Applications of the Alternating Direction Method of Multipliers to the Semidefinite Inverse Quadratic Eigenvalue Problem with Partial Eigenstructure

Zhengjian Bai * Meixiang Chen† Xiaoming Yuan‡

April 21, 2013

Abstract

This paper shows that the alternating direction method of multipliers (ADMM) is efficient for solving the semidefinite inverse quadratic eigenvalue problem (SDIQEP) with partial eigenstructure. We derive several ADMM-based iterative schemes for SDIQEP, and demonstrate their efficiency for large-scale cases of SDIQEP numerically.

Keywords. Inverse quadratic eigenvalue problem, alternating direction method of multipliers, partial eigenstructure, large scale

AMS subject classifications. 65F10, 65F15, 65F18, 90C25

1 Introduction

This paper is concerned with the semidefinite inverse quadratic eigenvalue problem (SDIQEP for short) with partial eigenstructure. That is, with a given original pencil:

$$P_a(\lambda) := \lambda^2 M_a + \lambda C_a + K_a \quad \forall \lambda \in \mathbb{C}, \tag{1}$$

where M_a, C_a and $K_a \in \mathbb{R}^{n\times n}$ are all symmetric; and a measured partial eigendata $(X, \Lambda) \in \mathbb{R}^{n\times p} \times \mathbb{R}^{p\times p}$ $(p \le n)$ where X is assumed to be a full column rank matrix and

$$\Lambda = \mathrm{diag}(\lambda_1^{[2]}, \ldots, \lambda_s^{[2]}, \lambda_{2s+1}, \ldots, \lambda_p) \tag{2}$$

*Author to whom any correspondence should be addressed. School of Mathematical Sciences, Xiamen University, Xiamen 361005, China (zjbai@xmu.edu.cn). The research of this author was partially supported by the National Natural Science Foundation of China grant 11271308, the Natural Science Foundation of Fujian Province of China for Distinguished Young Scholars (No. 2010J06002), and NCET.

†School of Mathematical Sciences, Huaqiao University, Quanzhou 362021, China, and School of Mathematical Sciences, Xiamen University, Xiamen 361005, China (mxchen@hqu.edu.cn).

‡Department of Mathematics, Hong Kong Baptist University, Hong Kong, China (xmyuan@hkbu.edu.hk). The research of this author was supported by the General Research Fund from Hong Kong Research Grants Council: 203712.

with

$$\lambda_k^{[2]} = \begin{bmatrix} \alpha_k & \beta_k \\ -\beta_k & \alpha_k \end{bmatrix} \in \mathbb{R}^{2\times 2}, \quad \beta_k \neq 0 \text{ for } k = 1, \ldots, s \quad \text{and} \quad \lambda_k \neq 0 \text{ for } k = 2s+1, \ldots, p;$$

we seek three real symmetric matrices $M, C, K \in \mathbb{R}^{n\times n}$ such that they solve the following least-squares problem subject to a linear and some semidefinite constraints:

$$\begin{aligned} \min \quad & \tfrac{c_1}{2}\|M - M_a\|_F^2 + \tfrac{c_2}{2}\|C - C_a\|_F^2 + \tfrac{1}{2}\|K - K_a\|_F^2 \\ \text{s.t.} \quad & MX\Lambda^2 + CX\Lambda + KX = 0, \\ & M \in \mathcal{S}_+^n, \quad C \in \mathcal{S}^n, \quad K \in \mathcal{S}_+^n, \end{aligned} \tag{3}$$

where $\|\cdot\|_F$ denotes the Frobenius matrix norm, $c_1, c_2 > 0$ are two weighting parameters, $\mathcal{S}^n$ and $\mathcal{S}_+^n$ are the sets of all real $n \times n$ symmetric matrices and of all real $n \times n$ symmetric positive semidefinite matrices, respectively.

SDIQEP arises in many applications such as structural dynamics and acoustics, circuit analysis, applied physics, vibration theory, and the finite element model updating in PDEs. We refer to nice monographs [7, 10, 13] and references therein for intensive study on theoretical properties and algorithmic aspects of SDIQEP models. In particular, the original given pencil (1) is often obtained via finite element techniques in many applications where nonzero entries of the analytic matrices M_a, C_a, K_a denote the physical mass, damping, and stiffness parameters, respectively. Thus, the matrices M_a, C_a, K_a often possess exploitable structure properties such as symmetry, definiteness, sparsity/bandedness, etc.

Aiming at generating a updated pencil $P(\lambda) := \lambda^2 M + \lambda C + K$ such that it coincides with the measured partial eigendata (X, Λ) while is closest to the original pencil (1) in sense of least squares, the updated matrices (M, C, K) via solving SDIQEP obviously are required to preserve structural properties of the analytic matrices M_a, C_a, K_a. In this paper, we mainly focus on the semidefiniteness requirement (see for instance the applications mentioned in [7, 10]). An interesting question is how to solve SDIQEP with the updated matrices (M, C, K) preserving semidefiniteness and sparsity simultaneously, which needs further study.

As in [1], let the QR factorization of X be given by

$$X = Q \begin{bmatrix} R \\ 0 \end{bmatrix},$$

where Q is an orthogonal matrix in order of n and R is a $p \times p$ nonsingular upper triangular matrix. By renaming $M := Q^T M Q$, $C := Q^T C Q$, $K := Q^T K Q$, $M_a := Q^T M_a Q$, $C_a := Q^T C_a Q$, and $K_a := Q^T K_a Q$, the model (3) of SDIQEP can be rewritten as

$$\begin{aligned} \min \quad & \tfrac{c_1}{2}\|M - M_a\|_F^2 + \tfrac{c_2}{2}\|C - C_a\|_F^2 + \tfrac{1}{2}\|K - K_a\|_F^2 \\ \text{s.t.} \quad & M \begin{bmatrix} R \\ 0 \end{bmatrix} \Lambda^2 + C \begin{bmatrix} R \\ 0 \end{bmatrix} \Lambda + K \begin{bmatrix} R \\ 0 \end{bmatrix} = 0, \\ & M \in \mathcal{S}_+^n, \quad C \in \mathcal{S}^n, \quad K \in \mathcal{S}_+^n. \end{aligned} \tag{4}$$

In what follows, we refer to (4) as the model of SDIQEP.

The semidefiniteness constraints on M and K in the model (4) increase the difficulty of solving (4) numerically due to its mixture with additional linear constraints. Another difficulty of solving (4) often is due to the high dimensionality of its variables. For instance, the problem size n is often large (say, $n \geq 1,000$ or even larger). Meanwhile, np, the number of equation constraints becomes also large even when p itself is small (say, when $p = 30$ but $n \geq 1,000$, then $np \geq 30,000$). This difficulty of dimensionality excludes efficient applications of interior-point or Newton-like algorithms for (4), for which solving a system of equations whose dimensionality is proportional to np is unavoidable at each iteration. For existing algorithms for SDIQEP, we refer to [24] for an algorithm for solving a relaxed version of SDIQEP where the semidefiniteness constraints on M and K in (3) are removed from consideration; [1] for a semismooth generalized Newton method which is capable of solving large-scale cases, and [25] for some semidefinite programming techniques, which may not be so effective for handling large-scale problems.

In this paper, we propose to apply the alternating direction method of multipliers (ADMM for short) proposed originally in [14] (see also [12]) to solving SDIQEP. An old method firstly appeared in PDE literature, ADMM has recently found remarkable applications in various areas of scientific computing including optimization, image processing and statistical learning. We refer to [4], a survey paper of ADMM, and references therein for some impressive development on ADMM. Generally speaking, ADMM is a splitting version of the augmented Lagrangian method (ALM) in [22, 28] which is a benchmark for convex programming with linear constraints, where the ALM subproblem at each iteration is decomposed in Gauss-Seidel fashion. This decomposition makes it possible to exploit the properties of different functions in the objective individually and alleviates the subproblems of ALM substantially. As we shall show, the model of SDIQEP (4) is perfectly in the applicable range of ADMM and thus we are inspired to investigate the application of ADMM to SDIQEP. More specifically, we shall demonstrate that the original ADMM in [14], an ADMM-based descent method in [33] and an accelerated (relaxed) ADMM in [6] are all applicable to SDIQEP. Three ADMM-based iterative schemes for SDIQEP are thus proposed. As other efficient applications of ADMM, the main advantage of ADMM for SDIQEP is that the resulting subproblems are easy (some are easy enough to have closed-form solution while the others are standard minimization problems which can be easily solved up to high precisions by existing methods). We shall compare these ADMM schemes numerically with some existing methods including the interior-point approach mentioned in [25] and the generalized Newton approach in [1], for some large scale cases of SDIQEP.

Throughout the paper, we use the following notations. Let A^T and $A^\dagger$ be the transpose and the Moore-Penrose pseudo-inverse of a matrix A, respectively. Let I be the identity matrix of an appropriate dimension.

The rest of this paper is organized as follows. In Section 2 we propose a reformulation of (4) to which ADMM is applicable. We also elaborate how to solve the resulting subproblems when ADMM is applied to solve this reformulation. We derive several ADMM-based iterative schemes for SDIQEP in Section 3, and investigate their convergence in Section 4. In Section 5 we report some numerical results to verify the efficiency of ADMM-based schemes. Finally, in Section 6 we give some concluding remarks.

2 Application of ADMM to SDIQEP

2.1 An ADMM-oriented reformulation

We observe that in the model (4), the objective function is simple in the sense that it is a summation of three separable least-squares terms. As mentioned, a main difficulty of (4) is due to the blend of two types of constraints: the semidefiniteness constraints on M and K, and the linear constraints coupling all matrices M, C and K. On the other hand, we notice that it is easy to minimize one least-squares term with only either the semidefiniteness constraints (via an spectral decomposition) or the linear constraints (via solving a standard quadratic programming problem). These observations thus motivate us to consider a reformulation of (4), to which the resulting subproblems are as easy as the just-mentioned minimization problems when ADMM is applied. The blend of two types of constraints thus can be decoupled by means of an application of ADMM to a reformulation of (4). More specifically, let the space $\mathcal{S}^n$ be equipped with the Frobenius inner product

$$\langle A, B\rangle_F = \mathrm{tr}(A^T B) \quad \forall\, A, B \in \mathcal{S}^n,$$

where "tr" means the trace of a matrix. Moreover, define

$$\Omega := \mathcal{S}^n \times \mathcal{S}^n \times \mathcal{S}^n, \quad \Omega_+ := \mathcal{S}^n_+ \times \mathcal{S}^n \times \mathcal{S}^n_+,$$

$$S_B := \left\{ (Z_1, Z_2, Z_3) \in \Omega : Z_1 \begin{bmatrix} R \\ 0 \end{bmatrix} \Lambda^2 + Z_2 \begin{bmatrix} R \\ 0 \end{bmatrix} \Lambda + Z_3 \begin{bmatrix} R \\ 0 \end{bmatrix} = 0 \right\};$$

let Ω be equipped with the natural inner product

$$\langle (M, C, K), (\widetilde{M}, \widetilde{C}, \widetilde{K})\rangle := \langle M, \widetilde{M}\rangle_F + \langle C, \widetilde{C}\rangle_F + \langle K, \widetilde{K}\rangle_F \quad \forall\, (M, C, K), (\widetilde{M}, \widetilde{C}, \widetilde{K}) \in \Omega$$

and its induced norm be denoted by $\|\cdot\|$. Then the SDIQEP model (4) can be reformulated as

$$\begin{array}{ll} \min & \frac{c_1}{2}\|M - M_a\|_F^2 + \frac{c_2}{2}\|C - C_a\|_F^2 + \frac{1}{2}\|K - K_a\|_F^2 \\ & \quad + \frac{c_1}{2}\|Z_1 - M_a\|_F^2 + \frac{c_2}{2}\|Z_2 - C_a\|_F^2 + \frac{1}{2}\|Z_3 - K_a\|_F^2 \\ \text{s.t.} & (M, C, K) - (Z_1, Z_2, Z_3) = 0, \\ & (M, C, K) \in \Omega_+, \quad (Z_1, Z_2, Z_3) \in S_B. \end{array} \tag{5}$$

In (5), the semidefiniteness and linear constraints in (4) are considered individually in different sets of Ω_+ and S_B, while the new linear constraints in (5) is easy to handle, as we shall show immediately.

Now, (5) is in the standard context of convex minimization problems with separable objective functions and linear constraints. To show how ADMM can be applied to (5), let the augmented Lagrangian function of (5) be given by

$$\begin{array}{ll} & L_\beta(M, C, K, Z_1, Z_2, Z_3, Y_1, Y_2, Y_3) \\ = & \frac{c_1}{2}\|M - M_a\|_F^2 + \frac{c_2}{2}\|C - C_a\|_F^2 + \frac{1}{2}\|K - K_a\|_F^2 \\ & \quad + \frac{c_1}{2}\|Z_1 - M_a\|_F^2 + \frac{c_2}{2}\|Z_2 - C_a\|_F^2 + \frac{1}{2}\|Z_3 - K_a\|_F^2 \\ & \quad - \langle (Y_1, Y_2, Y_3), (M, C, K) - (Z_1, Z_2, Z_3)\rangle + \frac{\beta}{2}\|(M, C, K) - (Z_1, Z_2, Z_3)\|^2, \end{array} \tag{6}$$

where $\beta > 0$ is a penalty parameter on the linear constraints and $(Y_1, Y_2, Y_3) \in \Omega$ is the Lagrange multiplier. When applying ALM in [22, 28] to (5), we obtain the iterative scheme

$$\begin{cases} (M^{k+1}, C^{k+1}, K^{k+1}, Z_1^{k+1}, Z_2^{k+1}, Z_3^{k+1}) \\ \quad = \underset{(M,C,K)\in\Omega_+,\ (Z_1,Z_2,Z_3)\in S_B}{\operatorname{argmin}} \{L_\beta(M, C, K, Z_1, Z_2, Z_3, Y_1^k, Y_2^k, Y_3^k)\}, \\ (Y_1^{k+1}, Y_2^{k+1}, Y_3^{k+1}) = (Y_1^k, Y_2^k, Y_3^k) - \beta[(M^{k+1}, C^{k+1}, K^{k+1}) - (Z_1^{k+1}, Z_2^{k+1}, Z_3^{k+1})], \end{cases} \tag{7}$$

where $(M^k, C^k, K^k, Z_1^k, Z_2^k, Z_3^k, Y_1^k, Y_2^k, Y_3^k)$ is a given iterate. Although it alleviates the difficulty caused by the linear constraints in (5), this straightforward application of ALM ignores completely the nice separable structure in the objective function of (5) and requires to minimize all the variables in one subproblem at each iteration.

A more customized application to (5) is the ADMM in [14], whose resulting scheme splits the minimization subproblem in (7) into two smaller ones (one is (M, C, K)-related and the other is (Z_1, Z_2, Z_3)-related) in Gauss-Seidel fashion:

$$\begin{cases} (M^{k+1}, C^{k+1}, K^{k+1}) = \underset{(M,C,K)\in\Omega_+}{\operatorname{argmin}} \{L_\beta(M, C, K, Z_1^k, Z_2^k, Z_3^k, Y_1^k, Y_2^k, Y_3^k)\}, \\ (Z_1^{k+1}, Z_2^{k+1}, Z_3^{k+1}) = \underset{(Z_1,Z_2,Z_3)\in S_B}{\operatorname{argmin}} \{L_\beta(M^{k+1}, C^{k+1}, K^{k+1}, Z_1, Z_2, Z_3, Y_1^k, Y_2^k, Y_3^k)\}, \\ (Y_1^{k+1}, Y_2^{k+1}, Y_3^{k+1}) = (Y_1^k, Y_2^k, Y_3^k) - \beta[(M^{k+1}, C^{k+1}, K^{k+1}) - (Z_1^{k+1}, Z_2^{k+1}, Z_3^{k+1})], \end{cases} \tag{8}$$

At each iteration, the scheme (8) requires to handel two subproblems which are much easier than the original SDIQEP model (4): one is a simple linear least-squares problem with merely semidefiniteness constraints while the other is a standard quadratic programming problem. Moreover, a further observation on (8) and (6) indicates that the first subproblem with respect to (M, C, K) can be decomposed into three independent smaller subproblems with respect to M, C and K, respectively, which are eligible for parallel computation if parallel infrastructures are available. Thus, we can rewrite the iterative scheme of ADMM for SDIQEP as

$$\begin{cases} M^{k+1} = \underset{M\in\mathcal{S}_+^n}{\operatorname{argmin}}\{\frac{c_1}{2}\|M - M_a\|_F^2 - \langle Y_1, M - Z_1^k\rangle_F + \frac{\beta}{2}\|M - Z_1^k\|_F^2)\}, \\ C^{k+1} = \underset{C\in\mathcal{S}^n}{\operatorname{argmin}}\{\frac{c_2}{2}\|C - C_a\|_F^2 - \langle Y_2, C - Z_2^k\rangle_F + \frac{\beta}{2}\|C - Z_2^k\|_F^2)\}, \\ K^{k+1} = \underset{K\in\mathcal{S}_+^n}{\operatorname{argmin}}\{\frac{1}{2}\|K - K_a\|_F^2 - \langle Y_3, K - Z_3^k\rangle_F + \frac{\beta}{2}\|K - Z_3^k\|_F^2)\}, \\ (Z_1^{k+1}, Z_2^{k+1}, Z_3^{k+1}) = \underset{(Z_1,Z_2,Z_3)\in S_B}{\operatorname{argmin}} \{L_\beta(M^{k+1}, C^{k+1}, K^{k+1}, Z_1, Z_2, Z_3, Y_1^k, Y_2^k, Y_3^k)\}, \\ (Y_1^{k+1}, Y_2^{k+1}, Y_3^{k+1}) = (Y_1^k, Y_2^k, Y_3^k) - \beta[(M^{k+1}, C^{k+1}, K^{k+1}) - (Z_1^{k+1}, Z_2^{k+1}, Z_3^{k+1})]. \end{cases} \tag{9}$$

2.2 Solving the subproblems in (9)

Now, we explain how to solve the resulting subproblems in (9). In fact, it is easy to verify that the first three subproblems are all easy enough to have closed-form solutions because of the simplicity of their objective functions and constraints. More specifically, we have

$$\begin{cases} M^{k+1} &= \Pi_{\mathcal{S}_+^n}\{\frac{1}{c_1+\beta}(c_1 M_a + Y_1^k + \beta Z_1^k)\}, \\ C^{k+1} &= \Pi_{\mathcal{S}^n}\{\frac{1}{c_2+\beta}(c_2 C_a + Y_2^k + \beta Z_2^k)\}, \\ K^{k+1} &= \Pi_{\mathcal{S}_+^n}\{\frac{1}{1+\beta}(K_a + Y_3^k + \beta Z_3^k)\}, \end{cases} \tag{10}$$

where $\Pi_{\mathcal{S}_+^n}$ and $\Pi_{\mathcal{S}^n}$ stand for the metric projections onto $\mathcal{S}_+^n$ and $\mathcal{S}^n$, respectively. In particular, for any given $A \in \mathcal{S}^n$, $\Pi_{\mathcal{S}^n}(A) = A$.

We now discuss how to compute $\Pi_{\mathcal{S}_+^n}(A)$ for a given $A \in \mathcal{S}^n$. According to [16], the spectral decomposition of A is

$$A = W\Psi W^T, \quad \Psi = \mathrm{diag}(\psi_1, \ldots, \psi_n),$$

where $W \in \mathbb{R}^{n\times n}$ is an orthogonal matrix of orthonormal eigenvectors of A and $\{\psi_k\}_{k=1}^n$ are eigenvalues of A. Here, the complexity of the spectral decomposition is $O(9n^3)$ flops [16, Chap. 8]. Then, using the Frobenius inner product, $\Pi_{\mathcal{S}_+^n}(A)$ has the following explicit analytic formula (e.g., [1, 23])

$$\Pi_{\mathcal{S}_+^n}(A) = W\Psi_+ W^T, \quad \Psi_+ = \mathrm{diag}(\max\{\psi_1, 0\}, \ldots, \max\{\psi_n, 0\}).$$

This shows that $\Pi_{\mathcal{S}_+^n}(A)$ can be formulated easily if all eigenvalues and eigenvectors of A are computed. In our numerical experiments, we use the efficient Matlab Mex interface[1] for spectral decomposition.

Meanwhile, the solution $(Z_1^{k+1}, Z_2^{k+1}, Z_3^{k+1})$ to the fourth subproblem in (9) is given by

$$(Z_1^{k+1}, Z_2^{k+1}, Z_3^{k+1}) = \Pi_{S_B}\big\{(V_1^k, V_2^k, V_3^k)\big\},$$

where

$$\left\{\begin{array}{lcl} V_1^k &=& \frac{1}{c_1+\beta}\big(c_1 M_a - Y_1^k + \beta M^{k+1}\big), \\ V_2^k &=& \frac{1}{c_2+\beta}\big(c_2 C_a - Y_2^k + \beta C^{k+1}\big), \\ V_3^k &=& \frac{1}{1+\beta}\big(K_a - Y_3^k + \beta K^{k+1}\big), \end{array}\right.$$

and $\Pi_{S_B}(\cdot)$ stands for the metric projections onto S_B.

In the following, we discuss how to compute $\Pi_{S_B}\{(V_1, V_2, V_3)\}$ for any given $(V_1, V_2, V_3) \in \Omega$. We know that $\Pi_{S_B}\{(V_1, V_2, V_3)\}$ is the unique solution to the following quadratic programming problem:

$$\begin{array}{ll} \min\limits_{(Z_1,Z_2,Z_3)\in\Omega} & \frac{1}{2}\left\|(Z_1, Z_2, Z_3) - (V_1, V_2, V_3)\right\|^2 \\ \text{s.t.} & Z_1\left[\begin{array}{c} R \\ 0 \end{array}\right]\Lambda^2 + Z_2\left[\begin{array}{c} R \\ 0 \end{array}\right]\Lambda + Z_3\left[\begin{array}{c} R \\ 0 \end{array}\right] = 0. \end{array} \tag{11}$$

One may solve problem (11) by many existing standard methods in the literature (see e.g. [24, 27]). Here, by using an idea similar as in [1], we propose to solve problem (11) as follows. Let

$$Z_1 := \left[\begin{array}{cc} Z_{11} & Z_{12} \\ Z_{12}^T & Z_{14} \end{array}\right], \quad Z_2 := \left[\begin{array}{cc} Z_{21} & Z_{22} \\ Z_{22}^T & Z_{24} \end{array}\right], \quad Z_3 := \left[\begin{array}{cc} Z_{31} & Z_{32} \\ Z_{32}^T & Z_{34} \end{array}\right],$$

$$V_1 := \left[\begin{array}{cc} V_{11} & V_{12} \\ V_{12}^T & V_{14} \end{array}\right], \quad V_2 := \left[\begin{array}{cc} V_{21} & V_{22} \\ V_{22}^T & V_{24} \end{array}\right], \quad V_3 := \left[\begin{array}{cc} V_{31} & V_{32} \\ V_{32}^T & V_{34} \end{array}\right],$$

where $Z_{11}, Z_{21}, Z_{31} \in \mathcal{S}^k$, $Z_{12}, Z_{22}, Z_{32} \in \mathbb{R}^{p\times(n-p)}$, $V_{11}, V_{21}, V_{31} \in \mathcal{S}^k$, and $V_{12}, V_{22}, V_{32} \in \mathbb{R}^{p\times(n-p)}$. Let

$$S := R\Lambda R^{-1}, \quad \mathcal{K} := \mathcal{S}^p \times \mathcal{S}^p \times \mathcal{S}^p, \quad \mathcal{M} := \mathbb{R}^{p\times(n-p)} \times \mathbb{R}^{p\times(n-p)} \times \mathbb{R}^{p\times(n-p)}.$$

[1] which computes the spectral decomposition via a divide-and-conquer routine (dgesdd) implemented in LAPACK, and its code can be downdloaded at:`http://videoprocessing.ucsd.edu/~karl/software.html`.

Then it is easy to check that $(\overline{Z}_1, \overline{Z}_2, \overline{Z}_3) \in \Omega$ is a solution of problem (11) if and only if

$$(\overline{Z}_{14}, \overline{Z}_{24}, \overline{Z}_{34}) = (V_{14}, V_{24}, V_{34})$$

and $(\overline{Z}_{11}, \overline{Z}_{21}, \overline{Z}_{31})$ and $(\overline{Z}_{12}, \overline{Z}_{22}, \overline{Z}_{32})$, respectively, solve the following quadratic problems

$$\begin{array}{ll} \min\limits_{(Z_{11},Z_{21},Z_{31})\in\mathcal{K}} & \frac{1}{2}\,\|(Z_{11}, Z_{21}, Z_{31}) - (V_{11}, V_{21}, V_{31})\|^2 \\ \text{s.t.} & \mathcal{H}(Z_{11}, Z_{21}, Z_{31}) := Z_{11}S^2 + Z_{21}S + Z_{31} = 0 \end{array} \tag{12}$$

and

$$\begin{array}{ll} \min\limits_{(Z_{12},Z_{22},Z_{32})\in\mathcal{M}} & \frac{1}{2}\,\|(Z_{12}, Z_{22}, Z_{32}) - (V_{12}, V_{22}, V_{32})\|^2 \\ \text{s.t.} & \mathcal{G}(Z_{12}, Z_{22}, Z_{32}) := (S^2)^T Z_{12} + S^T Z_{22} + Z_{32} = 0. \end{array} \tag{13}$$

We now focus on problems (12) and (13). We first discuss how to solve problem (13). We need the following preliminary lemma.

Lemma 2.1 [31, Lemma 1.3] *Let $A_1 \in \mathbb{C}^{l\times m}$, $A_2 \in \mathbb{C}^{n\times q}$, $A_3 \in \mathbb{C}^{l\times q}$, $E^* \in \mathbb{C}^{m\times n}$ be given. Define*

$$\mathcal{L} := \{E \in \mathbb{C}^{m\times n} : A_1 E A_2 = A_3\}.$$

Then $\mathcal{L} \neq \emptyset$ if and only if A_1, A_2, A_3 satisfy

$$A_1 A_1^\dagger A_3 A_2^\dagger A_2 = A_3,$$

and in case of $\mathcal{L} \neq \emptyset$, any $E \in \mathcal{L}$ can be expressed as:

$$E = A_1^\dagger A_3 A_2^\dagger + T - A_1^\dagger A_1 T A_2 A_2^\dagger,$$

where $T \in \mathbb{C}^{m\times n}$. Moreover, there is a unique matrix $E^{(0)} \in \mathcal{L}$ given by

$$E^{(0)} = A_1^\dagger A_3 A_2^\dagger + E^* - A_1^\dagger A_1 E^* A_2 A_2^\dagger$$

such that for any unitarily invariant norm $|||\cdot|||$,

$$|||E^{(0)} - E^*||| = \min_{E\in\mathcal{L}} |||E - E^*|||.$$

Based on Lemma 2.1, we can easily find the solution of problem (13).

Proposition 2.2 *The unique solution $(\overline{Z}_{12}, \overline{Z}_{22}, \overline{Z}_{32})$ of problem (13) is given by*

$$\left\{\begin{array}{rcl} \overline{Z}_{12} &=& V_{12} - S^2U^{-1}\big((S^2)^T V_{12} + S^T V_{22} + V_{32}\big), \\ \overline{Z}_{22} &=& V_{22} - SU^{-1}\big((S^2)^T V_{12} + S^T V_{22} + V_{32}\big), \\ \overline{Z}_{32} &=& V_{32} - U^{-1}\big((S^2)^T V_{12} + S^T V_{22} + V_{32}\big), \end{array}\right.$$

where $U = (S^2)^T S^2 + S^T S + I$.

Remark 2.3 *Note that the total computational cost for solving problem (13) is $O(np^2)$ flops.*

Next, we study how to solve problem (12). Define

$$\mathcal{R}(\mathcal{H}) := \{\mathcal{H}(Z_{11}, Z_{21}, Z_{31}) : (Z_{11}, Z_{21}, Z_{31}) \in \mathcal{K}\}.$$

Thus, the linear operator $\mathcal{H} : \Omega \to \mathcal{R}(\mathcal{H})$ defined in problem (12) is surjective. The Lagrangian function $L : \mathcal{K} \times \mathcal{R}(\mathcal{H}) \to \mathbb{R}$ associated with problem (12) is defined as

$$L(Z_{11}, Z_{21}, Z_{31}, Y) = \frac{1}{2} \|(Z_{11}, Z_{21}, Z_{31}) - (V_{11}, V_{21}, V_{31})\|^2 - \langle \mathcal{H}(Z_{11}, Z_{21}, Z_{31}), Y \rangle.$$

The Lagrangian dual function $\varphi : \mathcal{R}(\mathcal{H}) \to \mathbb{R}$ is defined as

$$\varphi(Y) = \inf_{(Z_{11}, Z_{21}, Z_{31}) \in \mathcal{K}} L(Z_{11}, Z_{21}, Z_{31}, Y) \quad \forall Y \in \mathcal{R}(\mathcal{H}). \tag{14}$$

We note that $L(Z_{11}, Z_{21}, Z_{31}, Y)$ is a convex quadratic function of (Z_{11}, Z_{21}, Z_{31}). Then we can find the minimum value of L in terms of (Z_{11}, Z_{21}, Z_{31}) based on the optimality condition

$$\nabla_{(Z_{11}, Z_{21}, Z_{31})} L(Z_{11}, Z_{21}, Z_{31}, Y) = (Z_{11}, Z_{21}, Z_{31}) - (V_{11}, V_{21}, V_{31}) - H^*(Y) = 0, \tag{15}$$

where $\mathcal{H}^* : \mathcal{R}(\mathcal{H}) \to \mathcal{K}$ is the adjoint of $\mathcal{H}$, which is given by

$$\mathcal{H}^*(Y) := (\mathcal{H}_1^*(Y), \mathcal{H}_2^*(Y), \mathcal{H}_3^*(Y)) \quad \forall Y \in \mathcal{R}(\mathcal{H})$$

and for each $Y \in \mathcal{R}(\mathcal{H})$,

$$\begin{aligned} \mathcal{H}_1^*(Y) &= \tfrac{1}{2}(Y(S^2)^T + S^2 Y^T), \\ \mathcal{H}_2^*(Y) &= \tfrac{1}{2}(YS^T + SY^T), \\ \mathcal{H}_3^*(Y) &= \tfrac{1}{2}(Y + Y^T). \end{aligned} \tag{16}$$

By (15), we have

$$(Z_{11}, Z_{21}, Z_{31}) = (V_{11}, V_{21}, V_{31}) + H^*(Y). \tag{17}$$

Hence, the Lagrangian dual function is given by

$$\begin{aligned} \varphi(Y) &= \frac{1}{2} \|\mathcal{H}^*(Y)\|^2 - \langle \mathcal{H}((V_{11}, V_{21}, V_{31}) + H^*(Y)), Y \rangle \\ &= \frac{1}{2} \|\mathcal{H}^*(Y)\|^2 - \langle (V_{11}, V_{21}, V_{31}) + H^*(Y), H^*(Y) \rangle \\ &= -\frac{1}{2} \|\mathcal{H}^*(Y)\|^2 - \langle (V_{11}, V_{21}, V_{31}), H^*(Y) \rangle \\ &= -\frac{1}{2} \left\{ \|(V_{11}, V_{21}, V_{31}) + \mathcal{H}^*(Y)\|^2 - \|(V_{11}, V_{21}, V_{31})\|^2 \right\} \quad \forall Y \in \mathcal{R}(\mathcal{H}). \end{aligned}$$

Therefore, the Lagrangian dual problem is given by

$$\max_{Y \in \mathcal{R}(\mathcal{H})} \quad \varphi(Y) \tag{18}$$

or

$$\min_{Y \in \mathcal{R}(\mathcal{H})} \quad \phi(Y) := \tfrac{1}{2} \|(V_{11}, V_{21}, V_{31}) + \mathcal{H}^*(Y)\|^2 - \tfrac{1}{2} \|(V_{11}, V_{21}, V_{31})\|^2 \tag{19}$$

Since $0 \in \mathcal{R}(\mathcal{H})$, Slater's condition (see for instance [5, §5.2.3]) holds for problem (12). By using [29, Theorems 17 and 18], there exists a solution $\overline{Y} \in \mathcal{R}(\mathcal{H})$ to the dual problem (18) such that $\varphi(\overline{Y})$ is equal to the optimal value of the dual problem (18) and is also equal to the optimal value of the original problem (12). That is, the optimal duality gap is zero. By (17), the unique solution to the original problem (12) is given by

$$(\overline{Z}_{11}, \overline{Z}_{21}, \overline{Z}_{31}) = (V_{11}, V_{21}, V_{31}) + H^*(\overline{Y}).$$

Hence, we only need to solve the dual problem (18) or (19). A solution $\overline{Y} \in \mathcal{R}(\mathcal{H})$ of the dual problem (19) solves

$$\nabla\phi(Y) = \mathcal{H}\big((V_{11}, V_{21}, V_{31}) + \mathcal{H}^*(Y)\big) = 0. \tag{20}$$

By (16), we have for any $Y \in \mathcal{R}(\mathcal{H})$,

$$\begin{aligned} & \mathcal{H}\big((V_{11}, V_{21}, V_{31}) + \mathcal{H}^*(Y)\big) = \mathcal{H}(V_{11}, V_{21}, V_{31}) + \mathcal{H}(\mathcal{H}^*(Y)) \\ = \;& \mathcal{H}(V_{11}, V_{21}, V_{31}) + \tfrac{1}{2}Y\big((S^2)^T S^2 + S^T S + I\big) + \tfrac{1}{2}(S^2 Y^T S^2 + S Y^T S + Y^T). \end{aligned}$$

We note that the linear operator $\mathcal{H} : \mathcal{K} \to \mathcal{R}(\mathcal{H})$ defined in problem (12) is onto, $\mathcal{H}\mathcal{H}^*$ is self-adjoint and positive definite. Therefore, one may apply the conjugate gradient (CG) method [16, Algorithm 10.2.1] to solving the equation (20), i.e.,

$$\frac{1}{2}Y\big((S^2)^T S^2 + S^T S + I\big) + \frac{1}{2}(S^2 Y^T S^2 + S Y^T S + Y^T) = -\mathcal{H}(V_{11}, V_{21}, V_{31}). \tag{21}$$

Remark 2.4 *The total cost of computing a solution $\overline{Y} \in \mathcal{R}(\mathcal{H})$ to (20) is $O(p^5)$ flops. This cost is very small since $p \ll n$ in practice. If all the given eigenvalues in Λ are distinct, then $\mathcal{R}(\mathcal{H}) = \mathbb{R}^{p\times p}$ [1, Proposition A.1]. In this case, we can directly solve (20) by using the Kronecker product and vec operator to rewrite (20) as*

$$\begin{aligned} & -\mathrm{vec}(\mathcal{H}(V_{11}, V_{21}, V_{31})) \\ & = \tfrac{1}{2}\left\{\big((S^2)^T S^2 + S^T S + I\big) \otimes I + \big((S^2)^T \otimes S^2 + S^T \otimes S + I\big)T_{p,p}\right\} \mathrm{vec}(Y), \end{aligned} \tag{22}$$

where the matrix T_{pp} transforms $\mathrm{vec}(Y)$ *to* $\mathrm{vec}(Y^T)$*:*

$$T_{p,p}\mathrm{vec}(Y) = \mathrm{vec}(Y^T).$$

In our numerical tests, if the given eigenvalues in Λ are all distinct, we solve (20) by solving (22) for $\mathrm{vec}(\overline{Y})$*, where the required computational cost for solving (22) is $O(p^6)$ flops, which is still acceptable since $p \ll n$. The numerical tests in Section 5 demonstrate that all the ADMM-based schemes to be proposed perform more efficiently for large-scale cases of SDIQEP than the semidefinite programming techniques in [25] and the generalized Newton method in [1].*

3 Three ADMM-based schemes for SDIQEP

Now, we are ready to present several ADMM-based iterative schemes for SDIQEP. More specifically, we apply the original ADMM in [14], the ADMM-based descent method in [33] and the accelerated ADMM in [6] in the context of SDIQEP, and obtain three customized ADMM schemes (i.e., Algorithms 1-3) for the SDIQEP model (5). For notational convenience (mainly for presenting Algorithms 2 and 3 which need to correct the output of (9) to generate a new iterate), the iterate generated by the ADMM scheme (9) is now denoted by $(\widetilde{M}^k, \widetilde{C}^k, \widetilde{K}^k)$, $(\widetilde{Z}_1^k, \widetilde{Z}_2^k, \widetilde{Z}_3^k)$, and $(\widetilde{Y}_1^k, \widetilde{Y}_2^k, \widetilde{Y}_3^k)$. That is, we have

$$\begin{cases} \widetilde{M}^k = \underset{M\in\mathcal{S}_+^n}{\operatorname{argmin}}\{\frac{c_1}{2}\|M-M_a\|_F^2 - \langle Y_1, M-Z_1^k\rangle_F + \frac{\beta}{2}\|M-Z_1^k\|_F^2)\}, \\ \widetilde{C}^k = \underset{C\in\mathcal{S}^n}{\operatorname{argmin}}\{\frac{c_2}{2}\|C-C_a\|_F^2 - \langle Y_2, C-Z_2^k\rangle_F + \frac{\beta}{2}\|C-C_1^k\|_F^2)\}, \\ \widetilde{K}^k = \underset{K\in\mathcal{S}_+^n}{\operatorname{argmin}}\{\frac{c_3}{2}\|K-K_a\|_F^2 - \langle Y_3, K-Z_3^k\rangle_F + \frac{\beta}{2}\|K-Z_3^k\|_F^2)\}, \\ (\widetilde{Z}_1^k, \widetilde{Z}_2^k, \widetilde{Z}_3^k) = \underset{(Z_1,Z_2,Z_3)\in S_B}{\operatorname{argmin}}\{L_\beta(\widetilde{M}^k, \widetilde{C}^k, \widetilde{K}^k, Z_1, Z_2, Z_3, Y_1^k, Y_2^k, Y_3^k)\}, \\ (\widetilde{Y}_1^k, \widetilde{Y}_2^k, \widetilde{Y}_3^k) = (Y_1^k, Y_2^k, Y_3^k) - \beta[(\widetilde{M}^k, \widetilde{C}^k, \widetilde{K}^k) - (\widetilde{Z}_1^k, \widetilde{Z}_2^k, \widetilde{Z}_3^k)]. \end{cases} \tag{23}$$

The first scheme is a straightforward application of the original ADMM in [14].

Algorithm 1: An ADMM method for SDIQEP
Generate the new iterate by the scheme (23), i.e.,

$$\begin{cases} (M^{k+1}, C^{k+1}, K^{k+1}) &= (\widetilde{M}^k, \widetilde{C}^k, \widetilde{K}^k), \\ (Z_1^{k+1}, Z_2^{k+1}, Z_3^{k+1}) &= (\widetilde{Z}_1^k, \widetilde{Z}_2^k, \widetilde{Z}_3^k), \\ (Y_1^{k+1}, Y_2^{k+1}, Y_3^{k+1}) &= (\widetilde{Y}_1^k, \widetilde{Y}_2^k, \widetilde{Y}_3^k). \end{cases}$$

Second, the authors of [33] demonstrate that the output of ADMM at each iteration can be used to construct a descent direction, along which an iterate closer to the solution set of the problem under consideration can be found easily by choosing an appropriate step size. An ADMM-based descent method was thus proposed in [33]. Now, we apply this method to SDIQEP and the resulting iterative scheme is as follows. As analyzed in [33], the parameter $\delta \in (0, 2)$ ensures the contraction of the sequence generated by the ADMM-based descent method and its efficiency of accelerating the convergence empirically was shown in [33].

Algorithm 2: An ADMM-based descent method for SDIQEP

Step 1: Let $(\widetilde{M}^k, \widetilde{C}^k, \widetilde{K}^k, \widetilde{Z}_1^k, \widetilde{Z}_2^k, \widetilde{Z}_3^k, \widetilde{Y}_1^k, \widetilde{Y}_2^k, \widetilde{Y}_3^k)$ be the iterate generated by the scheme (23).

Step 2: Generate the new iterate by

$$\begin{cases} (M^{k+1}, C^{k+1}, K^{k+1}) = (\widetilde{M}^k, \widetilde{C}^k, \widetilde{K}^k), \\ (Z_1^{k+1}, Z_2^{k+1}, Z_3^{k+1}) = (Z_1^k, Z_2^k, Z_3^k) - \delta\theta_k[(Z_1^k, Z_2^k, Z_3^k) - (\widetilde{Z}_1^k, \widetilde{Z}_2^k, \widetilde{Z}_3^k)], \\ (Y_1^{k+1}, Y_2^{k+1}, Y_3^{k+1}) = (Y_1^k, Y_2^k, Y_3^k) - \delta\theta_k[(Y_1^k, Y_2^k, Y_3^k) - (\widetilde{Y}_1^k, \widetilde{Y}_2^k, \widetilde{Y}_3^k)], \end{cases} \tag{24}$$

where $\delta \in (0, 2)$ and

$$\theta_k := \frac{1}{\eta_k}[\eta_k - \langle (Z_1^k, Z_2^k, Z_3^k) - (\widetilde{Z}_1^k, \widetilde{Z}_2^k, \widetilde{Z}_3^k), (Y_1^k, Y_2^k, Y_3^k) - (\widetilde{Y}_1^k, \widetilde{Y}_2^k, \widetilde{Y}_3^k)\rangle] \tag{25}$$

with

$$\eta_k := \beta\|(Z_1^k, Z_2^k, Z_3^k) - (\widetilde{Z}_1^k, \widetilde{Z}_2^k, \widetilde{Z}_3^k)\|^2 + \frac{1}{\beta}\|(Y_1^k, Y_2^k, Y_3^k) - (\widetilde{Y}_1^k, \widetilde{Y}_2^k, \widetilde{Y}_3^k)\|^2.$$

Finally, inspired by the work of [9] which demonstrates that ADMM is an application of the proximal point algorithm (PPA) in [26] to the dual problem of a separable model under consideration, it was recently illustrated in [6] that ADMM is also an application of PPA to the primal problem. Thus, it was proposed in [6] that the acceleration technique in the PPA context [17] can be immediately used to accelerate ADMM. We thus adopt this recent technique and propose the following accelerated ADMM for SDIQEP. Note that the parameter $\gamma \in (0, 2)$ is a relaxation parameter suggested in [17]; and in [6] this parameter is also explained as a parameter ensuring the contraction of ADMM's iterative sequence with respect to the solution set.

Algorithm 3: An accelerated ADMM method for SDIQEP

Generate the new iterate by:

$$\begin{cases} \widetilde{M}^k = \underset{M \in \mathcal{S}_+^n}{\operatorname{argmin}}\{\frac{c_1}{2}\|M - M_a\|_F^2 - \langle Y_1, M - Z_1^k\rangle_F + \frac{\beta}{2}\|M - Z_1^k\|_F^2)\}, \\ \widetilde{C}^k = \underset{C \in \mathcal{S}^n}{\operatorname{argmin}}\{\frac{c_2}{2}\|C - C_a\|_F^2 - \langle Y_2, C - Z_2^k\rangle_F + \frac{\beta}{2}\|C - C_1^k\|_F^2)\}, \\ \widetilde{K}^k = \underset{K \in \mathcal{S}_+^n}{\operatorname{argmin}}\{\frac{c_3}{2}\|K - K_a\|_F^2 - \langle Y_3, K - Z_3^k\rangle_F + \frac{\beta}{2}\|K - Z_3^k\|_F^2)\}, \\ (\widetilde{Y}_1^k, \widetilde{Y}_2^k, \widetilde{Y}_3^k) = (Y_1^k, Y_2^k, Y_3^k) - \beta[(\widetilde{M}^k, \widetilde{C}^k, \widetilde{K}^k) - (Z_1^k, Z_2^k, Z_3^k)], \\ (\widetilde{Z}_1^k, \widetilde{Z}_2^k, \widetilde{Z}_3^k) = \underset{(Z_1, Z_2, Z_3) \in S_B}{\operatorname{argmin}}\{L_\beta(\widetilde{M}^k, \widetilde{C}^k, \widetilde{K}^k, Z_1, Z_2, Z_3, \widetilde{Y}_1^k, \widetilde{Y}_2^k, \widetilde{Y}_3^k)\}, \\ (M^{k+1}, C^{k+1}, K^{k+1}) = (\widetilde{M}^k, \widetilde{C}^k, \widetilde{K}^k), \\ (Z_1^{k+1}, Z_2^{k+1}, Z_3^{k+1}) = (Z_1^k, Z_2^k, Z_3^k) - \gamma[(Z_1^k, Z_2^k, Z_3^k) - (\widetilde{Z}_1^k, \widetilde{Z}_2^k, \widetilde{Z}_3^k)], \\ (Y_1^{k+1}, Y_2^{k+1}, Y_3^{k+1}) = (Y_1^k, Y_2^k, Y_3^k) - \gamma[(Y_1^k, Y_2^k, Y_3^k) - (\widetilde{Y}_1^k, \widetilde{Y}_2^k, \widetilde{Y}_3^k)], \end{cases}$$

where $\gamma \in (0, 2)$ is a relaxation parameter.

Remark 3.1 *It is obvious that the ADMM scheme (23) is invariant in cyclical sense if we exchange the updating order of* $(\widetilde{Z}_1^k, \widetilde{Z}_2^k, \widetilde{Z}_3^k)$ *and* $(\widetilde{Y}_1^k, \widetilde{Y}_2^k, \widetilde{Y}_3^k)$*. Therefore, as emphasized in [6], compared to Algorithm 1, the additional computation of Algorithm 3 is just a linear combination of two triples of matrices. The computational load of this additional step is thus negligible.*

4 Convergence analysis

The convergence of ADMM has been well studied in the literature, see. e.g. [9, 11, 15, 18, 20]. In particular, we refer to [6, 21] where worst-case convergence rates of the original ADMM and the accelerated ADMM were established, respectively. Considering that SDIQEP is a model with matrix variables while most of ADMM literatures are in the context of vectors, we here provide more details about the convergence analysis of ADMM-based schemes for SDIQEP for completeness. On the other hand, for succinctness, we only analyze the convergence of Algorithm 2 and omit the analysis for Algorithms 1 and 3 (see, e.g. [20, Theorem 1]). The following analytic framework can be regarded as an extension of the proof proposed in [19].

We first rewrite the SDIQEP model (5) as a variational characterization, based on which the convergence can be easily established. In fact, by considering the problem of finding a saddle point of the Lagrange function of (5), it is easy to see that solving (5) is equivalent to finding $(M^*, C^*, K^*, Z_1^*, Z_2^*, Z_3^*) \in \Omega_+ \times S_B$ and $(Y_1^*, Y_2^*, Y_3^*) \in \Omega$ such that

$$\begin{cases} \langle (M,C,K) - (M^*,C^*,K^*), (c_1M^*, c_2C^*, K^*) - (c_1M_a, c_2C_a, K_a) - (Y_1^*, Y_2^*, Y_3^*)\rangle \geq 0 \\ \quad \forall\, (M,C,K) \in \Omega_+, \\ \langle (Z_1,Z_2,Z_3) - (Z_1^*, Z_2^*, Z_3^*), (c_1Z_1^*, c_2Z_2^*, Z_3^*) - (c_1M_a, c_2C_a, K_a) + (Y_1^*, Y_2^*, Y_3^*)\rangle \geq 0 \\ \quad \forall\, (Z_1,Z_2,Z_3) \in S_B, \\ \langle (Y_1,Y_2,Y_3) - (Y_1^*, Y_2^*, Y_3^*), (M^*,C^*,K^*) - (Z_1^*, Z_2^*, Z_3^*)\rangle \geq 0 \\ \quad \forall\, (Y_1,Y_2,Y_3) \in \Omega. \end{cases} \tag{26}$$

More compactly, (26) can be rewritten as finding $u^* \in \mathcal{K}$ such that

$$\langle u - u^*, F(u^*)\rangle \geq 0 \quad \forall u \in \mathcal{K},$$

where

$$u = \begin{bmatrix} M \\ C \\ K \\ Z_1 \\ Z_2 \\ Z_3 \end{bmatrix}, \quad F(u) = \begin{bmatrix} c_1(M - M_a) \\ c_2(C - C_a) \\ K - K_a \\ c_1(Z_1 - M_a) \\ c_2(Z_2 - C_a) \\ Z_3 - K_a \end{bmatrix}$$

and

$$\mathcal{K} := \{(M, C, K, Z_1, Z_2, Z_3) \,|\, (M, C, K) \in \Omega_+,\ (Z_1, Z_2, Z_3) \in S_B\}.$$

Note that for any $(M,C,K,Z_1,Z_2,Z_3),(\widetilde{M},\widetilde{C},\widetilde{K},\widetilde{Z}_1,\widetilde{Z}_2,\widetilde{Z}_3)\in\mathcal{K}$, we define

$$\begin{aligned}&\langle(M,C,K,Z_1,Z_2,Z_3),(\widetilde{M},\widetilde{C},\widetilde{K},\widetilde{Z}_1,\widetilde{Z}_2,\widetilde{Z}_3)\rangle\\ =\;&\langle M,\widetilde{M}\rangle_F+\langle C,\widetilde{C}\rangle_F+\langle K,\widetilde{K}\rangle_F+\langle Z_1,\widetilde{Z_1}\rangle_F+\langle Z_2,\widetilde{Z_2}\rangle_F+\langle Z_3,\widetilde{Z_3}\rangle_F.\end{aligned}$$

Recall that the ADMM scheme (9) is rewritten as (23) whose output is denoted by $(\widetilde{M}^k,\widetilde{C}^k,\widetilde{K}^k)$, $(\widetilde{Z}_1^k,\widetilde{Z}_2^k,\widetilde{Z}_3^k)$, and $(\widetilde{Y}_1^k,\widetilde{Y}_2^k,\widetilde{Y}_3^k)$. By deriving the optimality conditions of the minimization problems in (23), we obtain the variational characterization of (23)

$$\left\{\begin{array}{l}\langle M-\widetilde{M}^k,c_1\widetilde{M}^k-c_1M_a-\widetilde{Y}_1^k-\beta(Z_1^k-\widetilde{Z}_1^k)\rangle_F\geq 0\;\;\forall M\in\mathcal{S}_+^n,\\ \langle C-\widetilde{C}^k,c_2\widetilde{C}^k-c_2C_a-\widetilde{Y}_2^k-\beta(Z_2^k-\widetilde{Z}_2^k)\rangle_F\geq 0\;\;\forall C\in\mathcal{S}^n,\\ \langle K-\widetilde{K}^k,\widetilde{K}^k-K_a-\widetilde{Y}_3^k-\beta(Z_3^k-\widetilde{Z}_3^k)\rangle_F\geq 0\;\;\forall K\in\mathcal{S}_+^n,\\ \langle(Z_1,Z_2,Z_3)-(\widetilde{Z}_1^k,\widetilde{Z}_2^k,\widetilde{Z}_3^k),(c_1\widetilde{Z}_1^k,c_2\widetilde{Z}_2^k,\widetilde{Z}_3^k)-(c_1M_a,c_2C_a,K_a)\\ \qquad+(\widetilde{Y}_1^k,\widetilde{Y}_2^k,\widetilde{Y}_3^k)\rangle\geq 0\;\;\forall(Z_1,Z_2,Z_3)\in S_B,\\ \langle(Y_1,Y_2,Y_3)-(\widetilde{Y}_1^k,\widetilde{Y}_2^k,\widetilde{Y}_3^k),(\widetilde{M}^k,\widetilde{C}^k,\widetilde{K}^k)-(\widetilde{Z}_1^k,\widetilde{Z}_2^k,\widetilde{Z}_3^k)\\ \qquad+\frac{1}{\beta}[(\widetilde{Y}_1^k,\widetilde{Y}_2^k,\widetilde{Y}_3^k)-(Y_1^k,Y_2^k,Y_3^k)]\rangle\;\;\forall(Y_1,Y_2,Y_3)\in\Omega,\end{array}\right.\tag{27}$$

For notation simplicity, let us define

$$\begin{array}{lll}\overline{H}=(\overline{M},\overline{C},\overline{K}), & H^k=(M^k,C^k,K^k) & \widetilde{H}^k=(\widetilde{M}^k,\widetilde{C}^k,\widetilde{K}^k),\\ \overline{J}=(\overline{Z}_1,\overline{Z}_2,\overline{Z}_3), & J^k=(Z_1^k,Z_2^k,Z_3^k), & \widetilde{J}^k=(\widetilde{Z}_1^k,\widetilde{Z}_2^k,\widetilde{Z}_3^k),\\ \overline{L}=(Y_1^*,Y_2^*,Y_3^*), & L^k=(Y_1^k,Y_2^k,Y_3^k), & \widetilde{L}^k=(\widetilde{Y}_1^k,\widetilde{Y}_2^k,\widetilde{Y}_3^k).\end{array}$$

Then, the following results whose proofs are similar as those of Lemmas 3.1–3.3 and Theorem 3.4 in [19] can be easily shown. We thus omit their proofs.

Lemma 4.1 *Let $\overline{V}=(\overline{H},\overline{J},\overline{L})$ be a solution of* (26) *and $(H^{k+1},J^{k+1},L^{k+1})$ be generated by Algorithm 2 from the given iterate (H^k,J^k,L^k). Then we have*

(a)
$$\langle\widetilde{J}^k-\overline{J},\beta(J^k-\widetilde{J}^k)\rangle+\langle\widetilde{L}^k-\overline{L},\frac{1}{\beta}(L^k-\widetilde{L}^k)\rangle\geq-\langle J^k-\widetilde{J}^k,L^k-\widetilde{L}^k\rangle;$$

(b)
$$\begin{aligned}&\langle J^k-\overline{J},\beta(J^k-\widetilde{J}^k)\rangle+\langle L^k-\overline{L},\tfrac{1}{\beta}(L^k-\widetilde{L}^k)\rangle\\ \geq\;&\beta\|J^k-\widetilde{J}^k\|^2+\tfrac{1}{\beta}\|L^k-\widetilde{L}^k\|^2-\langle J^k-\widetilde{J}^k,L^k-\widetilde{L}^k\rangle;\end{aligned}$$

(c)
$$\begin{aligned}&\beta\|J^k-\widetilde{J}^k\|^2+\tfrac{1}{\beta}\|L^k-\widetilde{L}^k\|^2-\langle J^k-\widetilde{J}^k,L^k-\widetilde{L}^k\rangle\\ =\;&\tfrac{1}{2}\big(\beta\|J^k-\widetilde{J}^k\|^2+\tfrac{1}{\beta}\|L^k-\widetilde{L}^k\|^2+\|\sqrt{\beta}(J^k-\widetilde{J}^k)-\tfrac{1}{\sqrt{\beta}}(L^k-\widetilde{L}^k)\|^2\big);\end{aligned}$$

(d)

$$\begin{aligned} & \beta\|J^{k+1}-\overline{J}\|^2+\tfrac{1}{\beta}\|L^{k+1}-\overline{L}\|^2 \\ \leq\ & \beta\|J^{k}-\overline{J}\|^2+\tfrac{1}{\beta}\|L^{k}-\overline{L}\|^2-\tfrac{1}{2}\delta(2-\delta)\big(\beta\|J^k-\widetilde{J}^k\|^2+\tfrac{1}{\beta}\|L^k-\widetilde{L}^k\|^2\big) \\ & -\tfrac{1}{2}\delta(2-\delta)\big(\|\sqrt{\beta}(J^k-\widetilde{J}^k)-\tfrac{1}{\sqrt{\beta}}(L^k-\widetilde{L}^k)\|^2\big). \end{aligned}$$

In Lemma 4.1, the first three assertions can be obtained by elementary operations based on the resulting inequalities when the inequalities in (27) are applied for both the k-th and $(k+1)$-th iterations of Algorithm 2. Consequently, these assertions immediately imply the assertion (d) in Lemma 4.1. Then, according to [2], the assertion (d) in Lemma 4.1 essentially implies that the sequence $\{(H^{k+1},J^{k+1},L^{k+1})\}$ generated by Algorithm 2 is contractive with respect to the solution set of the variational characterization (26). Thus, the standard analytic framework for proving the convergence of a contractive sequence in [2] is applicable. Moreover, in the following lemma some simple conclusions which are useful for establishing the convergence of Algorithm 2 are summarized. These assertions are trivial conclusions based on the assertion (d) in Lemma 4.1, and thus omitted. We refer to [19, Corollary 3.6] for detail of proof.

Corollary 4.2 *Let $\overline{V}=(\overline{H},\overline{J},\overline{L})$ be the solution of* (26) *and $(H^{k+1},J^{k+1},L^{k+1})$ be generated by Algorithm 2 from the given iterate (H^k,J^k,L^k). Then we have*

(a) *The sequence $\{\beta\|J^k-\overline{J}\|^2+\frac{1}{\beta}\|L^k-\overline{L}\|^2\}$ is nonincreasing.*

(b) *The sequence $\{(H^{k+1},J^{k+1},L^{k+1})\}$ is bounded.*

(c) $\lim_{k\to\infty}(\beta\|J^k-\widetilde{J}^k\|^2+\frac{1}{\beta}\|L^k-\widetilde{L}^k\|^2)=0$.

(d) *The sequence $\{(\widetilde{H}^k,\widetilde{J}^k,\widetilde{L}^k)\}$ is bounded.*

Now we are ready to present the convergence result of Algorithm 2 by following the idea of the proof of [19, Theorem 3.7].

Theorem 4.3 *Let $(\widetilde{H}^k,\widetilde{J}^k,\widetilde{L}^k)$ be generated by (9) and $(H^{k+1},J^{k+1},L^{k+1})$ be generated Algorithm 2. Then the sequence $\{(H^{k+1},J^{k+1},L^{k+1})\}$ converges to a solution of* (26) *and the corresponding sequence $\{(H^{k+1},J^{k+1})\}$ converges to a solution of the SDIQEP model* (5).

Proof: According to the assertion (d) in Corollary 4.2, we know that the sequence $\{(\widetilde{H}^k,\widetilde{J}^k,\widetilde{L}^k)\}$ is bounded. Therefore, it has at least one limit point. Let $\{H^\infty,J^\infty,L^\infty\}$ be a limit point of $\{(\widetilde{H}^k,\widetilde{J}^k,\widetilde{L}^k)\}$, where $H^\infty=(M^\infty,C^\infty,K^\infty)$, $J^\infty=(Z_1^\infty,Z_2^\infty,Z_3^\infty)$, and $L^\infty=(Y_1^\infty,Y_2^\infty,Y_3^\infty)$. Let $\{(\widetilde{H}^{k_j},\widetilde{J}^{k_j},\widetilde{L}^{k_j})\}$ be the subsequence of $\{(\widetilde{H}^k,\widetilde{J}^k,\widetilde{L}^k)\}$ converging to $\{H^\infty,J^\infty,L^\infty\}$. Then by using (27) and the assertion (c) in Corollary 4.2, we have

$$\begin{cases} \lim\limits_{j\to\infty}\langle H-\widetilde{H}^{k_j},\widetilde{H}^{k_j}-H_a-\widetilde{L}^{k_j}\rangle\geq 0 \quad \forall H\in\Omega_+, \\ \lim\limits_{j\to\infty}\langle J-\widetilde{J}^{k_j},\widetilde{J}^{k_j}-H_a+\widetilde{L}^{k_j}\rangle\geq 0 \quad \forall J\in S_B, \\ \lim\limits_{j\to\infty}\langle L-\widetilde{L}^{k_j},\widetilde{H}^{k_j}-\widetilde{H}^{k_j}\rangle\geq 0 \quad \forall L\in\Omega, \end{cases}$$

i.e.,

$$\begin{cases} \langle H - H^\infty, H^\infty - H_a - L^\infty \rangle \geq 0 \quad \forall\, H \in \Omega_+, \\ \langle J - J^\infty, J^\infty - H_a + L^\infty \rangle \geq 0 \quad \forall\, J \in S_B, \\ \langle L - L^\infty, H^\infty - J^\infty \rangle \geq 0 \quad \forall\, L \in \Omega. \end{cases}$$

By (26), we know that $\{H^\infty, J^\infty, L^\infty\}$ is a solution of (26).

Next, we show that $\{(H^{k+1}, J^{k+1}, L^{k+1})\}$ converges to $\{H^\infty, J^\infty, L^\infty\}$. In view of the assertion (c) in Corollary 4.2, we have that for any given $\epsilon > 0$, there exists an integer $\widetilde{k} > 0$ such that

$$\beta\|J^k - \widetilde{J}^k\|^2 + \frac{1}{\beta}\|L^k - \widetilde{L}^k\|^2 < \frac{\epsilon}{2} \quad \forall\, k > \widetilde{k}.$$

Similarly, there exists an integer $k_l > 0$ such that

$$\beta\|\widetilde{J}^{k_j} - J^\infty\|^2 + \frac{1}{\beta}\|\widetilde{L}^{k_j} - L^\infty\|^2 < \frac{\epsilon}{2} \quad \forall\, k_j > k_l.$$

Considering the assertion (a) Corollary 4.2, we have

$$\begin{aligned} & \beta\|J^{k+1} - J^\infty\|^2 + \tfrac{1}{\beta}\|L^{k+1} - L^\infty\|^2 \leq \beta\|J^{k_j} - J^\infty\|^2 + \tfrac{1}{\beta}\|L^{k_j} - L^\infty\|^2 \\ \leq\ & \big(\beta\|J^{k_j} - \widetilde{J}^{k_j}\|^2 + \tfrac{1}{\beta}\|L^{k_j} - \widetilde{L}^{k_j}\|^2\big) + \big(\beta\|\widetilde{J}^{k_j} - J^\infty\|^2 + \tfrac{1}{\beta}\|\widetilde{L}^{k_j} - L^\infty\|^2\big) \\ <\ & \tfrac{\epsilon}{2} + \tfrac{\epsilon}{2} = \epsilon \quad \forall\, k \geq k_j \geq \max\{\widetilde{k}, k_l\}, \end{aligned}$$

which implies that the sequence $\{J^{k+1}, L^{k+1}\}$ converges to $\{J^\infty, L^\infty\}$.

Since $\Pi_{\mathcal{S}_+^n}(\cdot)$ is continuous, it follows from (10) and (24) that

$$\begin{aligned} & \lim_{k\to\infty} H^{k+1} = \lim_{k\to\infty} (M^{k+1}, C^{k+1}, K^{k+1}) = \lim_{k\to\infty} (\widetilde{M}^k, \widetilde{C}^k, \widetilde{K}^k) \\ =\ & \lim_{k\to\infty} \Pi_{\Omega_+}\{\tfrac{1}{1+\beta}\big((M_a, C_a, K_a) + (Y_1^k, Y_2^k, Y_3^k) + \beta(Z_1^k, Z_2^k, Z_3^k)\big)\} \\ =\ & \Pi_{\Omega_+}\{\tfrac{1}{1+\beta}\big((M_a, C_a, K_a) + (Y_1^\infty, Y_2^\infty, Y_3^\infty) + \beta(Z_1^\infty, Z_2^\infty, Z_3^\infty)\big)\}. \end{aligned}$$

Therefore, $\{H^{k+1}\}$ is convergent. Notice that H^∞ is a limit of $\{\widetilde{H}^k\}$. Thus,

$$\lim_{k\to\infty} H^{k+1} = H^\infty.$$

The proof is complete. □

Remark 4.4 *It follows from (26) and (27) that* $(\widetilde{M}^k, \widetilde{C}^k, \widetilde{K}^k, \widetilde{Z}_1^k,\ \widetilde{Z}_2^k, \widetilde{Z}_3^k,\ \widetilde{Y}_1^k, \widetilde{Y}_2^k, \widetilde{Y}_3^k)$ *is a solution of (26) if and only if* $(Z_1^k, Z_2^k, Z_3^k) = (\widetilde{Z}_1^k, \widetilde{Z}_2^k, \widetilde{Z}_3^k)$ *and* $(\widetilde{Y}_1^k, \widetilde{Y}_2^k, \widetilde{Y}_3^k) = (Y_1^k, Y_2^k, Y_3^k)$. *Hence, a simple stopping criterion for implementing all the proposed algorithms is immediately available:*

$$\begin{cases} \|(Z_1^k, Z_2^k, Z_3^k) - (Z_1^{k+1}, Z_2^{k+1}, Z_3^{k+1})\|_{\max} \leq \epsilon, \\ \|(Y_1^k, Y_2^k, Y_3^k) - (Y_1^{k+1}, Y_2^{k+1}, Y_3^{k+1})\|_{\max} \leq \epsilon. \end{cases} \tag{28}$$

where $\|A\|_{\max}$ *denotes the largest absolute value among* A*'s entries and* $\epsilon > 0$ *is a prescribed tolerance. We will use this stopping criterion to implement the proposed ADMM-based schemes in Section 5.1.1.*

5 Numerical Experiments

In this section, we report some numerical results when the proposed ADMM-based algorithms are implemented to solve SDIQEP, including some large scale cases where $n \geq 1,000$. All the proposed ADMM-based algorithms were coded by `MATLAB 7.1`. To show the efficiency of the proposed ADMM-based algorithms, we also compared them with some existing methods including an interior-point-based method and a generalized Newton method. All the numerical experiments were completed on a personal computer with a Intel(R) Core(TM)2 Quad of 2.33 GHz CPU and 3.00 GB of RAM. Throughout, ``IT.", ``CT.", ``Res.", and ``RRes." mean the number of iterations, the total computing time in seconds, the residual $\|M^k X\Lambda^2 + C^k X\Lambda + K^k X\|_F$ and the relative residual $\|M^k X\Lambda^2 + C^k X\Lambda + K^k X\|_F/(\|M^k X\Lambda^2\|_F + \|C^k X\Lambda\|_F + \|K^k X\|_F)$ at the final iterate by implementing the proposed algorithms, respectively.

5.1 Synthetic data-set

For the partial eigendata, throughout this subsection the matrix $\Lambda \in \mathbb{R}^{p\times p}$ is generated pseudo-randomly by the `MATLAB` function "`randn`", where the total number of complex-valued eigenvalues is set to be around $p/2$. The matrix $R \in \mathbb{R}^{p\times p}$ is computed by the QR factorization of a pseudo-random $n \times p$ matrix by using "`randn`". We test the following two examples.

Example 5.1 *Let $\widehat{M}$, $\widehat{C}$, and $\widehat{K}$ be given respectively by*

$$\widehat{M} = \begin{bmatrix} R^{-T}R^{-1} & 0 \\ 0 & I \end{bmatrix}, \quad \widehat{C} = \begin{bmatrix} -R^{-T}(\Lambda + \Lambda^T)R^{-1} & 0 \\ 0 & 0 \end{bmatrix}, \quad \widehat{K} = \begin{bmatrix} R^{-T}\Lambda^T\Lambda R^{-1} & 0 \\ 0 & I \end{bmatrix}.$$

Then we set

$$M_a := \widehat{M} + \tau R_M, \quad C_a := \widehat{C} + \tau R_C, \quad K_a := \widehat{K} + \tau R_K,$$

where R_M, R_C, and R_K are $n \times n$ symmetric matrices whose entries are generated pseudo-randomly and they are uniformly distributed within $[-1.0, 1.0]$, and $\tau \in \mathbb{R}$ is a perturbed parameter. For succinctness, we only report numerical results for the case where $\tau = 0.1$.

Example 5.2 *The matrices G_M and G_K are pseudo-random $n\times n$ correlation matrices generated by `MATLAB 7.10's gallery ('randcorr',n)`, and the matrix G_C is a pseudo-random $n \times n$ symmetric matrix with entries $(G_C)_{ij} \in [-1, 1]$ and $(G_C)_{ii} = 1.0$ for $i, j = 1, 2, \ldots, n$. Then M_a, C_a and K_a are, respectively, obtained via perturbing G_M, G_C and G_K by a pseudo-random $n \times n$ symmetric matrix with entries in $[-\tau, \tau]$. We only report the case where $\tau = 0.1$.*

For the SDIQEP model (4), we set $c_1 = c_2 = 1.0$. To implement the proposed ADMM-based algorithms, there are several parameters whose values should be specified. Theoretically, one may choose $\beta > 0$, $\delta \in (0, 2)$, and $\gamma \in (0, 2)$. For β, it is widely known that this parameter should be tuned for a particular problem empirically. Our tuned result is fixing $\beta = 35$ for all Algorithms 1-3. For the parameters δ in Algorithm 2 and γ in Algorithm 3, as we have explained, they are of the role of relaxation parameters and it is recommended in the literature to take "overdetermined" values (i.e., values in $(1, 2)$) in implementation, see e.g., [6, 9, 15, 17]. We choose $\delta = 1.6$ and $\gamma = 1.8$ in our experiments. In Section 5.3, we also report the sensitivity

of the proposed algorithms to these parameters. All the ADMM-based algorithms start their iterations with the initial iterate $(M^0, C^0, K^0) = (I_n, I_n, I_n)$, $(Z_1^0, Z_2^0, Z_3^0) = (I_n, I_n, I_n)$ and $(Y_1^0, Y_2^0, Y_3^0) = (0, 0, 0)$.

5.1.1 Comparison with an interior-point approach

In this section, we compare the ADMM approach with an interior-point approach for SDIQEP. More specifically, as in [25], we can reformulate the SDIQEP model (4) as a standard semidefinite programming (SDP) problem defined as follows:

$$\begin{array}{ll} \min & \frac{c_1}{2} z_1 + \frac{c_2}{2} z_2 + \frac{1}{2} z_3 \\ \text{s.t.} & M \left[\begin{array}{c} R \\ 0 \end{array} \right] \Lambda^2 + C \left[\begin{array}{c} R \\ 0 \end{array} \right] \Lambda + K \left[\begin{array}{c} R \\ 0 \end{array} \right] = 0, \\ & \sqrt{z_1} \geq \|M - M_a\|_F, \\ & \sqrt{z_2} \geq \|C - C_a\|_F, \\ & \sqrt{z_3} \geq \|K - K_a\|_F, \\ & M \in \mathcal{S}_+^n, \quad C \in \mathcal{S}^n, \quad K \in \mathcal{S}_+^n. \end{array}$$

Thus, the benchmark package SDPT3, which is based on an interior point algorithm (see [32]), is applicable to this SDP reformulation of SDIQEP. As we have mentioned, we use the stopping criterion (28) to implement Algorithms 1–3, where $\epsilon = 10^{-7}$.

In Tables 1–2, we compare Algorithms 1–3 with this interior-point approach (denoted by "IPM") for Examples 5.1–5.2 with different values of n and p.

The data in Tables 1–2 show the following conclusions: 1) When the problem is large (say, $n \geq 100$ and $p = 10$), out computer runs out of memory for the interior-point approach, while the ADMM approach is applicable for large scale cases; 2) The ADMM approach is much faster than the interior-point approach to achieve comparable accuracy; and 3) Algorithms 2–3 converge faster than Algorithm 1 and Algorithms 3 performs the best, especially for medium or large scale cases, which coincides with the conclusions in [6, 33].

Since IPM is already very slow for small values of n and p, we do not report more numerical results with other larger values of n and p for succinctness purpose.

5.1.2 Comparison with a generalized Newton approach

In this section, we compare the proposed ADMM approach with the generalized Newton method in [1], which has been shown to be applicable to large scale cases of SDIQEP. For demonstration purpose, the proposed ADMM approach and the generalized Newton method are stopped when

$$\|M^k X \Lambda^2 + C^k X \Lambda + K^k X\|_F \leq \epsilon. \tag{29}$$

In Tables 3–8, we test a number of scenarios of Examples 5.1 and 5.2 with different values of n, p and ϵ. Since the computation of each iteration of the generalized Newton method in [1] is dominated by computing Π_{Ω_+} and solving a system of linear equations, we count the time of both of these two tasks in the iterations and report them individually (denoted by "t_1" and

Table 1: Comparison of ADMM and an interior-point approache for Example 5.1 ("—" means out of memory).

		IPM			Algorithm 1			Algorithm 2			Algorithm 3		
n	p	IT.	CT.	Res.	IT.	CT.	Res.	IT.	CT.	Res.	IT.	CT.	Res.
30	6	20	9.4	4.5×10^{-9}	241	2.3	8.8×10^{-9}	150	1.4	5.2×10^{-9}	138	1.1	1.8×10^{-8}
40	8	24	17.3	7.9×10^{-9}	258	2.2	4.2×10^{-8}	156	1.3	2.1×10^{-8}	152	1.1	3.7×10^{-8}
50	10	21	32.7	5.3×10^{-8}	240	2.0	2.0×10^{-8}	149	1.2	1.0×10^{-8}	136	1.1	9.5×10^{-9}
60	10	21	64.7	9.2×10^{-8}	268	2.4	4.2×10^{-8}	157	1.6	3.2×10^{-8}	142	1.3	1.1×10^{-7}
100	10		—		240	3.2	1.9×10^{-7}	167	2.5	1.6×10^{-7}	165	2.3	1.6×10^{-7}
200	10		—		461	15.6	4.5×10^{-8}	281	12.7	3.8×10^{-8}	253	8.9	3.9×10^{-8}
500	10		—		469	106.8	6.7×10^{-8}	288	92.0	4.1×10^{-8}	259	65.0	3.9×10^{-8}
1000	10		—		485	583.2	9.1×10^{-8}	294	465.3	5.8×10^{-8}	263	340.6	5.4×10^{-8}

Table 2: Comparison of ADMM and an interior-point approach for Example 5.2 ("—" means out of memory).

		IPM			Algorithm 1			Algorithm 2			Algorithm 3		
n	p	IT.	CT.	Res.	IT.	CT.	Res.	IT.	CT.	Res.	IT.	CT.	Res.
30	6	18	6.0	5.1×10^{-9}	239	1.5	3.7×10^{-9}	149	1.0	5.2×10^{-9}	149	1.0	3.4×10^{-8}
40	8	19	14.1	1.3×10^{-8}	238	1.7	2.2×10^{-8}	148	1.1	9.9×10^{-9}	138	1.0	1.9×10^{-8}
50	10	20	31.3	3.4×10^{-9}	239	1.8	1.8×10^{-9}	149	1.3	2.1×10^{-9}	144	1.1	2.4×10^{-8}
60	10		—		271	2.4	1.9×10^{-7}	201	2.0	1.2×10^{-7}	191	1.7	1.2×10^{-7}
100	10		—		274	3.6	5.8×10^{-8}	161	2.4	3.5×10^{-8}	171	2.3	2.8×10^{-8}
200	10		—		252	8.6	8.6×10^{-10}	157	7.0	5.0×10^{-9}	151	5.5	3.7×10^{-8}
500	10		—		285	64.2	9.4×10^{-8}	196	62.3	5.1×10^{-8}	189	48.2	6.3×10^{-8}
1000	10		—		369	442.6	1.4×10^{-7}	250	393.8	5.1×10^{-8}	226	293.3	5.9×10^{-8}

"t_2", respectively). For Algorithms 1–3, "t_1" and "t_2" are the time of computing Π_{Ω_+} and Π_{S_B}, respectively. Moreover, for Algorithm 2, "t_3" is the aggregated time of computing the step size θ_k defined in (25).

Since both the compared approaches are applicable to large scale cases of SDIQEP, we observe their difference numerically for more scenarios of n and p than Section 5.1.1, and we test the sensitivity of their difference to different values of n and p. First, in Table 3, we fix $p = 30$ for Example 5.1 and compare these two approaches for different values of n. We see that the ADMM approach is more efficient than the generalized Newton approach, as it requires much less computing time. Note that "t_2" for "Newton" is dominated, which means the generalized Newton method in [1] spends most of its computing time in solving the resulting system of linear equations at each iteration. At the same time, the time for computing Π_{Ω_+} for the generalized Newton method in [1] is much less than that of the ADMM approach. This is because the generalized Newton method in [1] is a Newton-like method and it requires only a few iterations, while ADMM is a first-order method and it requires more iterations.

We have tested some other values of p with $p > 30$ and we could derive the same conclusion as above for Table 3. On the other hand, for cases where values of p are smaller (e.g. $p \leq 10$) and n is not large (e.g. $n \leq 500$), we found that the generalized Newton method in [1] could

be more competitive than the ADMM approach. The reason is clear: Because, for these cases, the dimensionality of the resulting systems of linear equations (proportional to np) are not in that high dimension and thus Newton-like methods could be faster because of their significantly smaller numbers of iterations). To see this fact clearly, we report some results in Tables 4 and 5 when n is fixed as 500 and $1,000$ for Example 5.1, respectively, while different values of p are chosen. Data in Tables 4 and 5 shows the efficiency of the ADMM approach whenever np is not very small (e.g., $np > 10,000$).

5.2 A model updating problem in structural engineering

In this section, we test the efficiency of the proposed algorithms for the model updating problem in structural engineering, where the mass and stiffness matrix $\widehat{M}$ and $\widehat{K}$ are obtained from the set BCSSTRUC1 in the Harwell-Boeing Collection [3] and the damping matrix $\widehat{C}$ are Rayleigh damping [8, 10]:

$$\widehat{C} = \omega_0\widehat{M} + \omega_1\widehat{K}.$$

Here, $\omega_0 > 0$ is the mass proportional Rayleigh damping coefficient and $\omega_1 > 0$ is the stiffness proportional Rayleigh damping coefficient.

Example 5.3 *Let $\widehat{M}$ and $\widehat{K}$ be BCSSTM02 and BCSSTK02, respectively, and $\widehat{C} = 0.025\widehat{M} + 0.025\widehat{K}$ ($n = 66$). We choose p eigenpairs $\{(\lambda_j, x_j)\}_{j=1}^{p}$ of the pencil $\widehat{P}(\lambda) := \lambda^2\widehat{M} + \lambda\widehat{C} + \widehat{K}$ as the given measured eigendata, where*

(a) $p = 8$ and

$$\begin{aligned}\{\lambda_j\}_{j=1}^{8} &= \{-453.3866, -387.7504, -326.9562, -341.0726, -283.1585, -240.3790,\\ &\quad -222.9141, -179.7670\}.\end{aligned}$$

(b) $p = 12$ and

$$\begin{aligned}\{\lambda_j\}_{j=1}^{12} &= \{-40.5213, -40.4280, -40.4562, -40.4723, -40.4964, -40.4904, -40.3643,\\ &\quad -\mathbf{40.2993}, -\mathbf{40.2997}, -40.3292, -40.3371, -40.3349\}.\end{aligned}$$

(c) $p = 16$ and

$$\begin{aligned}\{\lambda_j\}_{j=1}^{16} &= 10^3 \times \{-4.8140, -4.4298, -3.7764, -3.5453, -3.4259, -3.3003, -3.2611,\\ &\quad -3.1073, -2.4816, -2.4758, -2.1766, -2.1180, -1.9497, -1.7504,\\ &\quad -1.7414, -1.6292\}.\end{aligned}$$

Also, we set

$$M_a := \widehat{M} + \tau R_M .* \widehat{M}, \quad C_a := \widehat{C} + \tau R_C .* \widehat{C}, \quad K_a := \widehat{K} + \tau R_K .* \widehat{K},$$

where "." means element-by-element multiplication, R_M, R_C, and R_K are $n \times n$ symmetric matrices whose entries are generated pseudo-randomly and they are uniformly distributed within $[-1.0, 1.0]$, and $\tau \in \mathbb{R}$ is a perturbed parameter. For succinctness, we only report numerical results for the case where $\tau = 0.1$.*

Example 5.4 *Let $\widehat{M}$ and $\widehat{K}$ be BCSSTM06 and BCSSTK06, respectively, and $\widehat{C} = 0.025\widehat{M} + 0.025\widehat{K}$ ($n = 420$). We choose p eigenpairs $\{(\lambda_j, x_j)\}_{j=1}^{p}$ of the pencil $\widehat{P}(\lambda) := \lambda^2\widehat{M} + \lambda\widehat{C} + \widehat{K}$ as the given measured eigendata, where $p = 12$ and*

$$\begin{aligned}\{\lambda_j\}_{j=1}^{12} &= \{-40.0068, -40.0068, -40.0001, -40.0001, -40.0001, -40.0001, -40.0001,\\ &\quad -40.0001, -40.0001, -40.0001, -40.0001, -40.0001\}.\end{aligned}$$

Also, we set M_a, C_a, and K_a as in Example 5.3.

Example 5.5 *Let $\widehat{M}$ and $\widehat{K}$ be BCSSTM08 and BCSSTK08, respectively, and $\widehat{C} = 0.025\widehat{M} + 0.025\widehat{K}$ ($n = 1074$). We choose p eigenpairs $\{(\lambda_j, x_j)\}_{j=1}^{p}$ of the pencil $\widehat{P}(\lambda) := \lambda^2\widehat{M} + \lambda\widehat{C} + \widehat{K}$ as the given measured eigendata, where $p = 16$ and*

$$\begin{aligned}\{\lambda_j\}_{j=1}^{16} &= 10^5 \times \{-4.2161, -3.0247, -2.8267, -2.7637, -1.8946, -1.7195, -1.7009, -1.8438,\\ &\quad -1.5837, -1.8394, -1.8395, -1.5175, -1.8395, -1.3252, -1.2202, -1.2170\}.\end{aligned}$$

Also, we set M_a, C_a, and K_a as in Example 5.3.

We apply the ADMM-based algorithms to Examples 5.3–5.5 with the initial guess and other parameters as above. Here, when there exist multiple eigenvalues in Λ (see Ex. 5.4), we solve the linear system (20) (i.e., (21)) by using the CG method, where $Y \in \mathcal{R}(\mathcal{H})$ is guaranteed if the initial guess Y_0 is in $\mathcal{R}(\mathcal{H})$ (we set $Y_0 = 0$ in our tests). To improve the numerical performance, instead of (21), we use the CG method to the following preconditioned linear system:

$$\frac{1}{2}YU + \frac{1}{2}(S^2Y^TS^2 + SY^TS + Y^T)U^{-1} = -\mathcal{H}(V_{11}, V_{21}, V_{31})U^{-1}.$$

where $U = (S^2)^TS^2 + S^TS + I$. In our tests, the largest number of iterations in CG is set to be 100.

The numerical results when the ADMM-based algorithms are applied to solve Examples 5.3–5.5 and their comparison with the interior point and generalized Newton approaches are listed in Table 9. The stopping criterion is:

$$\|M^kX\Lambda^2 + C^kX\Lambda + K^kX\|_F/(\|M^kX\Lambda^2\|_F + \|C^kX\Lambda\|_F + \|K^kX\|_F) < \epsilon = 10^{-5}.$$

We can see from Table 9 that the proposed algorithms are very efficient for Examples 5.3–5.5. For Example 5.4, the average number of iterations for the CG methods is less than 10. This shows that the proposed ADMM approach works very effectively for solving the SDIQEP with multiple eigenvalues.

5.3 Sensitivity to parameters

Now, we report the sensitivity of the ADMM approach to the involved parameters β, δ and γ. For succinctness, we only focus on the case of Example 5.2 with $n = 200$ and $p = 30$. In Figures 1 and 2, we plot the number of iterations and the computing time (in seconds) satisfying the criterion (29) with different accuracy with respect to different choices of β and (δ, γ), respectively.

We see from Figures 1 and 2 that, for fixed δ and γ, the efficiency of the ADMM approach is affected by the value of β. According to our numerical experiments, we find that some extreme values of β (i.e., too large or small) are not preferred to implement the proposed ADMM-based algorithms. On the other hand, when the value of β is fixed, the ADMM approach is very robust to the values of δ and γ as long as they are larger than 1.0.

5.4 Some remarks

Based on the reported data, we remark that Algorithms 2 and 3 both outperform Algorithm 1 for most of the tested scenarios. In particular, Algorithm 3 performs the best because its additional acceleration step is inexpensive, while Algorithm 2 requires some considerable time to compute the step sizes in order to reduce the number of iterations. Thus, the effectiveness of the advanced ADMM-based methods in [6, 33] are further verified in the context of SDIQEP. Finally, it is interesting to notice that for the ADMM approach, the number of iterations seems not very sensitive to the numbers of constraints (i.e., np).

Table 3: Comparison of ADMM and the generalized Newton approach in [1] for Example 5.1 with $p = 30$.

$\epsilon = 10^{-3}$								
	Newton		Algorithm 1		Algorithm 2		Algorithm 3	
n	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2, t_3)	IT.	CT. (t_1, t_2)
100	13	36.8 (0.0, 36.3)	178	4.0 (1.7, 1.5)	107	3.0 (1.1, 1.2, 0.1)	97	2.7 (1.1, 1.2)
200	7	23.7 (0.2, 23.2)	136	6.1 (3.4, 1.4)	83	5.1 (1.8, 1.1, 0.8)	78	4.1 (2.0, 1.2)
500	7	106.7 (1.4, 104.8)	132	30.2 (21.8, 2.8)	90	29.7 (15.2, 2.1, 6.3)	90	23.2 (15.0, 2.1)
800	7	236.1 (4.7, 230.6)	125	81.9 (65.8, 4.2)	87	78.5 (46.4, 3.2, 14.8)	82	59.0 (43.4, 3.1)
1000	7	568.1 (8.1, 558.8)	113	135.8 (113.5, 5.6)	85	130.2 (82.1, 4.4, 22.8)	87	107.1 (82.9, 4.1)
1500	7	1874.7 (28.0, 1844.2)	126	452.7 (403.7, 10.3)	75	335.4 (242.8, 6.5, 44.7)	83	317.8 (267.7, 7.0)
$\epsilon = 10^{-5}$								
	Newton		Algorithm 1		Algorithm 2		Algorithm 3	
n	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2, t_3)	IT.	CT. (t_1, t_2)
100	14	38.2 (0.1, 37.8)	330	6.4 (3.7, 1.8)	201	5.0 (2.1, 1.6, 0.2)	181	4.3 (2.1, 1.5)
200	8	26.4 (0.2, 26.0)	221	9.9 (5.5, 1.9)	132	7.4 (2.9, 1.4, 1.2)	123	5.8 (3.1, 1.5)
500	8	117.7 (1.6, 115.6)	187	42.0 (30.7, 3.6)	136	44.9 (23.4, 2.7, 9.6)	136	33.9 (22.1, 2.7)
800	7	237.4 (4.8, 231.8)	204	132.2 (106.8, 6.2)	135	120.6 (71.8, 4.4, 23.0)	133	94.6 (70.0, 4.5)
1000	7	803.2 (8.2, 793.9)	172	194.7 (163.1, 7.3)	123	188.1 (118.4, 6.3, 33.0)	132	162.5 (126.1, 5.8)
1500	8	2486.9 (31.8, 2452.5)	227	817.0 (730.4, 17.7)	136	609.4 (443.1, 10.9, 81.2)	123	472.2 (398.2, 10.2)

6 Conclusions

This paper demonstrates efficient applications of the alternating direction method of multipliers (ADMM) to the semidefinte inverse quadratic eigenvalue problem (SDIQEP) with partial eigenstructure. The proposed ADMM approach can be easily implementable and coded, and is efficient for large scale cases of SDIQEP. An immediate application of the proposed algorithms is that they can serve as the initial process for second order methods if solutions of SDIQEP in very high accuracy are required.

Table 4: Comparison of ADMM and the generalized Newton approach in [1] for Example 5.1 with $n = 500$.

$\epsilon = 10^{-3}$								
	Newton		Algorithm 1		Algorithm 2		Algorithm 3	
p	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2, t_3)	IT.	CT. (t_1, t_2)
8	6	17.1 (1.3, 15.5)	120	25.3 (19.6, 1.0)	74	22.8 (12.4, 0.6, 5.0)	79	18.5 (13.0, 0.6)
10	7	17.1 (1.4, 15.3)	119	24.8 (19.4, 1.0)	72	21.8 (11.8, 0.7, 4.9)	82	19.0 (13.4, 0.7)
20	7	56.9 (1.4, 55.0)	160	33.7 (26.0, 1.8)	95	29.1 (15.6, 1.1, 6.5)	87	20.5 (14.1, 1.0)
30	9	217.3 (1.8, 214.9)	213	46.4 (34.6, 3.9)	128	40.6 (21.1, 2.7, 8.7)	116	29.2 (19.5, 2.5)
40	7	202.3 (1.4, 200.5)	156	40.5 (25.4, 8.9)	97	36.9 (15.8, 7.7, 6.6)	95	29.7 (15.6, 7.7)
50	9	225.1 (1.7, 222.8)	233	78.2 (37.9, 29.7)	139	70.1 (22.9, 27.1, 9.5)	126	57.5 (20.9, 26.9)
$\epsilon = 10^{-5}$								
	Newton		Algorithm 1		Algorithm 2		Algorithm 3	
p	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2, t_3)	IT.	CT. (t_1, t_2)
8	7	20.4 (1.4, 18.6)	196	40.8 (31.7, 1.6)	135	42.0 (22.9, 1.1, 9.4)	140	33.1 (23.0, 1.1)
10	8	20.1 (1.6, 18.2)	207	42.9 (33.6, 1.8)	122	37.0 (20.0, 1.0, 8.3)	131	30.4 (21.5, 1.2)
20	8	74.6 (1.6, 72.6)	256	53.8 (41.6, 2.7)	153	46.9 (25.3, 1.8, 10.3)	138	32.3 (22.4, 1.6)
30	10	265.7 (1.9, 263.3)	489	105.0 (79.4, 7.4)	299	92.8 (48.8, 4.9, 20.4)	268	63.7 (42.8, 4.7)
40	8	293.5 (1.5, 291.5)	245	59.8 (39.8, 10.4)	147	52.5 (24.2, 8.6, 10.0)	146	41.8 (23.7, 8.8)
50	10	276.1 (1.9, 273.7)	549	149.1 (89.1, 37.7)	337	133.2 (55.3, 32.0, 23.0)	301	101.5 (49.6, 31.6)

Some interesting problems in this scope include how to handle SDIQEP with more component-wise constraints (e.g., sparsity and bandedness, etc) on the involved matrices. Such a problem requires to combine more advanced optimization techniques with the idea of operator splitting methods in algorithmic design. This is our research interest in the future.

References

[1] Z. J. Bai, D. L. Chu, and D. F. Sun, *A dual optimization approach of inverse quadratic eigenvalue problems with partial eigenstructure*, SIAM J. Sci. Comput., 29 (2007), pp. 2531–2561.

[2] E. Blum and W. Oettli, Mathematische Optimierung. Grundlagen und Verfahren. Ökonometrie und Unternehmensforschung, Springer-Verlag, Berlin-Heidelberg-New York, 1975.

[3] R. Boisvert, R. Pozo, K. Remington,, B. Miller, and R. Lipman, *Matrix Market*, National Institute of Standards and Technology, Gaithersburg, 2007.

[4] S. Boyd, N. Parikh, E. Chu, B. Peleato and J. Eckstein, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Found. Trends Machine Learning, 3, 2010, pp. 1–122.

[5] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[6] X. J. Cai, G. Y. Gu, B. S. He, and X. M. Yuan, *A proximal point algorithm revisit on the alternating direction method of multipliers*, Sci. China Math., to appear.

Table 5: Comparison of ADMM and the generalized Newton approach in [1] for Example 5.1 with $n = 1,000$.

$\epsilon = 10^{-3}$								
	Newton		Algorithm 1		Algorithm 2		Algorithm 3	
p	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2, t_3)	IT.	CT. (t_1, t_2)
10	5	82.2 (6.2, 75.1)	93	103.5 (87.8, 2.7)	69	103.6 (65.7, 2.0, 18.8)	75	90.8 (71.4, 2.2)
20	6	160.2 (7.3, 152.0)	97	108.1 (91.1, 3.3)	69	103.8 (65.6, 2.4, 18.8)	71	86.2 (67.4, 2.4)
30	6	563.4 (7.2, 555.1)	118	133.7 (111.7, 5.7)	83	126.3 (78.9, 4.1, 22.5)	85	104.9 (80.9, 4.2)
40	6	370.9 (7.2, 362.6)	126	149.4 (119.4, 11.7)	86	136.7 (81.9, 9.8, 23.3)	89	117.0 (85.9, 10.0)
50	7	893.3 (8.3, 883.9)	153	202.6 (147.0, 31.9)	91	164.3 (86.5, 28.6, 24.5)	83	128.4 (78.9, 28.3)
$\epsilon = 10^{-5}$								
	Newton		Algorithm 1		Algorithm 2		Algorithm 3	
p	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2, t_3)	IT.	CT. (t_1, t_2)
10	6	103.3 (7.2, 95.2)	138	153.4 (130.2, 4.0)	103	154.1 (97.9, 2.9, 27.9)	115	138.5 (108.9, 3.4)
20	7	219.0 (8.2, 209.8)	165	184.3 (156.0, 5.5)	109	163.2 (103.1, 3.6, 29.4)	108	130.8 (102.5, 3.6)
30	7	850.5 (8.3, 841.2)	192	217.0 (182.1, 8.2)	127	192.4 (120.8, 5.6, 34.6)	128	157.3 (121.8, 5.7)
40	7	617.8 (8.2, 608.5)	173	202.3 (163.9, 13.7)	124	194.4 (118.0, 11.8, 33.5)	134	171.4 (127.9, 12.2)
50	8	1238.2 (9.3, 1227.7)	245	303.8 (436.8, 50.4)	166	277.3 (157.1, 32.5, 44.8)	150	210.2 (141.9, 31.7)

[7] M. T. Chu and G. H. Golub, *Inverse Eigenvalue Problems: Theory, Algorithms, and Applications*, Oxford University Press, Oxford, 2005.

[8] R. W. Clough and J. Penzien, *Dynamics of Structures*, McGraw-Hill, Dusseldorf, 1975.

[9] J. Eckstein and D.P. Bertsekas, *On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators*, Math. Program., 55, 1992, pp. 293–318.

[10] M. I. Friswell and J. E. Mottershead, *Finite Element Model Updating in Structural Dynamics*, Kluwer Academic Publishers, 1995.

[11] D. Gabay, *Application of the method of multipliers to variational inequalities*, in Augmented Lagrangian Methods: Application to the Numerical Solution of Boundary-value Problems, M. Fortin and R. Glowinski, eds., North-Holland, Amsterdam, The Netherlands, 1983, pp. 299–331.

[12] D. Gabay, and B, Mercier, *A dual algorithm for the solution of nonlinear variational problems via finite element approximations*, Comput. Math. Appl., 2 (1976), pp. 17–40.

[13] G. M. L. Gladwell, *Inverse Problems in Vibration*, 2nd Edition, Kluwer Academic Publishers, New York, Boston, Dordrecht, London, Moscow, 2004.

[14] R. Glowinski and A. Marrocco, *Approximation par éléments finis d'ordre un et résolution par pénalisation-dualité d'une classe de problèmes non linéaires*, R.A.I.R.O., 1975, pp. 41–76.

[15] R. Glowinski and P. Le Tallec, Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics, SIAM Studies in Applied Mathematics, Philadelphia, PA, 1989.

Table 6: Comparison of ADMM and the generalized Newton approach in [1] for Example 5.2 with $p = 30$ ("*" means the stopping criterion is not satisfied after 5000 seconds).

$\epsilon = 10^{-3}$								
	Newton		Algorithm 1		Algorithm 2		Algorithm 3	
n	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2, t_3)	IT.	CT. (t_1, t_2)
100	6	8.8 (0.1, 8.6)	104	2.6 (1.1, 1.2)	76	2.3 (0.8, 1.1, 0.1)	76	2.2 (0.8, 1.2)
200	7	20.0 (0.2, 19.7)	127	5.4 (3.0, 1.5)	88	5.0 (1.9, 1.4, 0.7)	81	4.0 (1.8, 1.3)
500	6	79.3 (1.2, 77.7)	146	31.3 (22.8, 2.9)	89	30.3 (15.7, 2.2, 6.3)	92	22.7 (14.7, 2.2)
800	7	269.4 (4.6, 264.0)	243	154.2 (124.6, 7.5)	147	129.6 (75.8, 4.9, 25.7)	134	94.0 (69.3, 4.4)
1000	6	449.1 (7.1, 441.0)	140	155.1 (129.3, 6.4)	101	151.0 (93.9, 4.7, 27.3)	100	121.5 (93.6, 4.5)
1500	6	1316.3 (24.6, 1289.5)	102	363.9 (323.9, 8.5)	85	378.3 (272.5, 7.3, 51.8)	88	333.9 (280.4, 7.5)
$\epsilon = 10^{-5}$								
	Newton		Algorithm 1		Algorithm 2		Algorithm 3	
n	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2, t_3)	IT.	CT. (t_1, t_2)
100	7	10.4 (0.0, 10.3)	167	3.6 (1.5, 1.4)	113	2.9 (1.2, 1.2, 0.1)	116	2.8 (1.2, 1.2)
200	8	23.5 (0.1, 23.2)	214	8.2 (4.9, 1.7)	129	6.8 (2.9, 1.5, 1.1)	129	5.7 (3.0, 1.6)
500	7	97.8 (1.5, 95.9)	209	45.0 (33.2, 3.9)	139	44.1 (22.5, 3.0, 9.4)	139	36.1 (24.3, 3.0)
800	8	382.3 (5.1, 376.5)	382	243.5 (197.4, 11.0)	232	203.4 (119.3, 7.2, 40.3)	210	146.1 (108.3, 6.6)
1000	6	448.6 (7.1, 440.5)	208	231.1 (193.2, 8.9)	144	215.3 (134.4, 6.4, 39.0)	140	169.0 (130.3, 6.1)
1500	*	> 5000 (*,*)	146	524.4 (467.3, 11.8)	123	547.0 (394.1, 10.2, 75.1)	125	473.8 (398.4, 10.5)

[16] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd edition, Johns Hopkins University Press, Baltimore and London, 1996.

[17] E. G. Gol'shtein and N. V. Tret'yakov, *Modified Lagrangian in convex programming and their generalizations*, Math. Program. Studies, 10, 1979, pp. 86–97.

[18] B. S. He, L. Z. Liao, D. R. Han and H. Yang, *A new inexact alternating directions method for monontone variational inequalities*, Math. Program., 92, 2002, pp. 103–118.

[19] B. S. He, M. H. Xu, and X. M. Yuan, *Solving large-scale least squares semidefinite probramming by alternating direction methods*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 136–152.

[20] B. S. He and H. Yang, *Some convergence properties of a method of multipliers for linearly constrained monotone variational inequalities*, Oper. Res. Lett., 23 (1998), pp. 151–161.

[21] B. S. He and X. M. Yuan, *On the $O(1/n)$ convergence rate of Douglas-Rachford alternating direction method*, SIAM J. Num. Anal., 50 (2012), pp. 700-709.

[22] M. R. Hestenes, *Multiplier and gradient methods*, J. Optim. Theory Appl., 4 (1969), pp. 303–320.

[23] N. J. Higham, *Computing a nearest symmetric positive semidefinite matrix*, Linear Algebra Appl., 103 (1988), pp. 103–118.

[24] Y. C. Kuo, W. W. Lin, and S. F. Xu, *New methods for finite element model updating problems*, AIAA J. 44 (2006), pp. 1310–1316.

Table 7: Comparison of ADMM and the generalized Newton approach in [1] for Example 5.2 with $n = 500$.

$\epsilon = 10^{-3}$								
	Newton		Algorithm 1		Algorithm 2		Algorithm 3	
p	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2, t_3)	IT.	CT. (t_1, t_2)
8	4	7.1 (0.9, 5.8)	74	15.7 (12.2, 0.8)	70	21.3 (11.6, 0.6, 4.7)	78	18.3 (12.8, 0.6)
10	6	25.4 (1.2, 23.7)	142	29.6 (22.7, 1.3)	86	25.8 (13.9, 0.7, 5.6)	93	21.5 (14.8, 0.8)
20	6	52.5 (1.2, 50.8)	134	27.8 (21.0, 1.5)	95	28.5 (15.3, 1.3, 6.3)	91	21.1 (14.4, 1.0)
30	9	142.8 (1.7, 140.6)	253	53.1 (39.1, 4.3)	154	47.1 (24.2, 3.0, 10.4)	140	33.2 (21.7, 2.8)
40	7	119.6 (1.4, 117.8)	155	39.8 (24.1, 9.1)	105	38.6 (16.4, 8.0, 6.9)	107	32.0 (16.8, 7.9)
50	7	251.7 (1.4, 249.8)	159	61.1 (24.5, 28.2)	98	56.8 (15.4, 26.6, 6.6)	94	48.9 (14.7, 26.4)
$\epsilon = 10^{-5}$								
	Newton		Algorithm 1		Algorithm 2		Algorithm 3	
p	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2, t_3)	IT.	CT. (t_1, t_2)
8	5	9.5 (1.0, 8.1)	118	24.5 (19.0, 0.9)	112	33.8 (18.3, 0.7, 7.4)	116	26.9 (18.7, 0.9)
10	7	28.1 (1.4, 26.3)	242	49.9 (38.9, 2.1)	145	43.3 (23.4, 1.2, 9.6)	147	33.6 (23.3, 1.2)
20	7	67.3 (1.4, 65.5)	209	43.1 (32.9, 2.5)	140	41.9 (22.3, 1.6, 9.2)	138	31.7 (21.6, 1.5)
30	10	184.7 (1.8, 182.3)	420	87.3 (64.7, 6.3)	257	77.7 (40.5, 4.3, 16.9)	232	54.1 (35.9, 4.0)
40	7	119.9 (1.3, 118.1)	230	56.6 (36.7, 10.3)	150	52.5 (23.5 8.9, 10.0)	157	43.7 (24.5, 8.9)
50	7	250.9 (1.3, 249.1)	244	78.9 (37.6, 29.6)	147	72.3 (23.6, 27.5, 10.1)	146	63.8 (23.4, 28.8)

[25] M. M. Lin, B. Dong, and M. T. Chu, *Semi-definite programming techniques for structured quadratic inverse eigenvalue problems*, Numer Algor., 53 (2010), pp. 419–437.

[26] B. Martinet, *Regularision d'inéquations variationnelles par approximations successive*, Revue Francaise d'Automatique et Informatique Recherche Opérationnelle, 126, 1970, pp. 154–159.

[27] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer, New York, 1999.

[28] M. J. D. Powell, *A method for nonlinear constraints in minimization problems*, In Optimization edited by R. Fletcher, pp. 283–298, Academic Press, New York, 1969.

[29] R. T. Rockafellar, *Conjugate Duality and Optimization*, CBMS-NSF Regional Conf. Ser. in Appl. Math. 16, SIAM, Philadelphia, 1974.

[30] R. T. Rockafellar and R. J. B. Wets, *Variational Analysis*, Springer, Berlin, 1998.

[31] J. G. Sun, *Backward perturbation analysis of certain characteristic subspaces*, Numer. Math., 65 (1993), 357–382.

[32] R. H. Tütüncü, K. C. Toh, and M. J. Todd, *Solving semidefinite-quadratic-linear programs using SDPT3*. Math. Program., 95 (2003), pp. 189–217.

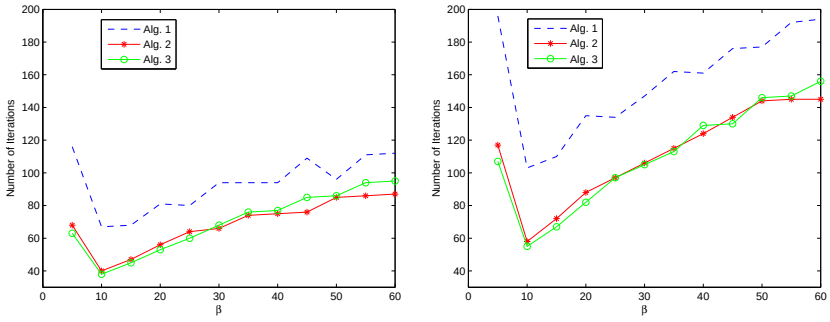
[33] C. H. Ye and X. M. Yuan, *A descent method for structured monotone variational inequalities*, Optim. Methods Soft., 22 (2007), pp. 329–338.

Table 8: Comparison of ADMM and the generalized Newton approach in [1] for Example 5.2 with $n = 1,000$.

$\epsilon = 10^{-3}$								
	Newton		Algorithm 1		Algorithm 2		Algorithm 3	
p	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2, t_3)	IT.	CT. (t_1, t_2)
10	6	73.3 (7.3, 65.0)	162	179.7 (152.5, 4.8)	108	160.6 (102.3, 3.1, 28.9)	105	126.7 (99.6, 3.1)
20	6	341.2 (7.3, 332.9)	142	157.4 (132.9, 4.7)	97	144.7 (91.4, 3.4, 26.1)	99	119.9 (93.8, 3.4)
30	7	564.6 (8.2, 555.3)	187	207.4 (173.1, 7.9)	123	183.6 (114.5, 5.5, 33.7)	114	138.0 (106.4, 5.2)
40	8	1459.7 (9.2, 1449.3)	244	276.0 (224.0, 16.8)	148	227.0 (137.7, 12.6, 39.7)	147	184.0 (136.6, 12.7)
50	7	1297.5 (8.2, 1288.0)	216	267.1 (197.5, 36.7)	133	224.4 (122.2, 31.5, 35.7)	122	175.3 (113.7, 31.0)
$\epsilon = 10^{-5}$								
	Newton		Algorithm 1		Algorithm 2		Algorithm 3	
p	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2)	IT.	CT. (t_1, t_2, t_3)	IT.	CT. (t_1, t_2)
10	7	93.0 (8.3, 83.8)	228	255.6 (217.1, 6.7)	149	222.3 (141.8, 4.3, 40.1)	146	176.5 (139.0, 4.1)
20	7	396.2 (8.3, 386.9)	212	234.3 (197.6, 7.1)	151	225.5 (143.0, 5.1, 40.7)	138	166.5 (130.1, 4.6)
30	7	537.2 (8.2, 527.9)	288	319.9 (267.4, 11.9)	182	271.9 (170.3, 7.9, 49.0)	168	203.0 (156.7, 7.1)
40	8	1460.4 (9.1, 1450.1)	416	468.8 (385.0, 24.8)	254	384.9 (236.6, 17.6, 68.3)	230	284.2 (214.0, 16.5)
50	8	1677.2 (9.1, 1666.8)	333	396.0 (304.6, 42.2)	202	327.0 (185.8, 35.3, 54.0)	184	247.6 (169.0, 33.6)

Figure 1: Numerical results for Example 5.2 with $n = 200$ and $p = 30$.

(a) $\epsilon = 10^{-3}$, $\delta = 1.6$, $\gamma = 1.8$ (b) $\epsilon = 10^{-5}$, $\delta = 1.6$, $\gamma = 1.8$



(c) $\epsilon = 10^{-3}$, $\delta = 1.6$, $\gamma = 1.8$ (d) $\epsilon = 10^{-5}$, $\delta = 1.6$, $\gamma = 1.8$

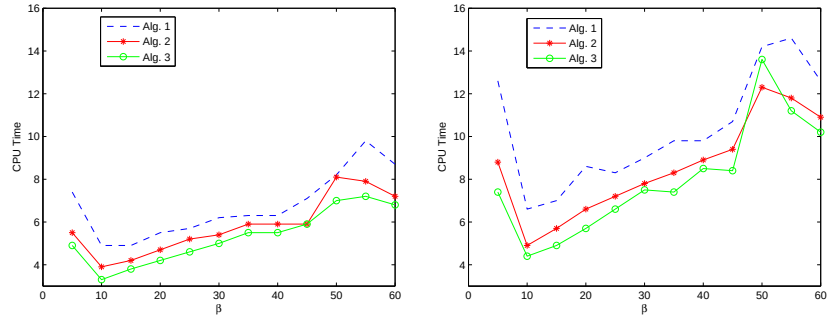
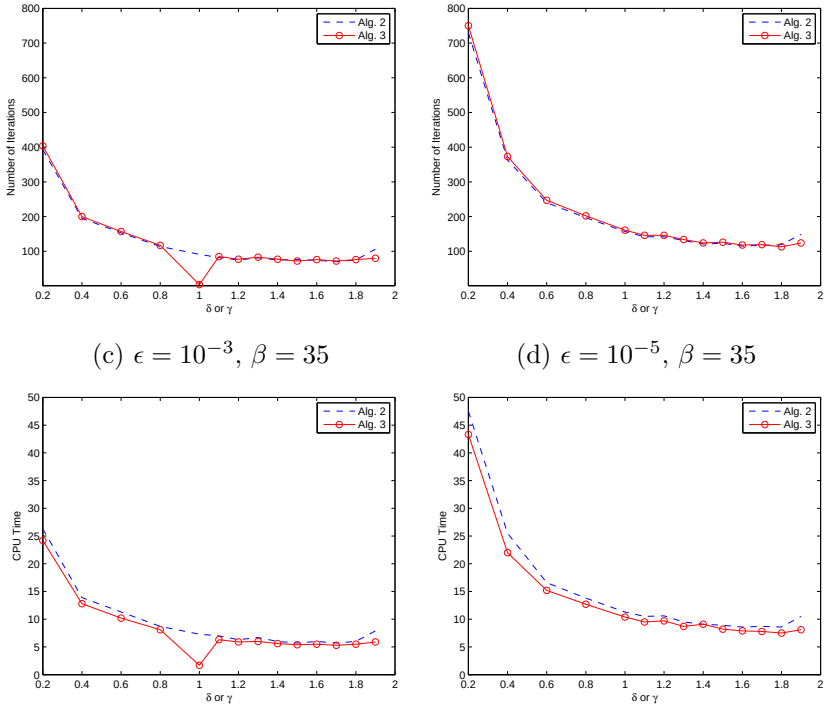


Figure 2: Numerical results for Example 5.2 with $n = 200$ and $p = 30$.

(a) $\epsilon = 10^{-3}$, $\beta = 35$ (b) $\epsilon = 10^{-5}$, $\beta = 35$



(c) $\epsilon = 10^{-3}$, $\beta = 35$ (d) $\epsilon = 10^{-5}$, $\beta = 35$

Table 9: Comparison of ADMM with IPM and the generalized Newton approach in [1] for Examples 5.3–5.5 ("—" means out of memory and "$*$" means the stopping criterion is not satisfied after 20 iterations).

$\epsilon = 10^{-3}$											
		IPM		Newton		Algorithm 1		Algorithm 2		Algorithm 3	
Ex.	β	IT.	CT.	IT.	CT.	IT.	CT.	IT.	CT.	IT.	CT.
5.3(a)	1	—		1	0.3	22	0.2	13	0.1	9	0.1
5.3(b)	800	—		≥ 20	*	76	0.5	51	0.4	14	0.1
5.3(c)	1	—		1	0.2	30	0.2	20	0.2	10	0.1
5.4	80	—		12	196.4	72	13.2	60	15.2	60	11.7
5.5	60	—		1	26.8	11	16.2	8	16.1	4	7.4
$\epsilon = 10^{-5}$											
Ex.	β	IPM		Newton		Algorithm 1		Algorithm 2		Algorithm 3	
5.3(a)	1	—		1	0.3	29	0.1	17	0.1	14	0.1
5.3(b)	800	—		≥ 20	*	86	0.5	54	0.4	17	0.1
5.3(c)	1	—		1	0.2	36	0.2	25	0.2	15	0.1
5.4	80	—		≥ 20	*	219	40.8	180	44.1	182	35.4
5.5	60	—		2	37.2	13	19.2	11	21.4	7	11.7