ATb + rho * (z - u); // temporary value
66      if (m >= n) {
67        x = solve(trimatu(U), solve(trimatl(L), q));
68      } else {
69        //x = q / rho - (A.t() * solve(trimatu(U),
70        solve(trimatl(L), A * q))) / rho2;
71        x = q / rho - (A.t() * solve(trimatu(U),
72                       solve(trimatl(L), A * q))) / rho;
73      }
74
75      // 4-2. update 'z' with adaptation
76      // see Boyd et al (2011) page 21
77      z_old = z;
78      z     = lasso_prox(x + u, lambda / rho);
79
80      // 4-3. update 'u'
81      u = u + x - z;
82
83      // 4-3. dianostics, reporting
84      h_objval(k) = lasso_objfun(A, b, lambda, x, z);
85      h_r_norm(k) = norm(x - z, 2);
86      h_s_norm(k) = norm(-rho * (z - z_old), 2);
87      if (norm(x, 2) > norm(-z, 2)) {
```

```
88        h_eps_pri(k) = sqrtn * abstol + reltol * norm(x, 2);
89      } else {
90        h_eps_pri(k) = sqrtn * abstol + reltol * norm(-z, 2);
91      }
92      h_eps_dual(k) = sqrtn * abstol + reltol * norm(rho * u, 2);
93
94      /* ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: */
95      /* Relaxation
*/
96      rho_old = rho;
97      if (rho_adaptation) {
98        if (h_r_norm(k) > mu * h_s_norm(k)) {
99          rho = rho * tau;
100         u   = u / tau;
101       } else if (h_s_norm(k) > mu * h_r_norm(k)) {
102         rho = rho / tau;
103         u   = u * tau;
104       }
105     }
106     rho_store(k+1) = rho;
107
108     /* ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: */
109     /* Print to screen
*/
110     if (ping > 0) {
111       if ((ping > 0) && (k < maxiter)) {
112         if (rho_adaptation) {
113           if (((k+1) % ping) == 0) {
114             Rcpp::Rcout << "\n\n\n" << std::endl;
115             Rcpp::Rcout << "ADMM␣adaption␣running" << std::endl;
116             Rcpp::Rcout << "Iteration␣n.:␣" << k+1 << "of"
117             << maxiter << "\n" << std::endl;
118           }
119         } else {
120           if (((k+1) % ping) == 0) {
121             Rcpp::Rcout << "\n\n\n" << std::endl;
```

```
122             Rcpp::Rcout << "ADMM␣for␣LASSO␣is␣running!" << std::endl;
123             Rcpp::Rcout << "Iteration␣n.:␣" << k+1 << "␣of␣"
124             << maxiter << "\n" << std::endl;
125           }
126         }
127       }
128     }
129
130     // 4-4. termination
131     if ((h_r_norm(k) < h_eps_pri(k)) && (h_s_norm(k) < h_eps_dual(k))) {
132       break;
133     }
134   }
135
136   /* ::::::::::::::::::::::::::::::::::::::::::::::::
137    Compute elapsed time
*/
138   elTime = timer.toc();
139
140   /* Get output          */
141   List output;
142   output["x"] = x;                // coefficient function
143   output["u"] = u;
144   output["z"] = z;
145   if (k < maxiter) {
146     output["convergence"] = 1.0;
147     output["niter"]       = k+1;              // number of iterations
148     output["objval"]      = h_objval.subvec(0, k);
// |x|_1
149     output["r_norm"]      = h_r_norm.subvec(0, k);
150     output["s_norm"]      = h_s_norm.subvec(0, k);
151     output["eps_pri"]     = h_eps_pri.subvec(0, k);
152     output["eps_dual"]    = h_eps_dual.subvec(0, k);
153     output["rho"]         = rho_store.subvec(0, k+1);
154   } else {
155     output["convergence"] = 0.0;
```

```
156     output["niter"]       = maxiter;
// number of iterations
157     output["objval"]      = h_objval;        // |x|_1
158     output["r_norm"]      = h_r_norm;
159     output["s_norm"]      = h_s_norm;
160     output["eps_pri"]     = h_eps_pri;
161     output["eps_dual"]    = h_eps_dual;
162     output["rho"]         = rho_store;
163   }
164   output["eltime"]   = elTime;
165   if ((k+1) < maxiter) {
166     output["convergence"] = 1.0;
167   } else {
168     output["convergence"] = 0.0;
169   }
170
171   /* Return output      */
172   return(output);
173 }
```

Listing B.2: Funzione ADMM lasso multivariata senza l'aggiornamento della varianza ad ogni step

```
1
2  // Implementazione admm_lasso con matrice di varianza e covarianza I
3  // [[Rcpp::export]]
4  Rcpp::List admm_lasso_m(const arma::mat& Ax, const arma::mat& Bx,
5                          arma::mat& Ux, arma::mat& Zx,
6                          const double lambda,
7                          bool rho_adaptation, double rho,
8                          const double tau, const double mu,
9                          const double reltol, const double abstol,
10                         const int maxiter, const int ping){
11
12     int n, q, p;
13
14     /* Definizione delle matrici e vettori */
```

```
15      n = Ax.n_rows;
16      q = Bx.n_cols;
17      p = Ax.n_cols;
18      arma::vec eyeQ(q, fill::ones);
19      arma::mat Iq = diagmat(eyeQ);
20
21      arma::mat A = kron(Iq, Ax);
22      arma::vec b = vecC(Bx);
23      arma::vec u = vecC(Ux);
24      arma::vec z = vecC(Zx);
25
26      Rcpp::List risultato = admm_lasso(A, b, u, z, lambda,
27      rho_adaptation, rho, tau, mu, reltol, abstol, maxiter, ping);
28
29      arma::mat par = invvecC(risultato["x"], p);
30      Ux = invvecC(risultato["u"], p);
31      Zx = invvecC(risultato["z"], p);
32
33      arma::mat res = (Bx - Ax*par);
34      arma::mat Sig_hat = res.t() * res / n;
35      int n_iter = risultato["niter"];
36
37      /* Get output          */
38      List output;
39      output["X"]         = par;                // matrice dei parametri
40      output["U"]         = Ux;
41      output["Z"]         = Zx;
42      output["Sig"]       = Sig_hat;
43      output["n_iter"]    = n_iter;
44
45      /* Return output       */
46      return(output);
47
48  }
```

Listing B.3: Funzione ADMM lasso multivariata con l'aggiornamento della varianza ad ogni step

```
1
2  // ADMM_Lasso per modelli multivariati
3  // [[Rcpp::export]]
4  Rcpp::List admm_mult(const arma::mat& A, const arma::mat& B,
5                       arma::mat& U, arma::mat& Z,
6                       const double lambda,
7                       bool rho_adaptation, double rho, const
8                       double tau, const double mu,
9                       const double reltol, const double abstol,
10                      const int maxiter, const int ping) {
11
12     /* dichiarazione delle variabili */
13     double sqrtn, rho_old, elTime;
14     int k, n1;
15
16     /* dimensioni */
17     const int m = A.n_rows; // n
18     const int n = A.n_cols; // p
19     const int Bcols = B.n_cols; // q
20
21     /* Definizioni vettori e matrici     */
22     sqrtn = std::sqrt(static_cast<float>(n));
23
24     arma::vec b(m*Bcols, fill::zeros);
25     arma::mat Ax(m*Bcols, n*Bcols, fill::zeros);
26
27     arma::mat X(n, Bcols, fill::zeros);
28     arma::colvec z_old(n*Bcols, fill::zeros);
29
30     arma::vec x(n*Bcols, fill::zeros);
31     arma::vec u(n*Bcols, fill::zeros);
32     arma::vec z(n*Bcols, fill::zeros);
33
34     arma::vec ATB(n*Bcols, fill::zeros);
```

```
35      arma::mat AA(n*Bcols, n*Bcols, fill::zeros);
36      arma::mat Uq(n*Bcols, n*Bcols, fill::zeros);
37      arma::mat L(n*Bcols, n*Bcols, fill::zeros);
38      arma::vec q(n*Bcols, fill::zeros);
39
40      arma::vec eyeBcols(Bcols, fill::ones);
41      arma::mat Sig = diagmat(eyeBcols);
42      arma::mat IBcols = diagmat(eyeBcols);
43
44      arma::vec eyeN(n, fill::ones);
45      arma::vec eyeM(m, fill::ones);
46
47      arma::mat V_lambda(n, n, fill::zeros);
48
49      arma::vec h_objval(maxiter, fill::zeros);
50      arma::vec h_r_norm(maxiter, fill::zeros);
51      arma::vec h_s_norm(maxiter, fill::zeros);
52      arma::vec h_eps_pri(maxiter, fill::zeros);
53      arma::vec h_eps_dual(maxiter, fill::zeros);
54      arma::vec rho_store(maxiter+1, fill::zeros);
55
56      /* Inizio Tempo Computazionale */
57      wall_clock timer;
58      timer.tic();
59
60      /* salvataggio rho solo per adaption */
61      rho_store(0) = rho;
62      rho_old      = rho;
63
64      /* calcoli importanti per il pre */
65      // usare A e b modificati
66      b = vecC(B);
67      Ax = kron(IBcols, A);
68      // vettorizzo quantit
69      z = vecC(Z);
70      u = vecC(U);
```

```
71
72      ATB = kron(IBcols, A.t()) * kron(diagmat(eyeM), inv(Sig)) *
73      b / static_cast<float>(m);
74      AA = squaredmat_m(A, Sig, n, Bcols);
75
76      lasso_factor_fast_mult(Uq, AA, rho, n, m, Bcols);
77      L = Uq.t();
78
79      /* loop principale */
80      for (k=0; k<maxiter; k++) {
81
82          ATB = kron(IBcols, A.t()) * kron(diagmat(eyeM), inv(Sig))
83          * b / static_cast<float>(m);
84          AA = squaredmat_m(A, Sig, n, Bcols);
85
86          // aggiornamento di x
87          if (rho != rho_old) {
88              lasso_factor_fast_mult(Uq, AA, rho, n, m, Bcols);
89              L = Uq.t();
90
91          }
92          q = ATB + rho * (z - u);
93          x = solve(trimatu(Uq), solve(trimatl(L), q)); // problemi
94          X = invvecC(x, n);
95
96          // aggiornamento di z
97          z_old = z;
98          z     = lasso_prox(x + u, lambda / rho);
99
100         // aggiornamento di u
101         u = u + x - z;
102
103         // ottengo Z e U
104         Z = invvecC(z, n);
105         U = invvecC(u, n);
106
```

```
107         // aggiornamento di Sigma
108         Sig = get_sigma_new(A, B, X);
109
110         // diagnostica
111         h_objval(k) = lasso_objfun(Ax, b, lambda, x, z);
112         h_r_norm(k) = norm(x - z, 2);
113         h_s_norm(k) = norm(-rho * (z - z_old), 2);
114
115         if (norm(x, 2) > norm(-z, 2)) {
116           h_eps_pri(k) = sqrtn * abstol + reltol * norm(x, 2);
117         } else {
118           h_eps_pri(k) = sqrtn * abstol + reltol * norm(-z, 2);
119         }
120         h_eps_dual(k) = sqrtn * abstol + reltol * norm(rho * u, 2);
121
122         // relaxation
123         rho_old = rho;
124         if (rho_adaptation) {
125           if (h_r_norm(k) > mu * h_s_norm(k)) {
126             rho = rho * tau;
127             u   = u / tau;
128           } else if (h_s_norm(k) > mu * h_r_norm(k)) {
129             rho = rho / tau;
130             u   = u * tau;
131           }
132         }
133         rho_store(k+1) = rho;
134
135         // presa visione
136         if (ping > 0) {
137           if ((ping > 0) && (k < maxiter)) {
138             if (rho_adaptation) {
139               if (((k+1) % ping) == 0) {
140                 Rcpp::Rcout << "\n\n\n" << std::endl;
141                 Rcpp::Rcout << "ADMM␣with␣adaptation␣for␣LASSO␣is␣running!"
142                 << std::endl;
```

```
143                 Rcpp::Rcout << "Iteration n.: " << k+1 << " of "
144                 << maxiter << "\n" << std::endl;
145               }
146             } else {
147               if (((k+1) % ping) == 0) {
148                 Rcpp::Rcout << "\n\n\n" << std::endl;
149                 Rcpp::Rcout << "ADMM for LASSO is running!"
150                 << std::endl;
151                 Rcpp::Rcout << "Iteration n.: " << k+1 << " of "
152                 << maxiter << "\n" << std::endl;
153               }
154             }
155           }
156         }
157 
158         // criteri di stop
159         if ((h_r_norm(k) < h_eps_pri(k)) &&
160         (h_s_norm(k) < h_eps_dual(k))) {
161           break;
162         }
163 
164 
165       }
166 
167       /* Fine Tempo Computazionale */
168       elTime = timer.toc();
169 
170       // Output
171       List output;
172       output["X"] = X;          // matrice coefficienti
173       output["U"] = U;
174       output["Z"] = Z;
175       output["Sig"] = Sig;
176       if (k < maxiter) {
177         output["convergence"] = 1.0;
178         output["niter"]       = k+1;      // numero di iterazioni
```

```
179         output["objval"]       = h_objval.subvec(0, k);
// |x|_1
180         output["r_norm"]       = h_r_norm.subvec(0, k);
181         output["s_norm"]       = h_s_norm.subvec(0, k);
182         output["eps_pri"]      = h_eps_pri.subvec(0, k);
183         output["eps_dual"]     = h_eps_dual.subvec(0, k);
184         output["rho"]          = rho_store.subvec(0, k+1);
185       } else {
186         output["convergence"] = 0.0;
187         output["niter"]        = maxiter;      // numero di iterazioni
188         output["objval"]       = h_objval;         // |x|_1
189         output["r_norm"]       = h_r_norm;
190         output["s_norm"]       = h_s_norm;
191         output["eps_pri"]      = h_eps_pri;
192         output["eps_dual"]     = h_eps_dual;
193         output["rho"]          = rho_store;
194       }
195       output["eltime"]    = elTime;
196       if ((k+1) < maxiter) {
197         output["convergence"] = 1.0;
198       } else {
199         output["convergence"] = 0.0;
200       }
201
202       /* Return output       */
203       return(output);
204
205
206 }
```

Bibliografia

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011a). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122.

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al. (2011b). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122.

Fang, Q., Yu, C., and Weiping, Z. (2020). Regularized estimation of precision matrix for high-dimensional multivariate longitudinal data. *Journal of Multivariate Analysis*, 176:104580.

Gabay, D. and Mercier, B. (1976). A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1):17–40.

Glowinski, R. and Marroco, A. (1975). Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires. *R.A.I.R.O. Analyse Numérique*, 9(2):41–76.

Gorski, J., Pfeuffer, F., and Klamroth, K. (2007). Biconvex sets and optimization with biconvex functions: a survey and extensions. *Mathematical methods of operations research*, 66:373–407.

Hastie, T., Tibshirani, R., and Wainwright, M. (2015). *Statistical learning with sparsity: the lasso and generalizations*. CRC press.

Lam, C. and Fan, J. (2009). Sparsistency and rates of convergence in large covariance matrix estimation. *Annals of statistics*, 37(6B):4254.

Peng, J., Zhu, J., Bergamaschi, A., Han, W., Noh, D.-Y., Pollack, J. R., and Wang, P. (2010). Regularized multivariate regression for identifying master predictors with application to integrative genomics study of breast cancer. *The annals of applied statistics*, 4(1):53.

Pesaran, H. H. and Shin, Y. (1998). Generalized impulse response analysis in linear multivariate models. *Economics letters*, 58(1):17–29.

Rothman, A. J., Levina, E., and Zhu, J. (2010). Sparse multivariate regression with covariance estimation. *Journal of Computational and Graphical Statistics*, 19(4):947–962.

Yin, J. and Li, H. (2012). Model selection and estimation in the matrix normal graphical model. *Journal of multivariate analysis*, 107:119–140.

Yuan, M., Ekici, A., Lu, Z., and Monteiro, R. (2007). Dimension reduction and coefficient estimation in multivariate linear regression. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 69(3):329–346.

Yuan, M. and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 68(1):49–67.