Uniwersytet Wrocławski
Wydział Matematyki i Informatyki
Data Science

Szymon Czop

Estimation of Low Rank and Sparse Covariance Matrices

Estymacja rzadkich macierzy kowariancji niskiego rzędu

Praca magisterska
napisana pod kierunkiem
dr. Damiana Brzyskiego

Wrocław, 2021

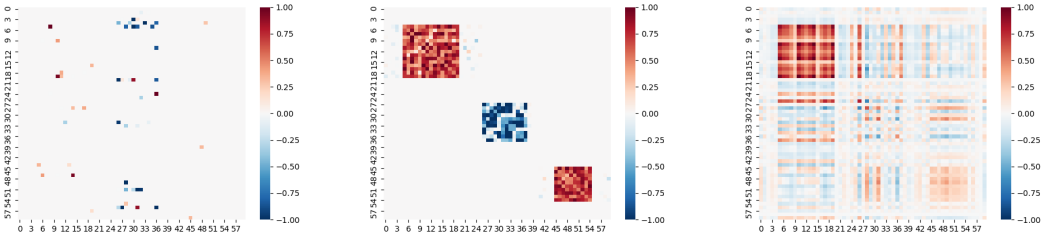
Contents

1 Streszczenie

W poniższej pracy przedstawiamy nowy algorytm, służący do estymacji rzadkich macierzy niskiego rzędu, wykorzystany w kontekście odzyskiwania macierzy kowariancji, na podstawie wygenerowanych danych. Algorytm korzysta z normy Frobeniusa, której zadaniem jest dopasowanie do empirycznej macierzy kowariancji, normy nuklearnej, która zapewnia niski rząd oraz normy l_1, która sprawa, że nieistotne korelacje są sprowadzane do zera. Kolejnym krokiem jest zbadanie matematycznych własności wyżej przedstawionego zagadnienia. Badamy tam między innymi cechy takie jak jednoznaczność, unikalności czy symetryczność optymalnego rozwiązania. Metoda numeryczne korzysta z algorytmu ADMM, który jest rozwiązany iteracyjnie, z podziałem na podproblemy, których rozwiązanie przedstawione jest w dalszej części pracy. Na koniec, dzięki implementacji algorytmu w języku Python, przeprowadziliśmy szczegółowe eksperymenty mające na celu porównania jakości nowego estymatora oraz jego klasycznego odpowiednika.

2 Introduction

Estimation of the correlation matrix is a common problem in data analysis. Even a very simple approximation of correlation gives us very important insight into how features in data are dependent on each other. This information can be used to create a proper statistical model if needed or exclude not important features. We can also explore the valid interaction between features that can help to understand the nature of the phenomenon that we observe through data.
In this paper, we will focus on the recovery of covariance matrices that are **sparse** and have **low rank**. Sparsity provides us with the identification of the strongest signals that can be found between features that correlation is highest and low-rank help us to recover clusters where groups of variables interact with each other. The signal of correlation that we estimate should be strong within these clusters and relatively weak between them. Covariance matrices that are following those rules are well known from the problem of genome-wide association study (GWA) or whole-genome association study (WGA) where genetics research is used to associate specific genetic variations with particular diseases [Emi21], [RNy10] [Llo18]. Matrix of this property is also the solution to the problem introduced in [Brz20] where the author was solving the problem of brain-connected regions and information they carried about HIV stage. To better understand such approximation consider Figure 1. The first plot on the left panel is sparse but at the same time full rank. The one on the right side on the other hand has a low rank but they are not sparse. The plot in the middle is both spare and low rank and is the structure of our interest.



(a) Sparse, full-rank (b) Low-rank and spars (c) Low-rank, not sparse

Figure 1: Illustration of the matrix that is sparse and full rank (a), (c) low-rank matrix, but not sparse, (b) low rank and sparse matrix

Estimation problem of such data was previously considered in various scientific works [RO10], [Wan14], [Zho14], [GC14]. They all failed to find low rank and sparse estimation. The method which recovers the sparse and low-rank estimate under matrix

regression problem was introduced in [Brz20]. The authors of this work were the first to use a combination of a nuclear-norm penalty and l_1 penalty, simultaneously imposed on the estimator. Both penalties are extremely important in terms of the new algorithm introduced in this paper, thus their work can be considered as the starting point of the following research.
Although the solution to the problem introduced in [Brz20] have same properties as we are looking for, the problem statement itself is different and requires different optimization process. The method called **Low-Rank Sparse Estimator** (LoRSEr) will be introduced in the following pages. We will investigate its properties, conduct simulation experiments and compare its performance to the classical widely used estimator of the covariance matrix. It is also worth mentioning that the main task of LoRSEr is to recover clusters with the strongest signal rather than the exact values of correlation.
Whole python code created for this research is open source and can be found in repository https://github.com/szymonczop/CovMatrixEstimator.

2.1 Research carried out during thesis creation

The following shortlist introduces the most important parts of the following thesis that were explored during its creation.

1. Introducing new optimization problem (LoRSEr)
2. Proving theoretical properties of the LoRSEr
3. Implementing numerical solution of LoRSEr in Python
4. Comparing performance of LoRSEr and classical covariance estimation in series of experiments.

3 Preliminaries

We will start from introducing some assumptions and definitions.

Definition 3.1. *We define* ***sample covariance matrix Y*** *as*

$$Y := X^T X, \tag{1}$$

where X is experiment matrix with columns standardized to zero mean values.

Definition 3.2. *A function $f : V \to \mathbb{R}$ in a vector space V is convex if*

$$f\Big(tx_1 + (1-t)x_2\Big) \leq tf(x_1) + (1-t)f(x_2),$$

for all $x_1, x_2 \in V$ and $t \in (0,1)$. If function f satisfies the above condition with strict inequality for all $x_1, x_2 \in V, x_1 \neq x_2$, then f is a ***strictly convex*** *function on vector space V.*

Definition 3.3. *We define* ***Frobenius inner product*** *between two matrices A and B with the same sizes as*

$$\langle A, B \rangle_F := \sum_{i=1}^{m} \sum_{j=1}^{n} A_{i,j} B_{i,j}. \tag{2}$$

Definition 3.4. *The Frobenius* ***matrix norm*** *of $m \times n$ matrix X is defined as*

$$||X||_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} x_{i,j}^2} = \sqrt{\langle X, X \rangle_F} = \sqrt{tr(XX^T)}. \tag{3}$$

Definition 3.5. *The* ***Hadamard*** *product between two matrices A and B with the same sizes is $m \times n$ matrix, $(A \circ B)$, defined as*

$$(A \circ B)_{i,j} := (A)_{i,j}(B)_{i,j}. \tag{4}$$

For matrices with different dimensions, the Handamard product is not defined.

Definition 3.6. *Singular Value Decomposition (****SVD****) for rectangular matrix A is the factorization of A into the product of three matrices:*

$$A = U\Sigma V^T, \tag{5}$$

where the matrices U and V are orthonormal and the Σ is diagonal matrix with nonnegative entries on diagonal arranged in non-increasing order. Columns of U and V are made of left and right singular vectors of A accordingly. The entries on diagonal of Σ are also called singular values.

Definition 3.7. *The nuclear norm of $m \times n$ matrix A is defined as*

$$||A||_* = \sum_{i=1}^{min\{m,n\}} \sigma_i(A), \tag{6}$$

where σ_i is i-th singular value of A.

Definition 3.8. *Given the permutation π of p elements $\pi : \{1, \ldots, p\} \to \{1, \ldots, p\}$. We define $p \times p$ **permutation matrix** P_π as identity matrix I_p that rows where permuted according to π formula. Every row has exactly single 1 and 0 elsewhere.*

$$P_\pi = \begin{bmatrix} e_{\pi(1)} \\ e_{\pi(2)} \\ e_{\pi(3)} \\ \vdots \\ e_{\pi(p)} \end{bmatrix} \text{ and } P_\pi v = \begin{bmatrix} v_{\pi(1)} \\ v_{\pi(2)} \\ v_{\pi(3)} \\ \vdots \\ v_{\pi(p)} \end{bmatrix}$$

for any column vector v. Where $e_{\pi(i)}$ is a vector with value 1 in place defended by permutation $\pi_{(i)}$ and zeros elsewhere.

Proposition 3.9. *For any permutation matrix defined in* (3.8) *it is true that $P_\pi P_\pi^T = I$ and $P_\pi^{-1} = P_{\pi^{-1}} = P_\pi^T$.*

Proposition 3.10. *For any matrix M operation $P_\pi M$ permutes its rows and $M P_\pi^T$ permute its columns.*

4 Introducing LoRSEr

4.1 Statistical model

Finding the correlated variables is one of the important problems occurring in data analysis. The classical estimator that is computed for this purpose is the covariance matrix defined as $Y = X^T X$ of size $p \times p$ where p indicates the number of features (columns) in experiment matrix X after standardisation. When we work with data

that covariance matrix is sparse, the classical estimation will give us a lot of false positives. Additionally when we have some a prior knowledge about the data we are analyzing, there is no way to include that source of information into estimation process of Y. Sometimes we suppose that only a few variables are correlated and what is more, they can be grouped into separate clusters. To solve this problem we are introducing **LoRSEr** which enables us to recover a more accurate covariance matrix under some assumptions. First assumption is that expected covariance matrix is **sparse**. It means that there is a lot of features in matrix X which are **not** correlated thus on the intersection of their indices in the estimated matrix we expect to find zero. By doing this we recover only the strongest connections. The second assumption is the existence of clusters of strongly correlated features, which corresponds to the existence of corresponding blocks in the correlation matrix. If the features are arranged in the "cluster-by-cluster" order, the corresponding matrix has block-diagonal structure. The indices of features building the clusters are not known a-priori but even for the origin features order such structure of correlation matrix is well reflected in its low-rank approximation which builds the motivation for the method introduced in this dissertation.

4.2 Optimisation problem

In order to get sparse and low-rank regression coefficient we have to introduce some constraints: l_1 norm will enforce entry-wise sparsity and **nuclear** norm give a convex relaxation of rank minimization [Bas18].

For the predefined pair of tuning nonnegative parameters, λ_N and λ_L, LoRSEr is defined as

$$\hat{B} := \underset{B}{\operatorname{argmin}} \left\{ \frac{1}{2}||Y - B||_F^2 + \lambda_N \left\|B\right\|_* + \lambda_L \left\| \operatorname{vec}(W \circ B) \right\|_1 \right\}, \tag{7}$$

where W is a symmetric matrix with non-negative entries and size same as Y. On the default matrix, W has zeros on the diagonal and ones elsewhere. Zeros assure that the diagonal entries of $\hat{B}$ will not be shrunk to zero by the l_1 norm which otherwise would make the recovery of the low-rank matrix more challenging. It is also worth mentioning that function $B \mapsto ||vec(W \circ B)||_1$ is not a norm if W contains zeros but it is always a convex function of B.

4.3 Properties

In this subsection, we will investigate the most important properties of LoRSEr (7), including the uniqueness of solution, its symmetry and invariability under reordering applied to rows and columns of Y matrix.

For the sake of simplicity, we will split the LoRSEr (7) objective function into three components which we denote as $f(b), g(b), h(b)$:

$$F(B) := \underbrace{\frac{1}{2}||Y - B||_F^2}_{f(B)} + \underbrace{\lambda_N \big\|B\big\|_*}_{g(B)} + \underbrace{\lambda_L \big\| \operatorname{vec}(W \circ B)\big\|_1}_{h(B)}. \tag{8}$$

Proposition 4.1. *For any parameters $0 \leq \lambda_N$ and $0 \leq \lambda_L$ there exists exactly one solution to the problem (7) .*

Proof. The first step is to show that problem (7) always has the solution. To start with, observe that if $||B||_F$ follows to infinity, then $||Y - B||_F$ follows to infinity as well. Indeed, we have:

$$\underbrace{||B||_F}_{\mapsto \infty} = ||Y + B - Y||_F \leq ||Y - B||_F + \underbrace{||Y||_F}_{const} \mapsto \infty.$$

Consequently we get:

$$||B||_F \mapsto \infty \Rightarrow F(B) \mapsto \infty. \tag{9}$$

Put $C := F(0)$. From (9) we know that there exists R such that $F(B) > C$ for all B from the outside of the ball of radius R centered in zero. Precisely,

$$\exists R > 0 : ||B||_F > R \Rightarrow F(B) > C.$$

This leads to following formula:

$$B^* \in \underset{B}{\operatorname{argmin}}\, F(B) \iff B^* \in \begin{cases} \underset{B}{\operatorname{argmin}}\, F(B) \\ \text{s.t } ||B||_F \leq R. \end{cases}$$

The set $||B||_F <= R$ is compact and the function F is continuous. Hence, from the **Weierstrass** theorem, we get the F attains the minimum and argmin $F(B)$ is not empty.

Next step is to show that $||Y - B||_F^2$ **is a strictly convex function**.
Firstly we will show that $K(B) := ||B||_F^2$ is strictly convex. For any x_1 and x_2 where $x_1 \neq x_2$:

$$0 < ||x_1 - x_2||_F^2$$
$$2\langle x_1, x_2\rangle < ||x_1||_F^2 + ||x_2||_F^2$$
$$2a(1-a)\langle x_1, x_2\rangle < a(1-a)||x_1||_F^2 + a(1-a)||x_2||_F^2$$
$$-a(1-a)||x_1||_F^2 - a(1-a)||x_2||_F^2 + 2a(1-a)\langle x_1, x_2\rangle < 0$$
$$a(a-1)||x_1||_F^2 + (1-a)\left[(1-a)-1\right]||x_2||_F^2 + 2a(1-a)\langle x_1, x_2\rangle < 0$$
$$a^2||x_1||_F^2 - a||x_1||_F^2 + (1-a)^2||x_2||_F^2 - (1-a)||x_2||_F^2 + 2a(1-a)\langle x_1, x_2\rangle < 0$$
$$a^2||x_1||_F^2 + (1-a)^2||x_2||_F^2 + 2a(1-a)\langle x_1, x_2\rangle < a||x_1||_F^2 + (1-a)||x_2||_F^2$$
$$||ax_1 + (1-a)x_2||_F^2 < a||x_1||_F^2 + (1-a)||x_2||_F^2,$$

where $a \in (0, 1)$.

Now, with usage of strictly convex definition we can write:

$$||Y - (ax_1 + (1-a)x_2)||_F^2 = ||Y - ax_1 - (1-a)x_2||_F^2 =$$
$$= ||a(Y - x_1) + (1-a)(Y - x_2)||_F^2$$

and now with usage of previous formula it is true that:

$$||a(Y - x_1) + (1-a)(Y - x_2)||_F^2 < a||Y - x_1||_F^2 + (1-a)||x_2||_F^2.$$

This clearly shows that $||Y - B||_F^2$ is strictly convex function.

Let us use the fact that the **sum of a convex and strictly convex function is strictly convex**. For the completeness, we show the proof of that well-known result [BV04].
Let g and f be, respectively, convex and strictly convex functions. Then:

$$(f+g)(\lambda x_1 + (1-\lambda)x_2) = f(\lambda x_1 + (1-\lambda)x_2) + g(\lambda x_1 + (1-\lambda)x_2) <$$
$$< \lambda f(x_1) + (1-\lambda)f(x_2) + \lambda g(x_1) + (1-\lambda)g(x_2) =$$
$$= \lambda(f+g)(x_1) + (1-\lambda)(f+g)(x_2).$$

Knowing this we can say that $F(B)$ is a strictly convex function and therefore, there is at most one solution to the problem (7) [BV04]. □

Intuitively the solution to (7) should be symmetric since all the predictor matrices, as well as the matrix of weights, are such.

Proposition 4.2. *If W is a symmetric matrix then the unique solution to the minimization problem defined in* (7) *is symmetric as well.*

Proof. Let $\hat{B}$ be a solution to the problem (7). Now let construct its symmetric part $\widetilde{B} := \frac{1}{2}(\hat{B} + \hat{B}^T)$. We will show that $\widetilde{B}$ is also the solution to the problem (7), thus according to uniqueness it has to be optimal one.

It is easy to see that $f(\widetilde{B}) = f(\hat{B})$. Now we will take under consideration the rest of the components.

$$g(\widetilde{B}) + h(\widetilde{B}) = \lambda_N \left|\left|\frac{1}{2}\hat{B} + \frac{1}{2}\hat{B}^T\right|\right|_* + \lambda_L \left|\left|\frac{1}{2}vec(W \circ \hat{B}) + \frac{1}{2}vec(W \circ \hat{B}^T)\right|\right|_1 \leq$$

$$\leq \frac{\lambda_N}{2}||\hat{B}||_* + \frac{\lambda_N}{2}||\hat{B}^T||_* + \frac{\lambda_L}{2}\left|\left|vec(W \circ \hat{B})\right|\right|_1 + \frac{\lambda_L}{2}\left|\left|vec(W \circ \hat{B}^T)\right|\right|_1 =$$

$$= \lambda_N||\hat{B}||_* + \lambda_L\left|\left|vec(W \circ \hat{B})\right|\right|_1 = g(\hat{B}) + h(\hat{B}).$$

We used the property of **convex function**, and fact $\hat{B}$ shares same singular values with $\hat{B}^T$ and therefore their nuclear norm is same. As a result we obtained the following inequality

$$g(\widetilde{B}) + h(\widetilde{B}) \leq g(\hat{B}) + h(\hat{B})$$

and thus:

$$F(\widetilde{B}) \leq F(\hat{B}).$$

We showed that $\widetilde{B}$ is a solution. According to uniqueness there exists only one optimal solution to the problem, therefore we have $\hat{B} = \widetilde{B}$. □

This proves the claim that the unique solution to (7) is a symmetric matrix.

Proposition 4.3. *Let $\widetilde{B}$ be a unique symmetric solution to the formula* (7)*. Consider conversion of the data based on reordering columns and rows. Let π be any permutation defined as $\pi : \{1, \ldots, p\} \to \{1, \ldots, p\}$ and let denote the corresponding permutation matrix by P^π. Consider the update of matrices Y and W relying on permuting their rows and columns, namely $Y^\pi = P_\pi Y P_\pi^T$ and $W^\pi = P_\pi W P_\pi^T$. Then $\widetilde{B}^\pi$, defined as $\widetilde{B}^\pi := P_\pi \widetilde{B} P_\pi^T$, is the solution to updated problem* (7)*.*

Proof. Let suppose that $\widetilde{B}^{\pi}$ is not an optimal solution to $F(B)$ and there exists symmetric matrix C such that:

$$\frac{1}{2}||Y^{\pi} - C||_F^2 + \lambda_N||C||_* + \lambda_L||vec(W^{\pi} \circ C)||_1 <$$
$$< \frac{1}{2}||Y^{\pi} - \widetilde{B}^{\pi}||_F^2 + \lambda_N||\widetilde{B}^{\pi}||_* + \lambda_L||vec(W^{\pi} \circ \widetilde{B}^{\pi})||_1.$$

We will start from the first component of the inequality above:

$$||Y^{\pi} - C||_F^2 = \langle Y^{\pi} - C, Y^{\pi} - C\rangle = \langle P_{\pi}YP_{\pi}^T - C, P_{\pi}YP_{\pi}^T - C\rangle =$$
$$= \langle P_{\pi}YP_{\pi}^T, P_{\pi}YP_{\pi}^T\rangle - 2\langle P_{\pi}YP_{\pi}^T, C\rangle + \langle C, C\rangle =$$
$$= tr(P_{\pi}YP_{\pi}^TP_{\pi}YP_{\pi}^T) - 2tr(P_{\pi}YP_{\pi}^TC) + tr(CC) \overset{*}{=}$$
$$\overset{*}{=} tr(Y, Y) - 2tr(Y \underbrace{P_{\pi}^TCP_{\pi}}_{\widetilde{C}}) + tr(\underbrace{P_{\pi}^TCP_{\pi}}_{\widetilde{C}}\underbrace{P_{\pi}^TCP_{\pi}}_{\widetilde{C}}) =$$
$$= \langle Y, Y\rangle - 2\langle Y, \widetilde{C}\rangle + \langle \widetilde{C}, \widetilde{C}\rangle = ||Y - \widetilde{C}||_F^2.$$

In place marked with (*) we used property of trace cyclic property. We also defined $\widetilde{C} := P_{\pi}^TCP_{\pi}$, thus $C = P_{\pi}\widetilde{C}P_{\pi}^T$.
Term with l_1 norm on the left side of inequality:

$$||vec(W^{\pi} \circ C)||_1 = ||vec(P_{\pi}WP_{\pi}^T \circ C)||_1 = ||vec(P_{\pi}WP_{\pi}^T \circ P_{\pi}\widetilde{C}P_{\pi}^T)||_1 =$$
$$= ||vec(P_{\pi}(W \circ \widetilde{C})P_{\pi}^T)|| = \sum_{i,j}|W_{\pi(j)\pi(l)}\widetilde{C}_{\pi(j)\pi(l)}| = \sum_{i,j}|W_{i,j}\widetilde{C}_{i,j}| = ||vec(W \circ \widetilde{C})||_1.$$

In reasoning above we used exchangeability of Hadamard product when the same permutation is used for two matrices.

Knowing that C and $\widetilde{C}$ share the same singular values we can rewrite that $||C||_* = ||\widetilde{C}||_*$. Finally taking under account calculations made above we can simplify the left side of the inequality, to $F(\widetilde{C})$.

Taking under consideration the term on the right side of inequality sign we start

again from first equation:

$$||Y^\pi - \widetilde{B}^\pi||_F^2 = ||P_\pi Y P_\pi^T - P_\pi \widetilde{B} P_\pi^T||_F^2 = \langle P_\pi Y P_\pi^T, P_\pi Y P_\pi^T\rangle - 2\langle P_\pi Y P_\pi^T, P_\pi \widetilde{B} P_\pi^T\rangle +$$

$$+\langle P_\pi \widetilde{B} P_\pi^T, P_\pi \widetilde{B} P_\pi^T\rangle = tr(P_\pi Y P_\pi^T P_\pi Y P_\pi^T) - 2tr(P_\pi Y \widetilde{B} P_\pi^T) + tr(P_\pi \widetilde{B}\widetilde{B} P_\pi^T) =$$

$$tr(YY) - 2tr(Y\widetilde{B}) + 2tr(\widetilde{B}\widetilde{B}) = \langle Y, Y\rangle - 2\langle Y, \widetilde{B}\rangle + \langle \widetilde{B}, \widetilde{B}\rangle = ||Y - \widetilde{B}||_F^2.$$

Term with l_1 on right side of inequality at:

$$||vec(W^\pi \circ B^\pi)||_1 = ||vec(P_\pi W P_\pi^T \circ P_\pi \widetilde{B} P_\pi^T)||_1 = ||vec(P_\pi (W \circ \widetilde{B}) P_\pi^T)|| =$$

$$= \sum_{i,j} |W_{\pi(j)\pi(l)} \widetilde{B}_{\pi(j)\pi(l)}| = \sum_{i,j} |W_{i,j} \widetilde{B}_{i,j}| = ||vec(W \circ \widetilde{B})||_1.$$

Again $\widetilde{B}$ and $\widetilde{B}^\pi$ share same singular values so it is true that $||\widetilde{B}||_* = ||\widetilde{B}^\pi||_*$. Right side of inequality can be expressed as $F(\widetilde{B})$.

As shown above:

$$F(\widetilde{C}) < F(\widetilde{B}),$$

which contradicts the optimality of $\widetilde{B}$ and proves claim. □

The proof above shows that LoRSEr is, in some sense, invariant with respect to the permutation of predictor matrices. Precisely, the rearrangement of the columns and rows corresponds exactly to the same rearrangement applied for the estimate. As a consequence, we do not have to fit the model again. The most important fact is that the permutation reveals the expected block or cluster structure can be found at the end of the algorithm procedure.

5 Implementation

To find the optimal solution to (8), we will use the Alternating Direction Method of Multipliers (ADMM) [GM76]. We announce $p \times p$ matrices C and D as new variables. Then, we consider the constrained version of the formula, which is equivalent to (8) with a separable objective function,

$$\underset{B,C,D}{\operatorname{argmin}} \ \big\{f(B) + g(C) + h(D)\big\} \qquad \text{such that} \ \begin{cases} D - B = 0 \\ D - C = 0 \end{cases}, \tag{10}$$

where functions f, b and g are defined as in (8). The algorithm introduces scalars $\delta_1 > 0$ and $\delta_2 > 0$ with dual variable $Z := \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} \in \mathbb{R}^{2p \times p}$ that leads to construction of full augmented formula.

$$L_\delta(B, C, D; Z) = f(B)+g(C)+h(D)+\langle Z_1, D-B\rangle+\langle Z_2, D-C\rangle+\frac{\delta_1}{2}\big\|D-B\big\|_F^2+\frac{\delta_2}{2}\big\|D-C\big\|_F^2.$$

This equation can be simplified since $\langle Z_1, D-B\rangle+\frac{\delta_1}{2}\|D-B\|_F^2 = \frac{\delta_1}{2}\|D+\frac{Z_1}{\delta_1}-B\|_F^2+c_1$ and $\langle Z_2, D-C\rangle + \frac{\delta_2}{2}\|D-C\|_F^2 = \frac{\delta_2}{2}\|D+\frac{Z_2}{\delta_2}-C\|_F^2+c_2$, where c_1 does not include B and c_2 does not include C. Knowing that, the augmented Lagrangian can be simplified to:

$$L_\delta(B, C, D; Z) = f(B) + g(C) + h(D) + \frac{\delta_1}{2}\|D + \frac{Z_1}{\delta_1} - B\|_F^2 + \frac{\delta_2}{2}\|D + \frac{Z_2}{\delta_2} - C\|_F^2.$$

ADMM is an iterative algorithm, which means that it produces the updates of the numerical solution in the iterative process. In each step matrices $B^{[k+1]}$, $C^{[k+1]}$ and $D^{[k+1]}$ are estimated by minimisation of $L_\delta(B, C, D; Z)$ with respect to every primal variable, when in the same time all remaining variables stay fixed. The last step in every iteration is to update dual variables. Taking this under consideration, the problem that should be optimized by the usage of the ADMM model looks as follow:

$$B^{[k+1]} := \underset{B}{\operatorname{argmin}} \left\{ 2f(B) \ + \ \delta_1^{[k]} \left\| D^{[k]} + \frac{Z_1^{[k]}}{\delta_1^{[k]}} - B \right\|_F^2 \right\} \tag{11}$$

$$C^{[k+1]} := \underset{C}{\operatorname{argmin}} \left\{ 2g(C) \ + \ \delta_2^{[k]} \left\| D^{[k]} + \frac{Z_2^{[k]}}{\delta_2^{[k]}} - C \right\|_F^2 \right\} \tag{12}$$

$$D^{[k+1]} := \underset{D}{\operatorname{argmin}} \left\{ 2h(D) \ + \ \delta_1^{[k]} \left\| D + \frac{Z_1^{[k]}}{\delta_1^{[k]}} - B^{[k+1]} \right\|_F^2 \ + \ \delta_2^{[k]} \left\| D + \frac{Z_2^{[k]}}{\delta_2^{[k]}} - C^{[k+1]} \right\|_F^2 \right\} \tag{13}$$

$$\begin{cases} Z_1^{[k+1]} := Z_1^{[k]} + \delta_1^{[k]}\big(D^{[k+1]} - B^{[k+1]}\big) \\ Z_2^{[k+1]} := Z_2^{[k]} + \delta_2^{[k]}\big(D^{[k+1]} - C^{[k+1]}\big) \end{cases} . \tag{14}$$

Updates showed above have analytical solutions and can be efficiently calculated,

which will be presented in the next section of this paper. Values of $\delta_1^{[k]}$ and $\delta_2^{[k]}$ are used as step sizes. Overall convergence of ADMM is assured when they are held constant, but they should be chosen with the highest attention because their size strongly influences the practical performance of ADMM [Xu17]. Stopping criteria for the algorithm are defined the same as in the work of [Brz20] since the problem introduced in this paper is strictly connected to the one in mentioned research. Precisely, the algorithm terminates when both the following criteria are satisfied:

$$\max\left\{\frac{\|C^{[k+1]}-B^{[k+1]}\|_F}{\|B^{[k+1]}\|_F}, \frac{\|D^{[k+1]}-B^{[k+1]}\|_F}{\|B^{[k+1]}\|_F}\right\} < \epsilon_P, \qquad \frac{\|D^{[k+1]}-D^{[k]}\|_F}{\|D^{[k]}\|_F} < \epsilon_D,$$

where the default settings are $\epsilon_P := 10^{-6}$ and $\epsilon_D := 10^{-6}$.

6 Solving optimization problem

In this section, methods of updates for ADMM will be discussed. In each subsection, we will introduce the analytical solution to each sub-problem presented in the previous paragraph.

6.1 Solution to (11)

In the first step we will consider the solution to the following minimisation problem :

$$B^{[k+1]} := \underset{B}{\operatorname{argmin}}\{||Y-B||_F^2 + \delta_1^{[k]}||D^{[k]} + \frac{Z_1^k}{\delta_1^{[k]}} - B||_F^2\}.$$

After introduction of variable $\widetilde{B} := B - (D^{[k]} + \frac{Z_1^k}{\delta_1^{[k]}})$ above equation can be rewritten as

$$B^{[k+1]} = \underset{\widetilde{B}}{\operatorname{argmin}}\{||\underbrace{Y - (D^{[k]} + \frac{Z_1^k}{\delta_1^{[k]}})}_{\widetilde{Y}} - \widetilde{B}||_F^2 + \delta_1^{[k]}||\widetilde{B}||_F^2\} = \underset{\widetilde{B}}{\operatorname{argmin}}\{||\widetilde{Y}-\widetilde{B}||_F^2 + \delta_1^{[k]}||\widetilde{B}||_F^2\}.$$

Optimal value of $B^{[k+1]}$ is found by comparing with zero its derivative:

$$\frac{d}{d\widetilde{B}}(||\widetilde{Y}-\widetilde{B}||_F^2+\delta_1^{[k]}||\widetilde{B}||_F^2)=-2(\widetilde{Y}-\widetilde{B})+2\delta_1^{[k]}\widetilde{B}=0.$$

Consequently, $\widetilde{B}=\frac{\widetilde{Y}}{1+\delta_1^{[k]}}$.

This can be rewritten as

$$B^{[k+1]}-D^{[k]}-\frac{Z_1^k}{\delta_1^k}=\frac{Y-D^{[k]}-\frac{Z_1^k}{\delta_1^k}}{1+\delta_1^k}$$

and finally :

$$B^{[k+1]}=\frac{Y+\delta_1^k D^{[k]}+Z_1^k}{1+\delta_1^k}.$$

6.2 Solution to (12)

The second subproblem of (8) is formulated as:

$$C^{[k+1]}:=\underset{C}{\operatorname{argmin}}\left\{\frac{1}{2}\Big\|\underbrace{D^{[k]}+\frac{Z_2^{[k]}}{\delta_2^{[k]}}}_{M^{[k]}}-C\Big\|_F^2+\frac{\lambda_N}{\delta_2^{[k]}}\|C\|_*\right\}.$$

This term is the same as in [Brz20] so we can use the same technique as shown in this paper.

Proposition 6.1. *For any matrix M and its singular value decomposition $M=U\Sigma V^T$ the optimal solution to:*

$$\underset{C}{\operatorname{argmin}}\left\{\frac{1}{2}\|C-M\|_F^2+\lambda\|C\|_*\right\},$$

has the same singular vectors as M and their singular values can be represented as $s_i^*=(s_i-\lambda)_+:=\max\{s_i-\lambda,0\}$, where s_i represents i-th singular value.

Because of variable substitution now the singular value decomposition that we want to find is defined as follow

$$M^{[k]}=U^{[k]}\Sigma^{[k]}{V^{[k]}}^T.$$

In the next step we can use proposition (6.1) and find next value of $C^{[k]}$ in two

steps:

$$\begin{cases} S^* := \operatorname{diag}\Big(\big[\ (s_1^{[k]} - \frac{\lambda_N}{\delta_2^{[k]}})_+ \ , \ldots , \ (s_p^{[k]} - \frac{\lambda_N}{\delta_2^{[k]}})_+ \ \big]^T \Big) \\ C^{[k+1]} = U^{[k]} S V^{T[k]} \end{cases} ,$$

where function $diag(v)$ creates square matrix with values of vector v on diagonal and zeros elsewhere.

6.3 Solution to (13)

Again the process of update for primal variable D was firstly introduced in [Brz20] but we will conduct the proof of the next proposition in a slightly different manner.

Firstly, let have a look at the formula to optimize but with changed notation. Let introduce variables $K := B - \frac{Z_1}{\delta_1}$ and $L := C - \frac{Z_2}{\delta_2}$. The outcome of formula (13) with changed notation is as follow

$$D^{[k+1]} := \underset{D}{\operatorname{argmin}} \left\{ 2h(D) \ + \ \delta_1^{[k]} \Big\| D - K \Big\|_F^2 \ + \ \delta_2^{[k]} \Big\| D - L \Big\|_F^2 \right\}.$$

Proposition 6.2. *If L, D, K are matrices with the same sizes then,*

$$\delta_1 \big\| D - K \big\|_F^2 \ + \ \delta_2 \big\| D - L \big\|_F^2 = (\delta_1 + \delta_2) \left\| D - \frac{\delta_1 K + \delta_2 L}{\delta_1 + \delta_2} \right\|_F^2 \ + \ \varphi(K, L, \delta_1, \delta_2),$$

where $\varphi(K, L, \delta_1, \delta_2)$ does not dependent on D and can be skipped in optimization problem.

Proof. We will show explicitly the value of $\varphi(K, L, \delta_1, \delta_2)$ by comparing values on the left and right side. We will denote $Left$ as left side of the equation and $Right$ as the right side.

$$Left = \delta_1 \big\| D - K \big\|_F^2 + \delta_2 \big\| D - L \big\|_F^2 = \delta_1 ||D||_F^2 - 2\delta_1 \langle D, K \rangle + \delta_1 ||K||_F^2 + \delta_2 ||D||_F^2 - 2\delta_1 \langle D, L \rangle + \delta_2 ||L||_F^2 =$$

$$= (\delta_1 + \delta_2) ||D||_F^2 - 2\langle D, \delta_1 K + \delta_2 L \rangle + \delta_1 ||K||_F^2 + \delta_2 ||L||_F^2$$

$$Right = (\delta_1 + \delta_2)\left\| D - \frac{\delta_1 K + \delta_2 L}{\delta_1 + \delta_2} \right\|_F^2 + \varphi(K, L, \delta_1, \delta_2) =$$

$$= (\delta_1 + \delta_2)||D||_F^2 - 2\langle D, \delta_1 K + \delta_2 L\rangle + \left|\left|\frac{\delta_1 K + \delta_2 L}{\delta_1 + \delta_2}\right|\right|_F^2 (\delta_1 + \delta_2) + \varphi(K, L, \delta_1, \delta_2)$$

For $Left$ and $Right$ side to be equal the following must hold:

$$\delta_1||K||_F^2 + \delta_2||L||_F^2 = \left|\left|\frac{\delta_1 K + \delta_2 L}{\delta_1 + \delta_2}\right|\right|_F^2 (\delta_1 + \delta_2) + \varphi(K, L, \delta_1, \delta_2).$$

Thus:

$$\varphi(K, L, \delta_1, \delta_2) = \delta_1||K||_F^2 + \delta_2||L||_F^2 - \left|\left|\frac{\delta_1 K + \delta_2 L}{\delta_1 + \delta_2}\right|\right|_F^2 (\delta_1 + \delta_2),$$

□

Now let denote $\delta_1 + \delta_2$ as Δ and move back from K and L notation to it's primal form

$$(\delta_1+\delta_2)\left|\left|D-\frac{\delta_1 K + \delta_2 L}{\delta_1 + \delta_2}\right|\right|_F^2 = \Delta\left|\left|D-\frac{\delta_1(B - \frac{Z_1}{\delta_1}) + \delta_2(C - \frac{Z_2}{\delta_2})}{\Delta}\right|\right|_F^2 = \Delta\left|\left|D-\frac{\delta_1 B + \delta_2 C - Z_1 - Z_2}{\Delta}\right|\right|.$$

Now we can solve (13), as modified Lasso regression problem under an orthogonal design matrix, that formula is as follow:

$$D^{[k+1]} = \underset{D}{\operatorname{argmin}} \left\{ \frac{1}{2}\left\| \underbrace{\left(\delta_1^{[k]} B^{[k+1]} + \delta_2^{[k]} C^{[k+1]} - Z_1^{[k]} - Z_2^{[k]}\right)/\Delta^{[k]}}_{Q^{[k+1]}} - D \right\|_F^2 + \frac{\lambda_L}{\Delta^{[k]}}\left\| \operatorname{vec}(W \circ D)\right\|_1 \right\}.$$

Because this is a typical Lasso problem this equation can be solved similarly. The

concept below is taken from [Tib96].

$$D_{ij}^{[k+1]} = \operatorname{sgn}\left(Q_{ij}^{[k+1]}\right) \cdot \left(\left|Q_{ij}^{[k+1]}\right| - \frac{\lambda_L W_{ij}}{\Delta^{[k]}}\right)_+, \quad \text{for} \quad i,j \in \{1,\ldots,p\}.$$

7 Cross Validation

The explicit solution to problem (7) has been introduced in the previous section. The problem that remains is the selection of optimal parameters λ_N and λ_L. To achieve that goal the cross-validation (CV) was incorporated in the estimation procedure and we will describe the details of that step in the following subsections.

7.1 Finding wide interval for λ_N and λ_L

As the first step, we find the range of the space for CV parameters. In the beginning, we set one of the lambdas to zero and try to find the value for the other one, that is big enough to make our estimation a zero matrix. We start from a value equals to 1 and every next value of tested lambda is one hundred times bigger than the previous. We keep increasing this value until the estimated matrix converges to zero. After estimation for one of the lambdas, we do the same for the other. In the end, we have potentially two wide intervals (one for each lambda) that should be narrowed for better computational performance.

7.2 Narrowing intervals for for λ_N and λ_L

In this step, we will try to find a minimum value for which the estimated covariance matrix is converging to zero. We take obtained interval from the previous step and test its middle value. By testing, we consider checking if its middle value is converging our estimator to zero or not. If this happens we set this value as a new ending of the interval, otherwise as the beginning. This is method is also known as **binary search**. When the difference between the start and end of the interval is smaller than 0.001 we stop the searching process. Having the smallest value of the penalty that is making our estimator zero matrices, we use it as the last value in the exponential interval, which will be searched in order to find the optimal value of penalty.

7.3 Cross validation process

We decided to use 5-fold cross-validation. In the beginning, each row of experiment matrix X is labeled randomly with the number from one to five (the size of each group should be the same or very similar). We take first of all proposed values, from the exponential interval of λ_N and match it in a loop with all possible values of λ_L. In every single matching of two parameters, we calculate 5-fold cross-validation on the standardized experiment matrix. We denote $Y_{train} = X_{train}^T X_{train}$ and $Y_{test} = X_{test}^T X_{test}$ wherein first inner 5-fold loop X_{test} denote sub-matrix of experiment matrix X with rows labeled previously with number one. Accordingly X_{train} is a sub-matrix of X with rows labeled with a different number than one. According to this design in each step we calculate covariance estimator for X_{train} denoted as $\widetilde{B}_{train}$ and loss defined as: $\frac{1}{2}||Y_{test} - \widetilde{B}_{train}||_F^2$. For fixed values of λ_N and λ_L this calculation is repeated 5 times for different X divisions. In the end, for each pair of lambdas values, we have five different numbers that we sum and divide by a number of rows in matrix X. Those values are stored in the final CV matrix where a specific entry denotes the normalized error for a specific value of λ_N and λ_L. After testing all possible combinations of penalties and saving their outcome in the CV matrix, we choose the smallest entry of this matrix and lambdas that were used to calculate it. As the last step, we make a final calculation of the matrix X covariance estimator with selected values.

7.4 CV algorithm pseudocode

Algorithm 1 Cross validation for LoRSEr

$Y = X^T X$ ▷ Basic covariance estimator
assignLabelToEachXRow(X, [1, 2, 3, 4, 5])
$\lambda_{Nstart} \leftarrow 1$
$\lambda_{Lstart} \leftarrow 1$
$\lambda_{NwideInterval} \leftarrow$ findWideInterval(λ_{Nstart}, 0)
$\lambda_{LwideInterval} \leftarrow$ findWideInterval(0, λ_{Lstart})
$\lambda_{NnarrowInterval} \leftarrow$binomalSearchLambda($\lambda_{NwideInterval}$)
$\lambda_{LnarrowInterval} \leftarrow$binomalSearchLambda($\lambda_{LwideInterval}$)
$\lambda_{NfinalInterval} \leftarrow$ expotentialInterval($\lambda_{NnarrowInterval}$)
$\lambda_{LfinalInterval} \leftarrow$ expotentialInterval($\lambda_{LnarrowInterval}$)
for each element **i** in $\lambda_{NfinalInterval}$ **do**
  for each element **j** in $\lambda_{LfinalInterval}$ **do**
    lostForSpecificLambdasPair $\leftarrow$ emptyList()
    for each element **g** in list [1, 2, 3, 4, 5] **do**
      $X_{test} \leftarrow$ findXRowsLabeledAsG(X)
      $X_{train} \leftarrow$ findXRowsNotLabeledAsG(X)
      $Y_{test} = X_{test}^T X_{test}$
      $Y_{train} = X_{train}^T X_{train}$
      covMatrixEstimator = LoRSEr(Y_{train}, i, j) ▷ This is algorithm that was introduced in previous sections
      Loss $\leftarrow$ EstimateLoss(covMatrixEstimator,Y_{test})
      lostForSpecificLambdasPair[g] $\leftarrow$ Loss
    end for
    losPerValidation $\leftarrow$ sumValuesInList(lostForSpecificLambdasPair)
  end for
end for
$\lambda_N, \lambda_L \leftarrow$ pickLambdasWithSmallestLoss()
finalEstimator $\leftarrow$ LoRSEr(Y, λ_N, λ_L)

8 Simulation Experiments

In this section, we will show and discuss the difference in performance between classical estimation and LoRSEr. The whole analysis is made on the assumption that the real covariance matrix that we want to recover has non-zero blocks inside.

Having covariance matrix we generate sample data and based on it we try to recover its real covariance matrix that is as close as possible to the original one that was used for its generation.

8.1 Simulation scenarios

We will consider three scenarios. In the first two, we take under consideration covariance matrices that have non-zero values on diagonal blocks. In the last scenario, we allow the possibility of having non-zero blocks but outside of diagonal. In all scenarios, matrices are symmetric and meet all criteria to be defined as covariance.

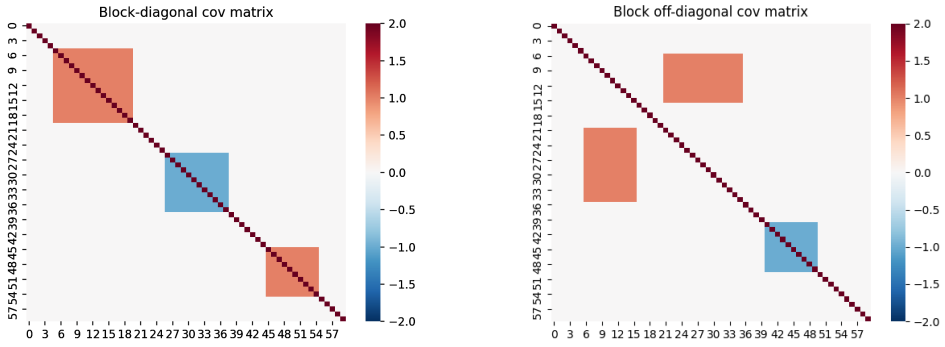
Scenario 1 At the begining we create block diagonal covariance matrix B $\{\mathbf{0}_{5\times5}, s\times\mathbb{1}_{15\times15}, \mathbf{0}_{6\times6}, -s\times\mathbb{1}_{12\times12}, \mathbf{0}_{7\times7}, s\times\mathbb{1}_{10\times10}, \mathbf{0}_{(p-36)\times(p-36)}\}$ where $p = 60$. Then we will test performance of introduced algorithm with different value $s = 2^k$ with $\{-3, -2, \ldots, 5\}$ and compare it to outcome of classical estimation with same values. Then we will conduct two experiments with constant value s. In first experiment $s = 1$ in second $s = 20$. For both $s \in \{1, 20\}$ we test performance of the algorithm for different number of generated rows of matrix X. In this scenario number of rows was in $n \in \{100, 150, 200, \ldots, 800\}$. For each of this number, loss was calculated for both classical estimator and LoRSEr.

Scenario 2 In the second scenario we use the same structure of covariance matrix as in Scenario 1 and experiment with amplification over time of certain diagonal block value signals. It means that one of the bloc-diagonal blocks in covariance matrix B has starting value 0 that increases over time till is significantly greater than values in other non-zero blocks, that remain constant through the whole experiment. This scenario was made to check how the performance of LoRSEr is handling the case when the values of one of the block-diagonal covariance matrices are much smaller or bigger than other non-zero entries.

Scenario 3 In the last scenario we check how LoRSEr is performing for covariance matrix B that has block not only on diagonals but also out of it. We used exactly the same structure of covariance matrix as in Figure (2(b)). Same experiments as in Scenario 1 were made.

8.1.1 Diagonal and off-diagonal covariance matrix

To make it easy to understand we will introduce in the plot below the construction of both diagonal and off-diagonal covariance matrices that were used in experiments described in Scenarios 1-3.



(a) Block-diagonal covariance matrix (b) Block off-diagonal covariance matrix

Figure 2: To create experiment data we used both kinds of covariance matrices structure. Values are re-scaled to make blocks more visible but the structure remains the same over simulation process. (a) Two positive and one negative non-zero block placed on diagonal of the covariance matrix. (b) Two positive blocks are out of diagonal, the one that is left, takes negative values.

8.2 Simulation implementation

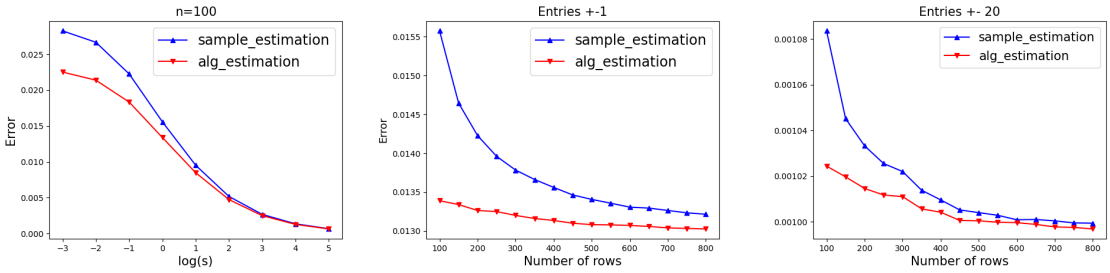
To create experimental data at the beginning we made covariance matrix B with size 60×60 that has three diagonal blocks on the diagonal, the rest of the entries were filled with zero. Then we sampled from a multivariate normal distribution with covariance B and mean vector equal to 0. The sampled variable was set as a row in the experimental matrix X. For each scenario introduced above, we used LoRSEr implementation in Python and cross-validation introduced in the previous chapter. Every time the best λ_N and λ_L were chosen and applied for the final evaluation. In every simulation, we used W matrix filled with ones everywhere except diagonal, where values were set to zero. To make differences between classical and LoRSEr estimators visible we also included some plots that highlight the contrast between final covariance matrix estimation

8.3 Simulation results

Performance of each method was measured by loss function, where $\hat{B}$ is estimator and B real covariance matrix, is defined as MSEr $= ||\hat{B} - B||^2_{F^\star} / ||B||^2_{F^\star}$ where $||.||_*$ is Frobenius Norm of matrix, with diagonal entries excluded.

8.3.1 Scenario 1 Covariance matrix estimation based on different sizes of experimental data and signal strength

Figures below represent the MSEr of classical (marked blue) and LoRSEr estimator, for which calculation we used formula in (7). The first plot on the right shows the error of final estimation on the y-axis and corresponding values of the correlation in diagonal blocks on the x-axis, for the number of rows $n = 100$. For all cases, LoRSEr is more accurate, especially when non-zero values are small. In the second experiment of this scenario, we filled real covariance matrix B with diagonal block entries and values $\{\pm 1\}$, where positive entries are set on two diagonal blocks and negative on the middle one as described in Scenario 1. Then we created experimental data with covariance matrix B as the parameter. In every iteration, we re-created the experimental matrix X but we were increased the number of rows. Having X we tried to recover B with classical and LoRSEr algorithm and calculate loss function for each estimated covariance matrix. LoRSEr also required cross-validation for every single different X size to create the best possible estimator of B. The same procedure was introduced for values $\{\pm 20\}$



(a) Different values (b) Small values (c) Big values

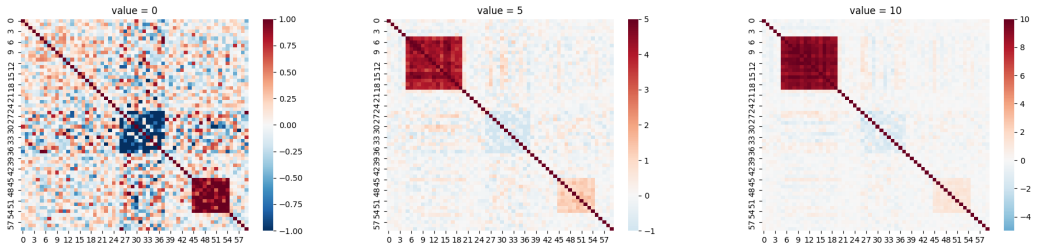
Figure 3: Relative mean squared errors (MSEr) of estimators obtained from classical covariance estimation and with usage of LoRSEr. Each point represents the MSEr calculated for one of the estimators taking into account the number of rows in the experiment matrix (a)MSEr for different values in diagonal blocks and experiment matrix X of size 100×60 (b) MSEr of estimators for entries $\{\pm 1\}$ in B (c) MSEr of estimators for entries $\{\pm 20\}$ in B

In all cases, LoRSEr outperforms the classical estimator in terms of MSEr. The difference is more visible when we include smaller entries in covariance matrix B especially when experimental matrix X has relatively small size ($n = 100$). Both algorithms are lowering their MSEr for the higher number of rows in the experimental matrix X but there is no single case where the classical estimator would perform better than LoRSEr. The value of MSEr itself and the difference between estimators tend to be smaller when B matrix is filled with greater values. This is natural behavior since signal strength in covariance matrix has its a reflection on generated experimental data. We see that LoRSEr data has the biggest advantage when the signal is weak and amount of data is small, what frequently happens in real-world applications and confirms the usefulness of LoRSEr in such situation.

8.3.2 Scenario 2: Increasing the signal strength in one of the block matrices

In the second scenario, we will focus attention not on MSEr but on the real values that are recovered by classical and LoRSEr algorithm. Each estimation method will be checked for different values $s \in \{0, 5, 10\}$ of upper-left diagonal block in the covariance matrix. Every change of the value is represented in the creation of the experiment matrix and estimation made on it. While the value of the upper-left block is changing, other two values remain the same during the whole experiment. Each estimation was based on experimental matrix $X_{500\times60}$. The diagonal value is higher than shown in the scale on the right side of each plot, to fulfill assumptions about being covariance matrix, but we kept this scale to make our research more readable.

We will start with form estimation made by the classical approach. When $s = 0$ other two blocks are visible but the whole matrix is full of noise. For $s = 5$ upper-left block is clearly visible, the other two are still there but the noise around is making them blurred and hard to find. Same situation occurs for $s = 10$.

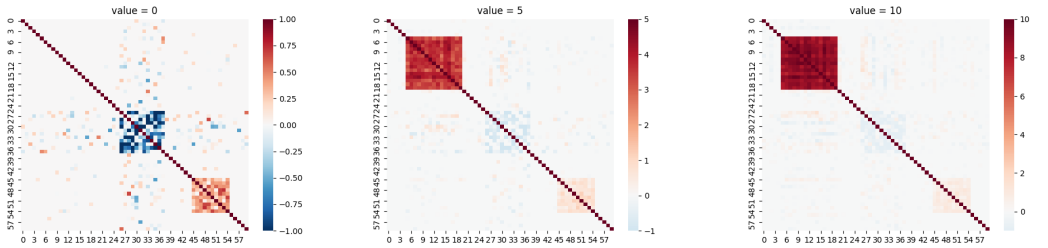


(a) No signal for first block (b) Strength of signal: 5 (c) Strength of signal: 10

Figure 4: Estimation of the covariance matrix with classical estimation. Plots show how the estimate changes when we are increasing the strength of the signal of the upper block in covariance matrix and generate data according to it. (a) Block set to zero; (b) values of block set to 5; (c) values of block set to 10.

Now let analyze the situation when we use LoRSEr for covariance matrix estimation. When the upper-left block does not exist, LoRSEr is recovering the other two blocks and is allowing to remove the noise around blocks, thanks to the introduced l_1 norm. This form is much less messy and allows to find blocks at the first glance. For the other two values of s LoRSEr has no problem with recovery highest-value block and two others which values are on a constant level. At the same time, LoRSEr

is canceling most of the noise around it so there is no problem with finding proper clusters.



(a) No signal (b) Strength of signal: 5 (c) Strength of signal: 10

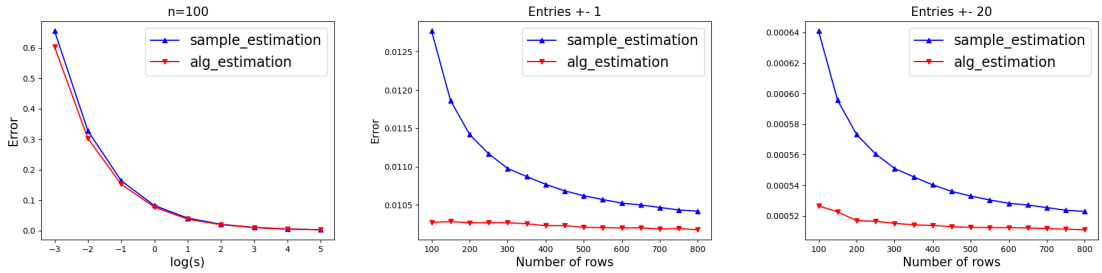
Figure 5: LoRSEr estimation of the covariance matrix with. Again we show different covariance estimations that depend on the value of the upper-left block. (a) Block set to zero; (b) values of block set to 5; (c) values of block set to 10.

Overall, the biggest difference between these two methods is noise removal. LoRSEr is allowing much less noise to the final estimate and clusters are easier to find. In both cases the two left clusters are visible but, because of the scale, they are less sharp than when values of the biggest cluster were set to zero. Although one of the clusters has much higher values than others, LoRSEr does not lose its signal and at the same time is revealing the structure more clearly.

8.3.3 Scenario 3: MSEr for off-diagonal blocks in covariance matrix

In this scenario, we are changing the structure of the true covariance matrix B. Instead of having all non-zero entries in the block that lies on diagonal, we left only the smallest block and create non-zero blocks out of diagonal. Again, as in Scenario 1, we are increasing the number of samples created with usage of this covariance matrix and stored in matrix X. Points indicate MSEr of each method and different row numbers n in matrix X. In this scenario, we also experiment with values inside of B matrix. The same rule as in Scenario 1 is applied, one experiment is with matrix filed with $\{\pm 1\}$ blocks, and second with $\{\pm 20\}$

Similarly, as in the previous situation, LoRSEr is creating an estimation of B more accurately than classical estimation. This difference, especially for the (a), is not significant, but still clearly visible, especially for small values. In the other two charts, there is a contrast between estimators, both for small and big values in B. The highest difference is visible again when experiment matrix X is the smallest, with



(a) Different values (b) Off diagonal small values (c) Off diagonal big values

Figure 6: MSEr is calculated again for classical estimator and LoRSEr but this time during data creation we used covariance matrix that has non zero blocks, not only on diagonal but also off it (a)MSEr for different values in off-diagonal and experiment matrix X of size 100×60 (b) MSEr of estimators for entries $\{\pm 1\}$ in B (c) MSEr of estimators for entries $\{\pm 20\}$ in B.

$n = 100$. The scale of error for a stronger signal in B is smaller than in otherwise, but LoRSEr seems to keep its advantage on a similar level as in the case with smaller entries in the covariance matrix. This experiment showed that LoRSEr works better than classical approximation also when covariance matrix B has off-diagonal blocks, especially when data has a small number of samples.

9 Conclusion

In this work, we have attempted to find the best possible approximation of covariance matrix that is sparse and low rank with the usage of LoRSEr. In the beginning, we defined an optimization problem (7) that should be solved and we proved the properties of the corresponding solution, such as the existence and uniqueness questions.. Then we introduced the ADMM algorithm that was iteratively solving problem (7) with the explanation of every optimization process divided into sub-functions. After introducing the numerical solution to the problem, we also defined a cross-validation algorithm that is capable of finding the optimal tuning parameters for nuclear and l_1 norm penalties. We conducted some experiments to compare the performance of classical and LoRSEr approximation of covariance matrix. In the beginning, we tested how the given algorithm is performing for different correlation values and how its accuracy is changing when the experiment matrix is increasing its number of observations. Then we tested LoRSEr when one of the block-diagonal values was constantly increasing and reaching values much greater than in other blocks. In the

end, we checked how LoRSEr is performing when potential clusters with a strong signal are off-diagonal. In all mentioned cases LoRSEr was performing much better, it was able to not only spot the most important clusters but also shrink the unimportant connections to zero.
There are still some improvements that can be done to enhance the performance of the newly introduced algorithm. One of them is to more precisely define values of weight matrix W. If we have prior knowledge about potential correlation structure we could lower the weights in these spots of the matrix W and on the other hand make them higher when we are certain about lack of correlation between given features. The concept of introducing the weights can be also employed for the first part of the objective function where in cooperation with Frobenius norm, it can solve potential problem concerning estimation of diagonal values. These remarks if improved in the future can make a decisive contribution to the efficiency of the new algorithm.

References

[Bas18] Michailidis Basu Li. "Low rank and structured modeling of high-dimensional vector autoregressions". In: arXiv:1812.03568 (2018).

[Brz20] Damian Brzyski. "A Sparsity Inducing Nuclear-Norm Estimator (SpINNEr) for Matrix-Variate Regression in Brain Connectivity Analysis". In: arXiv:2001.11548 (2020).

[BV04] Boyd and Vandenberghe. Convex Optimisation. Cambridge University Press, 2004.

[Emi21] Qin Qin Huang Emil Uffelmann. "Genome-wide association studies". In: Nat Rev Methods Primers 1, 59 (2021).

[GC14] Huang Goldsmith and Crainiceanu. "Smooth scalar-on-image regression via spatial Bayesian variable selection". In: Computational and Graphical Statistics (2014).

[GM76] D. Gabay and Mercier. "A dual algorithm for the solution of nonlinear variational problems via finite element approximation". In: Computers Mathematics (1976).

[Llo18] Fidel Alfaro-Almagro Lloyd T. Elliott Kevin Sharp. "Genome-wide association studies of brain imaging phenotypes in UK Biobank". In: Nature 562 (2018).

[RNy10] Jimmy Z.Liu Allan F.Mcrae Dale R.Nyholt. "A Versatile Gene-Based Test for Genome-wide Association Studies". In: American journal of human genetics (2010).

[RO10] P. T. Reiss and R. T. Ogden. "Functional generalized linear models with images as predictors". In: Biometrics, 66 (2010).

[Tib96] Robert Tibshirani. "Regression Shrinkage and Selection via the Lasso". In: Journal of the Royal Statistical Society (1996).

[Wan14] Zhu Wang Nan. "Regularized 3D functional regression for brain image data via haar wavelets". In: The Annals of Applied Statistics (2014).

[Xu17] Figueiredo Xu. "Adaptive relaxed admm: Convergence theory and practical implementation". In: 2017 IEEE Conference on Computer Vision (CVPR) (2017).

[Zho14] Li Zhou. "Regularized matrix regression". In: Royal Statistical Society (2014).