Dynamic Covariance Estimation under Structural Assumptions via a Joint Optimization Approach

Wenyu Chen
Massachusetts Institute of Technology
USA

Riade Benbaki
Massachusetts Institute of Technology
USA

Yada Zhu
MIT-IBM Watson AI Lab, IBM Research
USA

Rahul Mazumder
Massachusetts Institute of Technology
USA

ABSTRACT

Dynamic covariance estimation is a problem of fundamental importance in statistics, econometrics, with important applications in finance, especially portfolio optimization. While there is a large body of work on static covariance estimation, the current literature on dynamic covariance estimation is somewhat limited in comparison. We propose a flexible optimization framework to simultaneously learn covariance matrices across different time periods under suitable structural assumptions on the period-specific covariance matrices and time-varying regularizers. We propose a novel efficient joint optimization algorithm to learn the covariance matrices simultaneously. Our numerical experiments demonstrate the computation improvements of our algorithm over both off-the-shelf solvers and other dynamic covariance estimation methods. We also see notable gains in terms of test MSE and downstream portfolio optimization tasks on both synthetic and real datasets.

CCS CONCEPTS

• **Mathematics of computing** → **Multivariate statistics**; *Mathematical optimization*; **Time series analysis**; *Dimensionality reduction*; • **Applied computing** → **Economics**.

KEYWORDS

dynamic covariance matrix, joint optimization, time varying, factor model, sparsity, portfolio optimization

ACM Reference Format:
Wenyu Chen, Riade Benbaki, Yada Zhu, and Rahul Mazumder. 2023. Dynamic Covariance Estimation under Structural Assumptions via a Joint Optimization Approach. In *4th ACM International Conference on AI in Finance (ICAIF '23), November 27–29, 2023, Brooklyn, NY, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3604237.3626885

1 INTRODUCTION

Covariance matrix estimation is a fundamental problem in statistics with wide applications in various domains, ranging from economics and finance to biology, social networks, and health sciences [21]. In financial applications, for example, covariance matrices play a key role in understanding the relationship between different market entities, which is essential to build a diversified portfolio of assets [21, 37].

Covariance estimation and portfolio optimization. There is a large body of exciting work on estimating large static covariance matrices, as well as their applications in portfolio optimization; see [22] for a comprehensive review. Such static approaches are often reduced to using a small sample size in contexts where the dynamics can change through time, in order to capture the most recent correlations, as is the case of financial asset correlations for example. This can result in significantly underestimated portfolio risk compared to approaches that model the dynamics of these correlations.

Dynamic covariance estimation. For multivariate time-series data, there is an impressive literature on dynamic covariance estimation techniques using either parametric GARCH-based methods [4, 13, 15, 16, 37, 39] or non-/semi-parametric kernel approaches [8–10]. Parametric methods usually make use of stylized and somewhat restricted models. For non-/semi-parametric models, a common approach is to use kernel-smoothing approaches to learn covariance matrices, and it could be difficult to enforce additional structural assumptions on (a) individual covariance matrices and/or (b) the manner in which they evolve over time. To this end, we present a new optimization framework that overcomes both shortcomings — it is capable of incorporating minimal structural assumptions without assuming any specific dynamics. Our experiments show the relaxed assumptions lead to the benefits compared to the existing methods. The training processes of many current methods for dynamic covariance estimation [16, 39] are usually quite involved: they employ EM algorithms, multi-stage separate learning or Bayesian type MCMC approaches, and could be slow (as our experiments show) when the dimension of the covariance matrices is large. On the contrary, our proposed framework uses a joint optimization criterion to simultaneously learn the covariance matrices across time. By proposing specialized algorithms, we show that the learning can be done quite efficiently, 100 times faster than off-the-shelf solver. *Structural assumptions.* A major challenge in static covariance matrix estimation is the limited sample sizes in high-dimensional settings. This issue becomes even more severe for the dynamic covariance estimation, because there are more parameters to estimate in the time-varying model. To address the issue in static settings, researchers have proposed different regularization approaches [22]; two of the most common assumptions are

ICAIF '23, November 27–29, 2023, Brooklyn, NY, USA

ACM ISBN 979-8-4007-0240-2/23/11.
https://doi.org/10.1145/3604237.3626885

low-rank factor models [18, 20] and (conditional) sparsity [1, 24], or a combination of both [6, 23]. In this paper we propose a general framework to estimate time-varying covariance matrices under different structural assumptions (e.g., low-rank, sparsity, and their combination). In addition to the low-rank and sparsity assumptions imposed on the covariances themselves, we also make simple structural assumptions on how the covariance matrices evolve, to avoid specifying over-complicated, rare dynamics. To achieve this, we generalize the idea of earlier work on fused Lasso problem [49] to impose different types of regularization on the difference of covariances to enforce slow/sparse varying pattern across covariances.

Joint optimization framework. Solving such large-scale optimization problems can be computationally challenging for off-the-shelf optimization solvers, necessitating the development of specialized scalable algorithms. We propose a block-coordinate descent type method [51] to simultaneously learn all these components. Our approach results in significant improvements over the current methods, in terms of computational efficiency, estimation accuracy, as well as downstream portfolio quality.

Contributions. The key contributions of our work can be summarized as follows: (1) We propose a new optimization framework that learns the covariance matrix of time series for each period simultaneously under minimal structural assumptions of low-rank factor model and sparse idiosyncratic risks. (2) We propose three regularizers to cater to three different assumptions on the evolution pattern of low-rank and sparse parts of covariance matrices. (3) We propose to use a block-coordinate descent-type method to address the computational challenges of joint training and provide computational guarantees for the algorithm. Extensive experiments demonstrate that our method is 100 times faster than off-the-shelf SCS solver for 2 out of 3 of our models. (4) Finally we show that our proposed framework is not only more efficient but also provides a better covariance estimator, in terms of both estimation errors and downstream portfolio optimization tasks which demonstrates the benefits of using the relaxed structural assumptions compared to the existing methods.

2 RELATED WORK

Among the rich literature in dynamic covariance estimation in statistics/econometrics, one line of literature considers GARCH-type dynamic models, extending univariate ARCH and GARCH models [3, 14] to the multivariate case, such as VEC-GARCH model [4], BEKK [15]. Usually those papers make very specific assumptions on the dynamics of how the time series and covariance matrices evolve over time, and recent work extends and generalizes these models; see [16, 37, 39, 41] and references therein. Since these models are parametrized in a sophisicated way, joint learning of the model parameters could be very difficult, even through the EM algorithm [11]. Instead, the parameters are learned separately via multi-stage procedures or through Bayesian-type estimation approaches (e.g., simulation-based approaches [26]). Another line of literature focuses on nonparametric and semi-parametric estimation approaches, such as kernel smoothing estimations [8, 10]. Incorporating structural assumptions (e.g, sparsity) within these procedures is challenging. Our approach differs: we learn the covariances directly and simultaneously (instead of multi-stage separate learning), with minimal structural assumptions (instead of assuming specific dynamics on the time series).

In the context of dynamic covariance models, factor models are commonly used; see [15, 17, 52]. Another line of work includes Dynamic Factor Models (DFM [27])—see [48] for a nice review. Another line of work [33, 36]—known as Stochastic Volatility (SV) model—models the covariance process in a stochastic manner.

Not surprisingly, there is a larger body of work on estimating static covariance matrices compared to dynamic covariance estimation. To address the low-sample high-dimensional settings, researchers have proposed various approaches and techniques to consistently estimate large covariance matrices. For example, the Ledoit-Wolfe shrinkage estimator [40] is a well-known method that is commonly used as benchmark in literature. [23] proposes the POET estimator that uses PCA followed by thresholding to learn the low-rank-plus-sparse structure in the covariance space. Optimization-based procedures have been developed for latent-variable Graphical Lasso [6] and robust PCA [5]. However, to our knowledge, extensions of these models in to the dynamic time-series context have not yet been explored. A special case of our framework is the fused lasso estimator [49] where one performs signal estimation under an ℓ_1 constraint on the successive differences. Since the original work [49], several generalizations and extensions have been proposed [2, 28, 47, 53] in different settings. We further extend these models into dynamic covariance estimation settings. In addition to a fused lasso-type penalty, we also use other smooth regularizers, with different operational characteristics.

Our framework is most related to [36] and [52]. Both works use the same low-rank-plus-sparse decomposition assumption on covariances as we do, but all three frameworks utilize different approaches to model and learn the dynamics. [52] extends the POET estimator [23] to the dynamic setting using a two-stage nonparametric approach—local PCA followed by local generalized shrinkage. [36] proposes a latent-factor SV model that assumes specific dynamics of covariances, and estimation and inference are based on Bayesian MCMC. Our work differs from these two works in that (i) we have minimal structural assumptions on the dynamics which is more general than [36], while [52] has no assumption on the dynamics; (ii) we propose an efficient algorithm for joint training across different time-blocks, while [52] uses a two-stage separate learning procedure.

Comparisons with [36] show that our proposal leads to at least a 30-fold runtime improvement and significant gains in MSE performance (75% improvement) on synthetic data.

In this paper, we illustrate the performance of our proposed covariance matrix estimation procedure in the context of portfolio optimization [40, 42]. In portfolio optimization, using the sample covariance estimator to find the optimal weights, leads to an underestimated out-of-sample risk [38]. A great amount of work has therefore been done in order to reduce the amount of noise in estimating the covariance structures specifically to enhance the performance of optimal portfolios obtained using these correlation structures, e.g. [38, 40, 50]. Section 6 presents an in-depth empirical investigation on this topic.

3 PROBLEM SETUP AND ASSUMPTIONS

In this section, we present notations used in this paper, as well as the problem setup and model assumptions for the dynamic covariance matrix estimation.

Notations. We use $\mathbb{R}^d$ to denote the set of all d-dimensional vectors, and use $\mathbb{R}^{d\times p}$ to denote the set of all $d \times p$ matrices. $\mathbb{S}^p$ and $\mathbb{S}^p_+$ denote the set of all symmetric matrices and all positive-semidefinite (PSD) matrices in $\mathbb{R}^{p\times p}$, respectively. We denote by $[N] = \{1, 2, \ldots, N\}$. For a matrix $X = [x_{ij}]_{i,j} \in \mathbb{R}^{d\times p}$, its frobenius norm is defined as $\|X\|_F = \sqrt{\sum_{i,j} x_{ij}^2}$. For a square matrix $X = [x_{ij}]_{i,j=1}^p \in \mathbb{R}^{p\times p}$, the trace of X is defined as $\mathrm{tr}(X) = \sum_{i=1}^p x_{ii}$; $\|X\|_{1,\mathrm{off}}$ is defined as $\sum_{i=1}^p \sum_{j\neq i} |x_{ij}|$. For a set A, $\mathbb{1}\{A\}$ is a function such that if $x \in A$, $\mathbb{1}\{A\}(x) = 0$; otherwise, $\mathbb{1}\{A\}(x) = +\infty$.

Problem setup. We consider a multivariate time series $\{y_t\}_1^T$ with m periods and T time steps in total. Each period $k \in [m]$ contains T_k consecutive time steps indexed by $\mathcal{I}_k$. At each time step t, we observe the variables of interest $y_t \in \mathbb{R}^p$ of p different entities. We assume that within the same period k, y_t shares the same covariance matrix Σ_k, i.e. $\mathrm{Cov}(y_t, y_t) = \Sigma_k$ for $t \in \mathcal{I}_k$; across different periods, covariance matrices Σ_k are smoothly changing. The goal is to estimate the dynamically changing covariance matrix from the observed time series.

Structural assumptions on covariances. It is well-known that when the dimension is large, the sample covariance matrix is singular and it can lead to large estimation error due to limited sample sizes [22]. Therefore, for statistically meaningful inference, additional structural assumptions need to be imposed on the covariance matrix. A common assumption is that the different entities depend on a small subset of factors. For example, stock returns depend on global macro-economic factors and sector-specific factors [18, 19] that influence all stocks. In addition, each stock has its own individual idiosyncratic component, most of which are uncorrelated from each other, with some exceptions that can be due to mergers, acquisitions, competition or some local/domestic common events that is not captured by the global factors. Based on these observations, we consider the financial factor models [19, 20]. Specifically in our setting, for each period k, we assume that $y_t = B_k f_t + u_t$ for a few common (latent) factors $f_t \in \mathbb{R}^r$ and a coefficient matrix $B_k \in \mathbb{R}^{p\times r}$ and a residual u_t that is uncorrelated with the common factors f_t. Here, each component of the common factor f_t can be considered as one source of systematic risks, while each component of the residual u_t can be regarded as idiosyncratic risk of each entity. This factor model leads to the following decomposition of the covariance matrix: $\Sigma_k = B_k \Sigma_{f,k} B_k^\top + \Sigma_{u,k}$, of which the first part has rank of $r \ll p$. We further assume that $\Sigma_{u,k}$ is sparse, indicating that there are only a few strong connections between the idiosyncratic risks of different entities. In the rest of paper, we will use L_k to denote the low rank component $B_k \Sigma_{f,k} B_k^\top$ and use R_k to denote the sparse component $\Sigma_{u,k}$, which leads to the following low-rank-plus-sparse decomposition $\Sigma_k = L_k + R_k$. Such decomposition is very common in both financial econometrics and high-dimensional statistics literature [22, 23, 32].

Structural assumptions on evolution pattern. In addition to imposing a low-rank-plus sparse structure on the covariance matrices, we also make structural assumptions on the way these covariance matrices evolve over time (e.g slow/sparse evolving). To this end, we use different regularizations on the matrix differences between successive periods, which we present in more details in Section 4. Instead of learning each period separately, the joint learning framework allows us to make full use of all observations during the training phase, while also allowing the covariance structures to change over time. As discussed previously, there is a trade-off when deciding on the amount of observations for learning a covariance matrix. More observations allow for a more accurate and robust estimation, while limiting the observations to a more recent period allows estimating the most recent covariances in a potentially changing dynamic. Our estimator, through the regularization over the evolution pattern, offers more flexibility to deal with this issue, and through out our experiments, we show that this flexibility offers significant gains, both in terms of estimation error on future unseen observations, as well as improved portfolio selection when used in for portfolio optimization.

Organization of paper. In what follows, we present our joint modeling framework in Section 4. In Section 5, we propose an efficient joint optimization algorithm to solve the modeling framework, provide some convergence properties for our algorithm, and present extensive numerical results to demonstrate the efficiency of our algorithm, compared to an off-the-shelf optimization solver. In Section 6, we present numerical experiments that showcase the improvements of our proposed framework over the baseline methods on both synthetic and real datasets, in terms of different performance metrics.

4 MODELING

We start our exposition with the single-period problem under the low-rank-plus-sparse structure to learn the (static) covariance matrix, and then extend the estimator to the multi-period setting.

4.1 Single-period problem—static estimation

We first consider the static covariance matrix problem with the low-rank-plus-sparse structure. There are two general approaches: two-stage approach and joint learning approach. The two-stage approach (e.g., [20, 23]) learns the latent factors f_t by principal component analysis (PCA), and then learns the covariance matrix of the residual u_t by shrinkage or thresholding estimators (e.g., [1, 40]). However, the two-stage approach cannot be easily adapted to the dynamic setting where slow/sparse varying patterns are considered.

The alternative joint learning approach learns the low-rank part L and the sparse part R in the decomposition $\Sigma = L + R$ simultaneously, via the following optimization problem

$$\min_{L,R} \; f(L, R; S_k) = \frac{1}{2}\|L + R - S_k\|_F^2 + \rho_L(L) + \rho_R(R). \qquad (1)$$

Above, the penalty function ρ_L induces a low-rank and ρ_R induces sparsity. Here, $S_k \in \mathbb{R}^{p\times p}$ denotes the sample covariance matrix of the time-series in period k. Assume that $(\hat{L}_k, \hat{R}_k)$ is the optimal solution to the optimization problem (1), the (static) covariance estimation for the single period k is given by $\hat{\Sigma}_k = \hat{L}_k + \hat{R}_k$.

To obtain a low-rank matrix L, a natural choice is to directly constrain its rank with ρ_L being the corresponding indicator function.

One can also use a convex penalty function—a common approach is to take ρ_L as the nuclear norm of a matrix (i.e., the sum of singular values). Note that for a PSD matrix L, the nuclear norm becomes the trace $\mathrm{tr}(L)$. Here, we combine these two penalties to get a more general regularizer

$$\rho_L(L) = \alpha\,\mathrm{tr}(L) + \mathbb{1}\{L \in \mathbb{S}^p_+, \mathrm{rank}(L) \le \bar{r}\}. \tag{2}$$

It can be easily verified that the aforementioned two penalties are special cases of (2). Note that this penalty is essentially an ℓ_0-ℓ_1 type penalty [45], which is shown to help prevent overfitting in the regime of low-signal-to-noise ratios in sparse regression.

To enforce sparsity, we use an ℓ_1-type regularization on the off-diagonal entries of R with a symmetry constraint on R, i.e.

$$\rho_R(R) = \beta\|R\|_{1,\mathrm{off}} + \mathbb{1}\{R \in \mathbb{S}^p\}. \tag{3}$$

We do not penalize the diagonals of R because they correspond to the variances of u_t, not the covariances between different entities. Also, we drop the PSD constraint on R for faster computations. In numerical experiments, we observe that the optimal R is still PSD[1].

We note that the latent variable graphical lasso [6] has a similar formulation to (1), but it is based on the log-likelihood loss in the inverse covariance space. Here, we are interested in modeling the covariance matrix directly, and the square loss is computationally tractable. Besides, (1) is closely related to the well-known robust PCA problem [5].

4.2 Multi-period problem—dynamic estimation

To extend the single-period setting to the dynamic setting, based on the objective for the single period k, we consider the following optimization problem

$$\min_{L_k,R_k,k\in[m]} \sum_{k=1}^{m} f(L_k, R_k; S_k) + \sum_{k=1}^{m-1} \mathcal{D}((L_{k+1}, R_{k+1}), (L_k, R_k)), \tag{4}$$

with some discrepancy metric $\mathcal{D}(\cdot,\cdot)$ penalizing the distance between the estimates across two successive periods. Intuitively, in addition to looking at the loss $f(L_k, R_k; S_k)$ for each single-period k, the objective in (4) ensures that the estimates from $(k+1)$-th period cannot be too far away from the estimates from k-th period. Depending on the nature of evolution, we can choose different discrepancy metrics. Specifically, we consider the following three different discrepancy metrics that correspond to the structural assumptions mentioned in Section 3:[2]

$$\mathcal{D}_1((L_{k+1}, R_{k+1}), (L_k, R_k)) = \frac{\gamma}{2}\|L_{k+1} + R_{k+1} - L_k - R_k\|_F^2 \quad \text{(WLR)}$$

$$\mathcal{D}_2((L_{k+1}, R_{k+1}), (L_k, R_k)) = \frac{\gamma}{2}\|L_{k+1} - L_k\|_F^2 + \frac{\rho}{2}\|R_{k+1} - R_k\|_F^2 \quad \text{(WLWR)}$$

$$\mathcal{D}_3((L_{k+1}, R_{k+1}), (L_k, R_k)) = \frac{\gamma}{2}\|L_{k+1} - L_k\|_F^2 + \rho\|R_{k+1} - R_k\|_{1,\mathrm{off}} \quad \text{(WLPR)}$$

where, $\gamma, \rho \ge 0$ are regularization parameters. In these formulations, we use square Frobenius norms on the differences to enforce the smooth changes between either covariances or their low-rank/sparse components; we use ℓ_1 norms on the differences to enforce the sparse changes between the sparse components of covariances. These regularizers are inspired by the fused lasso problem [49] and its extensions [47, 53]. Such total variation-type penalty functions are commonly used in time-varying regression problems, but to our knowledge, estimator (4) has not been studied earlier. We note that a similar penalty is also used in time-varying network inference in the context of the so-called time-varying graphical lasso estimator [28]. This estimator however, operates on the inverse covariance matrices Θ_k's, and uses the log-likelihood loss with penalties of the form $\psi(\Theta_{k+1} - \Theta_k)$. However, the PSD constraints on Θ_k's potentially make the computations intractable. [28] do not consider a low-rank and sparsity structure in their Θ, and hence do not have separate penalties over the changes in L_k and R_k.

With all these three different types of regularization, our final objective function for the matrix estimation problem with respect to $\mathcal{L} = \{L_k\}_1^m$ and $\mathcal{R} = \{R_k\}_1^m$ is:

$$\begin{aligned} \min_{\mathcal{L},\mathcal{R}} F(\mathcal{L},\mathcal{R}) := \frac{1}{2}\sum_{k=1}^{m}\|L_k + R_k - S_k\|_F^2 + \alpha\sum_{k=1}^{m}\mathrm{tr}(L_k) \\ + \beta\sum_{k=1}^{m}\|R_k\|_{1,\mathrm{off}} + \sum_{k=1}^{m-1}\mathcal{D}((L_{k+1}, R_{k+1}), (L_k, R_k)), \\ \text{s.t. } L_k \in \mathbb{S}^p_+, \mathrm{rank}(L_k) \le \bar{r}, R_k \in \mathbb{S}^p. \end{aligned} \tag{5}$$

Notice that when taking $\bar{r} = p$, the rank constraint is redundant, and the problem (5) becomes convex. The problem (5) has $O(mp^2)$ variables and 4 hyper-parameters $(\alpha, \beta, \gamma, \bar{r})$. This means that for one set of hyperparameters, we need to solve a problem with about $O(10^4) - O(10^6)$ variables if we consider hundreds or thousands of companies (i.e., $p \sim O(10^2) - O(10^3)$). Fine-tuning over thousands of hyperparameter requires therefore a scalable algorithm to solve the problem in a few seconds. In what follows, we introduce our efficient algorithm to solve this large-scale optimization problem.

5 JOINT OPTIMIZATION ALGORITHM

In this section, we present an efficient joint training algorithm for our optimization framework. Our algorithm is based on the well-known block coordinate descent (BCD) method. In Section 5.1, we provide a brief introduction to the block coordinate descent. In Section 5.2, we present details on how to apply BCD to our optimization framework (5), as well as the convergence properties of our algorithm. In Section 5.3, we compare our algorithm with the off-the-shelf solver SCS [46] and demonstrate the significant speedups of our algorithm.

5.1 Block Coordinate Descent

We apply a cyclic block coordinate descent (BCD) type method [51] for (5). Cyclic BCD is used to minimize a function of the form

$$\Phi(x_1, \ldots, x_d) = \varphi(x_1, \ldots, x_d) + \sum_{i=1}^{d} h_i(x_i), \tag{6}$$

where φ is a smooth function and h_i can be potentially nondifferentiable[3]. In brief, the BCD algorithm updates x_i (keeping other x_j's

[1] If not, we apply a PSD projection step on the final estimate

[2] Here, W denotes sloWly varying, and P denote sParsely varying

[3] Note that the separability of the nonsmooth part (h_i's) is a key to the global convergence given the convexity of the objective function.

fixed), by minimizing over it. Formally,

$$x_i \leftarrow \arg\min_y \Phi(x_{<i}, y, x_{>i}\}), \text{ for } i = 1, \ldots, d, \tag{7}$$

where $x_{<i}$ and $x_{>i}$ are the collections of x_j's with $j < i$ and $j > i$, respectively. It updates the coordinates in a cyclic fashion.

BCD-type methods are widely used for solving huge-scale optimization problems in statistical learning, especially those problems with sparsity structure, due to their inexpensive iteration updates and capability of exploiting problem structure. For example, they have been used to solve the lasso problem [25], the support vector machines [7], and the graphical lasso [43], and other problems with structured sparsity [29], etc. The convergence guarantees of BCD applied to (6) can be found in [30, 51].

5.2 Algorithm Details

Algorithm 1 Algorithm for optimizing (5)

Input: Data $S_k, k \in [m]$. Hyperparameters $\alpha, \beta, \gamma, \bar{r}$; Initialization: $L_k = L_k^{(0)}, R_k = R_k^{(0)}$ for all k

1: **for** $\ell = 0, 1, 2, \ldots$ **do**
2: **for** $k = 0, 1 \ldots, m$ **do**
3: $L_k \leftarrow \arg\min_X F(\{\mathcal{L}_{<k}, X, \mathcal{L}_{>k}\}, \mathcal{R}\})$
4: $R_k \leftarrow \arg\min_X F(\mathcal{L}, \{\mathcal{R}_{<k}, X, \mathcal{R}_{>k}\})$ ((WLR) and (WLWR) only)
5: **end for**
6: $\mathcal{R} \leftarrow \arg\min_{\mathcal{X}} F(\mathcal{L}, \mathcal{X})$ ((WLPR) only)
7: **end for**

We first present the BCD algorithm for solving (5) in Algorithm 1, and show how we can treat (5) with 3 different metrics (WLR), (WLWR) and (WLPR) as special cases of (6)—we also discuss computation of (7) resulting in Algorithm 1. We discuss efficient ways to compute the updates in lines 3, 4 and 6 in Algorithm 1. Finally, we provide the convergence property of our algorithm.

5.2.1 Algorithm. The formal algorithm to solve (5) is presented in Algorithm 1. Note that the update in line 4 is for (WLR) and (WLWR) exclusively; while the update in line 6 is for (WLPR) exclusively.

For (5) with (WLR) and (WLWR), the corresponding discrepancy metrics $\mathcal{D}_1$ and $\mathcal{D}_2$ are smooth wrt L_k and R_k. We can view (5) as an instance of (6) with $L_1, R_1, \ldots, L_m, R_m$ as $2m$ blocks. Here, $\varphi(\mathcal{L}, \mathcal{R}) = \frac{1}{2}\sum_{k=1}^m \|L_k + R_k - S_k\|_F^2 + \sum_{k=1}^{m-1} \mathcal{D}((L_{k+1}, R_{k+1}), (L_k, R_k))$, $h_{L,k}(L_k) = \rho_L(L_k)$, and $h_{R,k}(R_k) = \rho_R(R_k)$. Applying (7) yields the updates in lines 3 and 4 of Algorithm 1.

For (WLPR), the discrepancy metric $\mathcal{D}_3$ is smooth wrt L_k, but nonsmooth wrt R_k, so we need to treat $\mathcal{R}$ as a whole block, i.e. (5) is a special case of (6) with $L_1, \ldots, L_m, \mathcal{R}$ as $m + 1$ blocks. Here, $\varphi(\mathcal{L}, \mathcal{R}) = \frac{1}{2}\sum_{k=1}^m \|L_k + R_k - S_k\|_F^2 + \frac{\gamma}{2}\sum_{k=1}^{m-1} \|L_{k+1} - L_k\|_F^2$, $h_{L,k}(L_k) = \rho_L(L_k)$ and $h_{\mathcal{R}}(\mathcal{R}) = \sum_{k=1}^m \rho_R(R_k) + \rho\sum_{k=1}^{m-1} \|R_{k+1} - R_k\|_{1,\text{off}}$. Applying (7) results in the updates in lines 3 and 6 of Algorithm 1.

5.2.2 Coordinate updates. All the minimization sub-problems in lines 3, 4 and 6 of Algorithm 1 can be solved very efficiently by either closed-form expressions or dynamic programming. To this end, we first introduce two optimization problems with closed-form expressions, and then show how updates in lines 3 and 4 can be reduced to these two operators.

The first one is the well-known soft-thresholding operator [12]:

$$\mathcal{T}_1(\tilde{x}; \lambda) := \arg\min_y \frac{1}{2}(y - \tilde{x})^2 + \lambda|y| = \text{sign}(\tilde{x})(|\tilde{x}| - \lambda)_+. \tag{8}$$

The other operator $\mathcal{T}_*(\tilde{X}; \lambda, \bar{r}) : \mathbb{S}^p \to \mathbb{S}_+^p$ is defined as

$$\mathcal{T}_*(\tilde{X}; \lambda, \bar{r}) := \arg\min_{Y \in \mathbb{S}_+^p} \frac{1}{2}\|Y - \tilde{X}\|_F^2 + \lambda \,\text{tr}(Y), \text{ s.t. } \text{rank}(Y) \le \bar{r}. \tag{9}$$

Given the eigenvalue decomposition of $\tilde{X} = U\tilde{D}U^\top$, where $D = \text{diag}(\tilde{d}_1, \ldots, \tilde{d}_p)$ with $\tilde{d}_1 \ge \ldots \ge \tilde{d}_p$, the solution to (9) is given by $\mathcal{T}_*(\tilde{X}; \lambda, \bar{r}) = UD^*U^\top$, where $D^* = \text{diag}(d_1^*, \ldots, d_p^*)$ with $d_i^* = (\tilde{d}_i - \lambda)_+, \ \forall i \le r$ and $d_i^* = 0, \ \forall i > r$. The derivation of this closed-form solution can be found in [44].

Now we show how to use these two operators to obtain the updates in lines 3 and 4. For simplicity, we will look into the updates for (WLWR) and $k \notin \{0, m\}$. In this case, by simple linear algebra, it is easy to see that the update in line 3 is equivalent to

$$\begin{aligned} L_k &= \arg\min_{X \succeq 0} \frac{1 + 2\gamma}{2}\|X - \tilde{L}_k\|_F^2 + \alpha \,\text{tr}(X), \text{ s.t. } \text{rank}(X) \le \bar{r} \\ &= \mathcal{T}_*(\tilde{L}_k; \frac{\alpha}{1 + 2\gamma}, \bar{r}), \end{aligned} \tag{10}$$

where $\tilde{L}_k = \frac{1}{1+2\gamma}\Big[(S_k - R_k) + \gamma L_{k+1} + \gamma L_{k-1}\Big]$. The update in line 4 is equivalent to

$$R_k = \arg\min_{X \in \mathbb{S}^p} \frac{1 + 2\gamma}{2}\|X - \tilde{R}_k\|_F^2 + \beta\|X\|_{1,\text{off}}, \tag{11}$$

where $\tilde{R}_k = \frac{1}{1+2\gamma}\Big[(S_k - L_k) + \gamma R_{k+1} + \gamma R_{k-1}\Big]$. Since $\|X\|_{1,\text{off}}$ is separable wrt each component of X, the minimization problem (11) can be solved separately over each component, and each subproblem is in the form of (8). Thus, the solution to (11) is given by X^*, with

$$X_{ij}^* = X_{ji}^* = \mathcal{T}_1(\tilde{R}_{k,ij}; \frac{\beta}{1 + 2\gamma}), \ \forall i < j; \ X_{ii}^* = \tilde{R}_{k,ii}, \ \forall i \in [p].$$

For special cases where $k \in \{0, m\}$ as well as (WLR), the updates are similar with simple modifications.

For (WLPR), the update in line 3 is same as that for (WLR); the update in line 6 is equivalent to

$$\begin{aligned} \mathcal{R} = \arg\min_{\mathcal{X}} \frac{1}{2}\sum_{k=1}^m \|X_k + L_k - S_k\|_F^2 + \sum_{k=1}^m \beta\|X_k\|_{1,\text{off}} \\ + \sum_{k=1}^{m-1} \rho\|X_{k+1} - X_k\|_{1,\text{off}}. \end{aligned} \tag{12}$$

For $i \le j$ and any $\{Y_k\}_1^m$, let $Y_{\cdot,ij}$ denote the vector of $\{Y_{k,ij}\}_{k=1}^m$, then the objective of (12) can be written as $\sum_{i=1}^p g_{ii}(X_{\cdot,ii}) + 2\sum_{i<j} g_{ij}(X_{\cdot,ij})$, where

$$g_{ii}(x) = \frac{1}{2}\|x + L_{\cdot,ii} - S_{\cdot,ii}\|^2 \tag{13}$$

$$g_{ij}(x) = \frac{1}{2}\|x + L_{\cdot,ij} - S_{\cdot,ij}\|^2 + \beta\|x\|_1 + \rho\sum_{k=1}^{m-1} |x_{k+1} - x_k|, \tag{14}$$

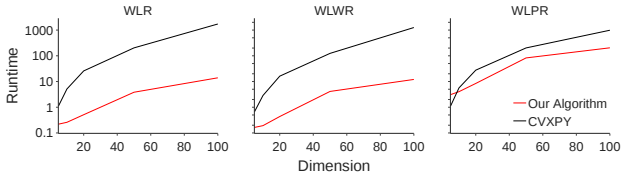

Figure 1: Mean runtime, in seconds, for $m = 5$. For every dimension value, we sample 20 random subsets of companies and hyper-parameters. Y-axis in log scale.

which can be solved separately. Solving (13) is straightforward; (14) is a fused lasso type problem [49] that can be solved efficiently in $O(m)$ by dynamic programming [34].

5.2.3 Convergence property. The following theorem states the convergence property of Algorithm 1:

Theorem 5.1. *Let $(\mathcal{L}^{(\ell)}, \mathcal{R}^{(\ell)})$ be the ℓ-th iterate of $(\mathcal{L}, \mathcal{R})$ given by Algorithm 1, then we have $F(\mathcal{L}^{(\ell+1)}, \mathcal{R}^{(\ell+1)}) \leq F(\mathcal{L}^{(\ell)}, \mathcal{R}^{(\ell)})$. When $\bar{r} = p$, i.e. there is no rank constraint, then the problem becomes convex, and the following holds: (i) any limiting point of $\{(\mathcal{L}^{(\ell)}, \mathcal{R}^{(\ell)})\}_1^\infty$ is a global minimizer of the problem; (ii) there exists a constant $C > 0$, such that $F(\mathcal{L}^{(\ell)}, \mathcal{R}^{(\ell)}) - F^* \leq \frac{C}{\ell}$, where F^* is the optimal value of* (5)*. All the statements above hold for all regularizers* (WLR)*,* (WLWR) *and* (WLPR)*.*

5.3 Comparison with off-the-shelf solver

Since our optimization framework is new, there is no existing specialized algorithm to solve it. Therefore, we compare our algorithm with state-of-the-art conic optimization solver SCS (Splitting Conic Solver, [46]). SCS utilizes the homogeneous self-dual reformulation and solves the embedding with operator splitting methods and it is a widely used optimization package to solve large-scale convex quadratic cone problems[4].

We run both algorithms on a returns dataset of companies in S&P500 with various sizes $20 \leq p \leq 100$ and a fixed number of periods $m = 5$; see Section 6.1 for more details on the return dataset. For each regularizer and value of p, we draw a random subset of S&P500 companies, construct the sample covariance of their returns for 5 consecutive quarters, and run both algorithms on a random set of hyperparameters. We repeat this procedure 20 times and report the mean runtime in Figure 1. We run both algorithms with the same convergence tolerance and the same maximum number of iterations[5].

As Figure 1 shows, our algorithm is significantly faster than SCS, specially for large problem sizes. In addition, our algorithm can address the optimization problem with rank constraints, which makes the problem non-convex and therefore out-of-reach of off-the-shelf convex solvers.

6 NUMERICAL EXPERIMENTS

In this section, we present a few numerical experiments to show the strength of our joint optimization framework compared to some existing covariance estimation methods. In Section 6.1, we compare the estimation errors of our estimators with those from a latent-factor stochastic-volatility (SV) model [36] (with implementation in [31]) on both synthetic and real datasets. In Section 6.2, we look into a specific downstream task—portfolio optimization. We compare portfolios obtained using our estimators with those obtained using the sample covariance matrix, as well as the well-known Ledoit-Wolfe shrinkage estimator [40].

6.1 Estimation errors

We start by looking at estimation errors of different methods on both synthetic and real datasets. We specifically compare our estimators to those obtained from a latent-factor SV model (proposed in [36] and implemented in [31])[6], using global-local shrinkage prior on factor loading coefficients as proposed in [35], as well as a standard Gaussian prior. Row-wise NG (resp. Col-wise NG) refers to a row-wise (resp. col-wise) Normal-Gamma prior on the factor loading matrix. For each prior, we compare two versions, one where we assume time-varying variances for factors and idiosyncratic errors (SV) and one where we assume these variances to be constant.

Datasets. We run our comparisons on both synthetic and real datasets: *The synthetic dataset* is made of randomly generated time series from a stochastic volatility model using random factors and loading parameters[7]. For different values of the number of factors and the number of variables in the time series, we generate 100 such time series with 1300 observations (roughly 5 years of financial data). *The real dataset* is composed of stock returns of companies that are part of the S&P500 between 2013 to 2019, which results in a time series of 453 variables. We randomly choose 100 starting dates from this period and take the first 1300 observations following each starting date.

Performance metric. We split each dataset into two parts with 650 observations each. The first part is used to find the best hyperparameters for our model by training the algorithm on the first 520 observations and measuring the mean-square-error (MSE) between our last estimated covariance matrix and the target covariance matrix for the next 130. After finding the best set of hyper-parameters for this first MSE, we use the second part to evaluate the performances of each estimator by training again on the first 520 observations and taking the MSE relative to the target matrix. This somewhat involved procedure is necessary to predict the covariance matrix on a period that immediately follows the training one.

For a predicted matrix $\hat{\Sigma}$ and a target matrix $\bar{\Sigma}$, we measure the MSE as $\|\hat{\Sigma} - \bar{\Sigma}\|_F^2/p^2$. The target covariance matrix is computed from the original factor model on synthetic datasets, whereas on the real dataset, is taken to be the sample covariance matrix on out-of-sample test observations.

Results. As shown in Table 1, the MSE of our estimators is slightly bigger than that of the best SV model for a small number of variables and factors, but as those two numbers grow, our models exhibit a better prediction with a smaller MSE. Looking at the runtime figures (Table 1), we also see that our models are significantly faster

[4]Since the SCS requires the problem to be convex, we consider the special case where the rank restriction is abundant ($\bar{r} = p$).
[5]We use the dynamic programming algorithm for Fused Lasso in [34] as implemented in https://www.stat.cmu.edu/~ryantibs/convexopt-F13/homeworks/prox_R.cpp for updating $\mathcal{R}$ in Algorithm 1
[6]The SV model generates multiple covariance matrices at each time point from the estimated distribution. To obtain a single covariance matrix, we average out the estimators obtained for the last observation of the time series across 1000 draws
[7]We use the function `fsvsim` in [31] with default arguments for this

Table 1: Mean MSE and runtime (in seconds) of different covariance estimators with synthetic data generated with 3 factors.

	MSE				Runtime			
Estimator/ # variables	10	30	100	300	10	30	100	300
Col-wise NG	**0.214**	0.151	0.161	0.262	10.104	21.103	64.909	266.13
Col-wise NG (SV)	0.454	0.197	0.226	0.306	19.011	45.03	146.258	510.541
Row-wise NG	0.217	**0.149**	0.161	0.262	10.156	20.39	66.788	268.46
Row-wise NG (SV)	0.462	0.205	0.233	0.299	18.762	44.983	147.553	522.967
Standard Gaussian	0.476	0.566	0.709	0.806	10.621	20.337	68.237	263.249
Standard Gaussian (SV)	0.693	0.413	0.455	0.287	17.273	43.063	153.165	520.162
WLPR **(this paper)**	0.246	0.196	0.138	**0.168**	0.615	0.895	7.176	87.952
WLWR **(this paper)**	0.244	0.196	**0.135**	0.17	**0.09**	**0.521**	**2.745**	**16.412**

and more accurate than the SV ones. These differences are also present when applied to real financial data (we omit these results due to space constraints).

6.2 Application to portfolio optimization

Portfolio optimization refers to a classical asset management problem in which the practitioner constructs a portfolio that optimizes some objective value, usually combining two goals: maximizing expected returns and minimizing their variance. The framework was first introduced by Markowitz [42] and has since then become a topic of many studies. To measure the quality of our estimated covariance matrices, we focus on the minimum variance portfolio problem, because it does not require estimating the expected returns, a problem beyond the scope of this paper. Specifically, we are interested in the following optimization problem:

$$\min_{w} \; w^\top \hat{\Sigma} w, \qquad \text{s.t.} \quad 1^\top w = 1, \; w \geq 0, \tag{15}$$

where $\hat{\Sigma}$ is an estimate of the covariance of returns. Using the sample covariance matrix as $\hat{\Sigma}$ may be the most obvious choice, but this has a lot of drawbacks in terms of the actual performance of the selected portfolio [40, 50]. There has been therefore a great amount of work to improve the estimation of the covariance matrix, and specifically for the use case of portfolio optimization. In the following, we compare portfolios obtained by our algorithms to the ones obtained using the sample covariance matrix, as well as other sophisticated covariance matrix estimators.

Metrics. We use two metrics to measure the quality of a portfolio w, both used in [50] and both based on the predicted volatility in-sample $\sigma^2[w] = w^\top S^{train} w$ and the realized volatility out-of-sample $\hat{\sigma}^2[w] = w^\top S^{test} w$ (S^{train} and S^{test} are the sample covariance matrices on the training period and testing period, and w is built based on S^{train} alone).

The first metric R_σ, defined as $|\sigma[w] - \hat{\sigma}[w]|/\sigma[w]$, measures the reliability of the portfolio in predicting the true realized volatility out-of-sample from the predicted volatility in-sample. A small R_σ value means that the portfolio's out-of-sample risk is close to its in-sample risk. *The second metric* we use is simply the realized volatility $\hat{\sigma}[w]$ out-of-sample, which is a direct measure of the portfolio's risk and is the metric that the optimization problem is trying to minimize.

This difference between the estimated volatility in sample and that realized out-of-sample can be explained by at least two factors. The first one is due to the statistical uncertainty and the presence of noise, specially when using a small number of observations. In order to mitigate this effect, one possible solution is to simply increase the sample size and use more observations. And while this is not always possible, it also assumes that the correlation structures are stationary. Our estimators try to find a balance between these two effects: Use regularization to mitigate noise (and low-sample sizes), and allow for correlation structures to change through time.

Table 2: Reliability and volatility statistics for the different estimators on 2 datasets. The smallest value for each metric is highlighted in bold.

	Stock returns		Currency rates	
Estimator	Reliability	Volatility	Reliability	Volatility
SAMPLE	1.318 (±0.358)	10.417 (±0.752)	6.315 (±2.678)	0.089 (±0.033)
SAMPLE_AP	0.475 (±0.152)	9.44 (±0.627)	4.355 (±2.027)	0.068 (±0.027)
LDWF	1.043 (±0.28)	10.113 (±0.77)	4.817 (±2.043)	0.091 (±0.034)
LDWF_AP	0.475 (±0.152)	9.401 (±0.632)	3.74 (±1.725)	0.084 (±0.029)
WLR	0.417 (±0.1)	11.54 (±0.805)	1.018 (±0.421)	0.086 (±0.031)
WLWR	0.316 (±0.084)	9.42 (±0.645)	**0.423 (±0.1)**	0.025 (±0.005)
WLPR	**0.304 (±0.078)**	**9.397 (±0.634)**	6.262 (±2.749)	**0.018 (±0.003)**

Datasets and baselines. We run our comparisons on two types of datasets: Stock returns of the S&P500, which we already introduced in Section 6.1 and currency rates (from 2013 to 2020) with $p = 151$[8].

We compare our 3 models to the sample covariance matrix as well as the Ledoit-Wolf shrinkage estimator [40]. This estimator has been shown to offer enhanced performance both in terms of realized volatility and reliability. Moreover, for each of these two baselines, we compute 2 covariance matrices: one using only the last training period (*SAMPLE* and *LDWF*) and one using data from all 4 periods (*SAMPLE_AP* and *LDWF_AP*). In the following, we drop the SV model as it is expensive to compute and delivers point-wise estimates.

Evaluation procedure. For both datasets, we use 4 quarters to build each estimator and use the next one to test the estimated portfolios (so the portfolios are rebalanced every quarter). Our models learn 4 covariance matrices, one for each quarter, but only the last one is used to build the minimum variance portfolio. Using a sliding window on the quarters, we obtain 24 data points for the returns dataset and 28 on the currency rates one. In order to find the best hyperparameters[9] for our estimators, we use the first 3 windows as validation set and take, for each metric, the set of hyperparameters with the best average score across these 3 windows.

Results. For the stock returns, as shown in Tables 2 and **??**, our estimators offer better portfolios both in terms of reliability and realized volatility. In addition, both *WLWR* and *WLPR* out-perform all other estimators for specific datasets and metrics, which highlights the impact of the different discrepancy penalties in modelling different behaviours. *WLPR* out-performs all 4 baselines on the returns dataset in 58% of the data points and an average reliability that is reduced by 36% relative to the best baseline. In terms of realized volatility, *WLWR* has the best estimated portfolios in 33% of the periods, which is the highest rate between all the estimators we use (including baselines). *LDWF_AP* has the closer performance for realized volatility, with an average of 9.401 (9.397 for *WLWR*).

[8] Data obtained from https://openexchangerates.org/

[9] We fix $\alpha = 0$ and use a grid of 20 values in $[10^{-3}, 10^4] \bigcup \{0, \infty\}$ for γ, β, ρ and 7 values in [2,15] for the rank of L_k

As for the currency rates, (WLWR) has the best reliability, and out-performs the 4 baselines in 43% of the times, with an average improvement of 3.317 relative to the best baseline.

In terms of realized volatility $\hat{\sigma}$, (WLPR) provides the least risky portfolio in 44% of the cases, with both (WLWR) and (WLPR) showing significant improvements compared to all baselines.

7 CONCLUSION

In summary, we propose a joint optimization framework for simultaneously learning covariance matrices over periods under minimal structural assumptions on covariances and their evolution pattern. The large-scale optimization problem can be efficiently solved by our proposed BCD type algorithm, leading to smaller estimation errors and better downstream portfolio optimization performances.

ACKNOWLEDGMENTS

This work is supported in part by MIT-IBM Watson AI Lab a research collaboration as part of the IBM AI Horizons Network. Rahul Mazumder also acknowledges partial funding support from the Office of Naval Research. The views and conclusions are those of the authors and should not be interpreted as representing the official policies of the funding agencies or the government.

REFERENCES

[1] Peter J Bickel and Elizaveta Levina. 2008. Covariance regularization by thresholding. *The Annals of Statistics* 36, 6 (2008), 2577–2604.
[2] Kevin Bleakley and Jean-Philippe Vert. 2011. The group fused lasso for multiple change-point detection. *arXiv preprint arXiv:1106.4199* (2011).
[3] Tim Bollerslev. 1986. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics* 31, 3 (1986), 307–327.
[4] Tim Bollerslev, Robert F Engle, and Jeffrey M Wooldridge. 1988. A capital asset pricing model with time-varying covariances. *Journal of political Economy* 96, 1 (1988), 116–131.
[5] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. 2011. Robust principal component analysis? *Journal of the ACM (JACM)* 58, 3 (2011), 1–37.
[6] Venkat Chandrasekaran, Pablo A Parrilo, and Alan S Willsky. 2010. Latent variable graphical model selection via convex optimization. In *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 1610–1613.
[7] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)* 2, 3 (2011), 1–27.
[8] Jia Chen, Degui Li, and Oliver Linton. 2019. A new semiparametric estimation approach for large dynamic covariance matrices with multiple conditioning variables. *Journal of Econometrics* 212, 1 (2019), 155–176.
[9] Xiaohui Chen, Mengyu Xu, and Wei Biao Wu. 2013. Covariance and precision matrix estimation for high-dimensional time series. *The Annals of Statistics* 41, 6 (2013), 2994–3021.
[10] Ziqi Chen and Chenlei Leng. 2016. Dynamic covariance models. *J. Amer. Statist. Assoc.* 111, 515 (2016), 1196–1207.
[11] Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)* 39, 1 (1977), 1–22.
[12] David L Donoho. 1995. De-noising by soft-thresholding. *IEEE transactions on information theory* 41, 3 (1995), 613–627.
[13] Robert Engle. 2002. Dynamic conditional correlation: A simple class of multivariate generalized autoregressive conditional heteroskedasticity models. *Journal of Business & Economic Statistics* 20, 3 (2002), 339–350.
[14] Robert F Engle. 1982. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica: Journal of the econometric society* (1982), 987–1007.
[15] Robert F Engle and Kenneth F Kroner. 1995. Multivariate simultaneous generalized ARCH. *Econometric theory* 11, 1 (1995), 122–150.
[16] Robert F Engle, Olivier Ledoit, and Michael Wolf. 2019. Large dynamic covariance matrices. *Journal of Business & Economic Statistics* 37, 2 (2019), 363–375.
[17] Robert F Engle, Victor K Ng, and Michael Rothschild. 1990. Asset pricing with a factor-ARCH covariance structure: Empirical estimates for treasury bills. *Journal of econometrics* 45, 1-2 (1990), 213–237.
[18] Eugene F Fama and Kenneth R French. 1989. Business conditions and expected returns on stocks and bonds. *Journal of financial economics* 25, 1 (1989), 23–49.
[19] Eugene F Fama and Kenneth R French. 1993. Common risk factors in the returns on stocks and bonds. *Journal of financial economics* 33, 1 (1993), 3–56.
[20] Jianqing Fan, Yingying Fan, and Jinchi Lv. 2008. High dimensional covariance matrix estimation using a factor model. *Journal of Econometrics* 147, 1 (2008), 186–197.
[21] Jianqing Fan, Fang Han, and Han Liu. 2014. Challenges of big data analysis. *National science review* 1, 2 (2014), 293–314.
[22] Jianqing Fan, Yuan Liao, and Han Liu. 2016. An overview of the estimation of large covariance and precision matrices. *The Econometrics Journal* 19, 1 (2016), C1–C32.
[23] Jianqing Fan, Yuan Liao, and Martina Mincheva. 2013. Large covariance estimation by thresholding principal orthogonal complements. *Journal of the Royal Statistical Society. Series B, Statistical methodology* 75, 4 (2013).
[24] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2008. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* 9, 3 (2008), 432–441.
[25] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. 2010. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software* 33, 1 (2010), 1.
[26] Andrew Gelman and Donald B Rubin. 1992. Inference from iterative simulation using multiple sequences. *Statistical science* 7, 4 (1992), 457–472.
[27] John Geweke. 1977. The dynamic factor analysis of economic time series. *Latent variables in socio-economic models* (1977).
[28] David Hallac, Youngsuk Park, Stephen Boyd, and Jure Leskovec. 2017. Network inference via the time-varying graphical lasso. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 205–213.
[29] Hussein Hazimeh and Rahul Mazumder. 2020. Fast best subset selection: Coordinate descent and local combinatorial optimization algorithms. *Operations Research* 68, 5 (2020), 1517–1537.
[30] Mingyi Hong, Xiangfeng Wang, Meisam Razaviyayn, and Zhi-Quan Luo. 2017. Iteration complexity analysis of block coordinate descent methods. *Mathematical Programming* 163, 1-2 (2017), 85–114.
[31] Darjus Hosszejni and Gregor Kastner. 2021. Modeling Univariate and Multivariate Stochastic Volatility in R with stochvol and factorstochvol. *Journal of Statistical Software* 100, 12 (2021), 1–34. https://www.jstatsoft.org/index.php/jss/article/view/v100i12
[32] Shibal Ibrahim, Wenyu Chen, Yada Zhu, Pin-Yu Chen, Yang Zhang, and Rahul Mazumder. 2021. Knowledge Graph Guided Simultaneous Forecasting and Network Learning for Multivariate Financial Time Series. In *2021 KDD Workshop on Machine Learning in Finance*.
[33] Eric Jacquier, Nicholas G Polson, and Peter E Rossi. 2002. Bayesian analysis of stochastic volatility models. *Journal of Business & Economic Statistics* 20, 1 (2002), 69–87.
[34] Nicholas A Johnson. 2013. A dynamic programming algorithm for the fused lasso and l 0-segmentation. *Journal of Computational and Graphical Statistics* 22, 2 (2013), 246–260.
[35] Gregor Kastner. 2019. Sparse Bayesian time-varying covariance estimation in many dimensions. *Journal of Econometrics* 210, 1 (2019), 98–115. Annals Issue in Honor of John Geweke "Complexity and Big Data in Economics and Finance: Recent Developments from a Bayesian Perspective".
[36] Gregor Kastner, Sylvia Frühwirth-Schnatter, and Hedibert Freitas Lopes. 2017. Efficient Bayesian inference for multivariate factor stochastic volatility models. *Journal of Computational and Graphical Statistics* 26, 4 (2017), 905–917.
[37] Yuan Ke, Heng Lian, and Wenyang Zhang. 2022. High-dimensional dynamic covariance matrices with homogeneous structure. *Journal of Business & Economic Statistics* 40, 1 (2022), 96–110.
[38] Laurent Laloux, Pierre Cizeau, Marc Potters, and Jean-Philippe Bouchaud. 2000. RANDOM MATRIX THEORY AND FINANCIAL CORRELATIONS. *International Journal of Theoretical and Applied Finance* 03 (2000), 391–397.
[39] Shiwei Lan, Andrew Holbrook, Gabriel A Elias, Norbert J Fortin, Hernando Ombao, and Babak Shahbaba. 2020. Flexible bayesian dynamic modeling of correlation and covariance matrices. *Bayesian analysis* 15, 4 (2020), 1199.
[40] Olivier Ledoit and Michael Wolf. 2004. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of multivariate analysis* 88, 2 (2004), 365–411.
[41] Degui Li. 2021. Estimation of Large Dynamic Covariance Matrices: A Selective Review. *Econometrics and Statistics* (2021).
[42] Harry Markowitz. 1952. Portfolio Selection. *The Journal of Finance* 7, 1 (1952), 77–91.
[43] Rahul Mazumder and Trevor Hastie. 2012. The graphical lasso: New insights and alternatives. *Electronic journal of statistics* 6 (2012), 2125.
[44] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. 2010. Spectral regularization algorithms for learning large incomplete matrices. *The Journal of Machine Learning Research* 11 (2010), 2287–2322.
[45] Rahul Mazumder, Peter Radchenko, and Antoine Dedieu. 2017. Subset selection with shrinkage: Sparse linear modeling when the SNR is low. *arXiv preprint*

arXiv:1708.03288 (2017).

[46] Brendan O'Donoghue, Eric Chu, Neal Parikh, and Stephen Boyd. 2016. Conic Optimization via Operator Splitting and Homogeneous Self-Dual Embedding. *Journal of Optimization Theory and Applications* 169, 3 (June 2016), 1042–1068.

[47] Henrik Ohlsson, Lennart Ljung, and Stephen Boyd. 2010. Segmentation of ARX-models using sum-of-norms regularization. *Automatica* 46, 6 (2010), 1107–1111.

[48] James H Stock and Mark Watson. 2011. Dynamic factor models. *Oxford Handbooks Online* (2011).

[49] Robert Tibshirani, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. 2005. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67, 1 (2005), 91–108.

[50] Vincenzo Tola, Fabrizio Lillo, Mauro Gallegati, and Rosario N. Mantegna. 2008. Cluster analysis for portfolio optimization. *Journal of Economic Dynamics and Control* 32, 1 (2008), 235–258. Applications of statistical physics in economics and finance.

[51] Paul Tseng. 2001. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of optimization theory and applications* 109, 3 (2001), 475–494.

[52] Hanchao Wang, Bin Peng, Degui Li, and Chenlei Leng. 2021. Nonparametric estimation of large covariance matrices with conditional sparsity. *Journal of Econometrics* 223, 1 (2021), 53–72.

[53] Matt Wytock. 2014. Time-varying Linear Regression with Total Variation Regularization. (2014).