An inner-outer nonlinear programming approach for constrained quadratic matrix model updating*

M. Andretta † E. G. Birgin‡ M. Raydan§

March 17, 2015

Abstract

The Quadratic Finite Element Model Updating Problem (QFEMUP) concerns with updating a symmetric second-order finite element model so that it remains symmetric and the updated model reproduces a given set of desired eigenvalues and eigenvectors by replacing the corresponding ones from the original model. Taking advantage of the special structure of the constraint set, it is first shown that the QFEMUP can be formulated as a suitable constrained nonlinear programming problem. Using this formulation, a method based on successive optimizations is then proposed and analyzed. To avoid that spurious modes (eigenvectors) appear in the frequency range of interest (eigenvalues) after the model has been updated, additional constraints based on a quadratic Rayleigh quotient are dynamically included in the constraint set. A distinct practical feature of the proposed method is that it can be implemented by computing only a few eigenvalues and eigenvectors of the associated quadratic matrix pencil. The results of our numerical experiments on illustrative problems show that the algorithm works well in practice.

Keywords: Quadratic matrix model updating, quadratic Rayleigh quotient, nonlinear programming, algorithms.

1 Introduction

The Quadratic Finite Element Model Updating Problem (QFEMUP) concerns with updating a finite-element generated model of a vibrating structure of the form:

$$M\ddot{x}(t) + D\dot{x}(t) + Kx(t) = 0, \tag{1}$$

*This work was supported by PRONEX-CNPq/FAPERJ (E-26/111.449/2010-APQ1), FAPESP (grants 2010/10133-0, 2013/03447-6, 2013/05475-7, and 2013/07375-0), and CNPq (PVE 71/2013, proj. 400926/2013-0 and 476792/2013-4).

†Department of Applied Mathematics and Statistics, Institute of Mathematical and Computer Sciences, University of São Paulo, Av. Trabalhador São-carlense, 400, Centro, 13566-590, São Carlos, SP, Brazil. e-mail: andretta@icmc.usp.br

‡Department of Computer Science, Institute of Mathematics and Statistics, University of São Paulo, Rua do Matão, 1010, Cidade Universitária, 05508-090, São Paulo, SP, Brazil. e-mail: egbirgin@ime.usp.br

§Departamento de Cómputo Científico y Estadística, Universidad Simón Bolívar, Ap. 89000, Caracas 1080-A, Venezuela. e-mail: mraydan@usb.ve

where M, D, and K are real $n \times n$ matrices known as mass, damping, and stiffness, respectively; and $\dot{x}(t)$ and $\ddot{x}(t)$ denote the first and second derivatives of the time-dependent vector $x(t)$. The eigenvalues of the associated quadratic pencil

$$Q(\lambda) = \lambda^2 M + \lambda D + K \tag{2}$$

are related to natural frequencies and the eigenvectors are the mode shapes of the vibrating system (1) (see, e.g., [18, 19, 32]). The quadratic pencil (2) has $2n$ eigenvalues and $2n$ eigenvectors. The dynamics of the system are modeled by these eigenvalues and eigenvectors. For example, it is well-known that the stability of a vibrating system is determined by the nature of a few dominating natural frequencies. It is also well-known that sometimes the vibrating structures experience dangerous vibrations, called *resonance*, when a natural frequency becomes close or equal to a frequency of an external force, such as earthquake, gusty wind, weights of the human bodies, among others. Failures of many structures like buildings, bridges, airplane wings, and turbines have been attributed to resonance.

Equation (1) is usually obtained by discretization of a distributed parameter system with finite element techniques, and therefore, known as the finite element model. The matrices M, D, and K are often very large and sparse but have some structure, such as, M is symmetric and positive definite ($M = M^T > 0$) and often diagonal, and D and K are symmetric ($D = D^T$ and $K = K^T$).

The QFEMUP consists in updating the quadratic pencil $Q(\lambda)$ to another quadratic pencil

$$\tilde{Q}(\lambda) = \lambda^2 M + \lambda \tilde{D} + \tilde{K} \tag{3}$$

in such a way that a small number $1 \leq p < 2n$ of given measured eigenvalues and eigenvectors from a real-life structure or an experimental structure are reproduced by the updated pencil. Besides the basic requirements of preserving the symmetry and sparsity pattern of the new matrices $\tilde{D}$ and $\tilde{K}$ and reproducing the p measured eigenvalues and eigenvectors, there are certain other engineering issues that must be taken into account while solving the problem in practice. For instance, it is important that the new matrices $\tilde{D}$ and $\tilde{K}$ are as close as possible to the original ones D and K, respectively, which imposes an optimization approach. It is also very important that no spurious modes appear in the frequency range of interest after the model has been updated; see [28]. The so called no spill-over constraint which, assuming that the p eigenpairs to be replaced are known, forces the additional $2n - p$ eigenvalues and corresponding eigenvectors to remain unchanged, clearly guarantees that no spurious modes will appear in the frequency range of interest. Several numerical schemes have been recently proposed to accomplish all the mentioned requirements, including the no spill-over constraint, for several different scenarios; see, e.g., [6, 12, 13, 14, 15, 16, 17, 21, 22, 25, 26, 32, 33, 34, 39] and references in there. In most cases, the no spill-over constraint is accomplished by using some clever linear algebra theoretical results that involve the solution of several large-scale matrix equations (Lyapunov, Sylvester, and block linear systems); see, e.g., [13, 14, 34, 39].

In several important applications, instead of forcing the no spill-over constraint, what is important from a practical point of view is to guarantee that spurious modes are not introduced into the frequency range of interests (see [28]). For this scenario, we present in this paper a new optimization approach that not only maintains the symmetry, the sparsity structure,

and the nearness of the matrices $\tilde{D}$ and $\tilde{K}$, while reproducing the p measured eigenvalues and eigenvectors, but also pays special attention to the fundamental engineering requirement of making sure that no spurious modes appear in the frequency range of interest. In this work, we accomplish all these requirements without forcing the no spill-over constraint. For that, our new scheme combines an optimization procedure with the dynamical inclusion of additional constraints in an inner-outer iterative scheme. The additional constraints are based on a suitable recent extension of the Rayleigh quotient for quadratic eigenvalue problems [30, 40]. A key practical feature of the proposed scheme is that it can be implemented by computing only a few additional eigenvalues and eigenvectors of the associated quadratic matrix pencil.

Variations of finite element model updating problems, with different levels of difficulty, have been solved in the past using iterative numerical optimization techniques of several types and for different objective functions; see, e.g., [1, 6, 7, 8, 12, 20, 37]. In particular, the ones that guarantee the no spill-over constraint need to solve several large-scale matrix equations for each function and gradient evaluation, which require a computational cost of $O(pn^3)$ floating point operations (flops) per iteration; see, e.g., [6, 12].

The rest of the paper is organized as follows. In Section 2, we formulate the QFEMUP as a constrained optimization problem and describe the variables and the constrains, including the way of forcing the matrices' sparsity structure and symmetry. In Section 3, we describe the suitable use of the Rayleigh quotient for quadratic eigenvalue problems for building the constraints or cuts, to avoid when necessary the presence of spurious modes in the frequency range of interest. We also describe in detail the inner-outer iterative scheme, and discuss its theoretical properties. In Section 4, we show the performance of our scheme on some illustrative examples. Concluding remarks are presented in Section 5.

2 Mathematical programming formulation

Consider the quadratic pencil $Q(\lambda)$ given by (2), where $M, D, K \in \mathbb{R}^{n\times n}$ are given matrices such that M is symmetric positive definite and D and K are symmetric. Let $1 \leq p < 2n$, $\lambda_i \in \mathbb{C}$, and $x_i \in \mathbb{C}^n$ be such that (λ_i, x_i) for $i = 1, \ldots, p$ are the desired eigenpairs. The goal is to find matrices $\tilde{D}, \tilde{K} \in \mathbb{R}^{n\times n}$ such that (λ_i, x_i) are eigenpairs of the updated quadratic eigenpencil $\tilde{Q}(\lambda)$ given by (3), i.e.,

$$(\lambda_i^2 M + \lambda_i \tilde{D} + \tilde{K})x_i = 0,\ i = 1, \ldots, p. \tag{4}$$

Matrices $\tilde{D} = (\tilde{d}_{ij})$ and $\tilde{K} = (\tilde{k}_{ij})$ must be symmetric and must preserve the sparsity pattern of $D = (d_{ij})$ and $K = (k_{ij})$, respectively. In addition, $\tilde{D}$ and $\tilde{K}$ must be as close as possible to D and K, respectively.

Let I_D and I_K be the sets of indexes of non-zero elements in the upper triangle of the given matrices D and K, respectively, i.e.,

$$I_D = \{(i, j) \mid 1 \leq i \leq j \leq n \text{ such that } d_{ij} \neq 0\} \tag{5}$$

and

$$I_K = \{(i, j) \mid 1 \leq i \leq j \leq n \text{ such that } k_{ij} \neq 0\}. \tag{6}$$

In our setting, elements $\tilde{d}_{ij}$ with $(i, j) \in I_D$ are the unknown elements or variables of the desired matrix $\tilde{D}$. For the remaining elements of $\tilde{D}$ we have that: (a) if $i > j$ and $(j, i) \in I_D$

then $\tilde{d}_{ij} = \tilde{d}_{ji}$ in order to preserve symmetry, and (b) if neither (i,j) nor (j,i) are in I_D then $\tilde{d}_{ij} = \tilde{d}_{ji} = 0$ in order to preserve the desired sparsity pattern. Similarly, the unknowns related to the matrix $\tilde{K}$ are $\tilde{k}_{ij}$ with $(i,j) \in I_K$. It is assumed that a non-negative real number U such that $-U \leq \tilde{d}_{ij} \leq U$ for all $(i,j) \in I_D$ and $-U \leq \tilde{k}_{ij} \leq U$ for all $(i,j) \in I_K$ is known.

Summing up, the mathematical programming formulation of the QFEMUP is given by

$$\begin{array}{ll} \text{Minimize} & \|\tilde{D} - D\|_F^2 + \|\tilde{K} - K\|_F^2 \\ \text{subject to} & M X \Lambda^2 + \tilde{D} X \Lambda + \tilde{K} X = 0, \\ & -U \leq \tilde{d}_{ij} \leq U \ \text{ for all } \ (i,j) \in I_D, \\ & -U \leq \tilde{k}_{ij} \leq U \ \text{ for all } \ (i,j) \in I_K, \end{array} \tag{7}$$

where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_p) \in \mathbb{C}^{p \times p}$ and $X \in \mathbb{C}^{n \times p}$ has columns $x_1, \ldots, x_p$.

Note that in (7) the variables are $\tilde{d}_{ij}$ with $(i,j) \in I_D$ and $\tilde{k}_{ij}$ with $(i,j) \in I_K$. Hence the number of variables is $n_{\mathrm{nz}} = |I_D| + |I_K|$, and the sparsity pattern and symmetry of D and K are preserved because

$$\tilde{d}_{ij} = \begin{cases} 0, & \text{if } i \leq j \text{ and } (i,j) \notin I_D, \\ \tilde{d}_{ji}, & \text{if } i > j, \end{cases} \quad \text{and} \quad \tilde{k}_{ij} = \begin{cases} 0, & \text{if } i \leq j \text{ and } (i,j) \notin I_K, \\ \tilde{k}_{ji}, & \text{if } i > j. \end{cases}$$

This means that the symmetry and the sparsity pattern of $\tilde{D}$ and $\tilde{K}$ are guaranteed by a clever choice of the variables, and that those requirements do not need to be considered as explicit constraints, thus reducing the number of constraints as well. Moreover, note that in the objective function in (7) we have

$$\|\tilde{D} - D\|_F^2 = \sum_{(i,i) \in I_D} (\tilde{d}_{ii} - d_{ii})^2 + 2 \sum_{\substack{(i,j) \in I_D \\ i \neq j}} (\tilde{d}_{ij} - d_{ij})^2,$$

since the remaining terms are zero, and, similarly,

$$\|\tilde{K} - K\|_F^2 = \sum_{(i,i) \in I_K} (\tilde{k}_{ii} - k_{ii})^2 + 2 \sum_{\substack{(i,j) \in I_K \\ i \neq j}} (\tilde{k}_{ij} - k_{ij})^2.$$

Therefore, since the number of variables, the number of constraints, and the computational complexity of evaluating the objective function and the constraints in (7) are of the order of the number of non-zero entries in M, D, and K, the model (7) is suitable for solving potentially large-sized instances of the QFEMUP.

The mathematical programming formulation (7) is equivalent (in the sense of providing the same solution) to the approach based on alternating projection methods introduced in [37], the novelty of the former being the possibility of dealing in an efficient way with the requirement of avoiding spurious modes in the frequency range of interest after the model has been updated, as will be described in Section 3.

A remark on the way we tackled the requirement "having the new matrices $\tilde{D}$ and $\tilde{K}$ as close as possible to the original ones D and K" is in order. The objective function in (7) minimizes

the Frobenius norm of the difference between $(\tilde{D}, \tilde{K})$ and (D, K). The squaring of the Frobenius norm has no effect in this case other than providing differentiability of the objective function. However, having $(\tilde{D}, \tilde{K})$ as close as possible to (D, K) is not exactly the same as having $\tilde{D}$ as close as possible to D and *at the same time* $\tilde{K}$ as close as possible to K. This bi-objective problem (see, for example, [36]) would be much more difficult to be solved (since computing the Pareto frontier [36] would be needed). An alternative would be to minimize the sum of those distances, i.e., $\|\tilde{D} - D\| + \|\tilde{K} - K\|$, where $\|\cdot\|$ is an arbitrary norm. Note that squaring the norms would transform the problem into a different problem and not squaring the norms would make, for example, the case in which the Frobenius norm is considered, a non-differentiable problem. Another alternative would be to minimize the largest between $\|\tilde{D} - D\|$ and $\|\tilde{K} - K\|$ for any arbitrary norm, i.e.,

$$\begin{array}{lrcl} \text{Minimize} & z & & \\ \text{subject to} & \|\tilde{D} - D\| & \leq & z, \\ & \|\tilde{K} - K\| & \leq & z, \\ & MX\Lambda^2 + \tilde{D}X\Lambda + \tilde{K}X & = & 0, \\ & -U \leq \tilde{d}_{ij} \leq U \text{ for all } (i,j) & \in & I_D, \\ & -U \leq \tilde{k}_{ij} \leq U \text{ for all } (i,j) & \in & I_K. \end{array} \tag{8}$$

If the Frobenius norm is considered in (8), squaring both sides in the first two inequalities does not alter the problem and it becomes continuous and differentiable. All these alternatives correspond to different interpretations of the nearness requirement and they probably have different solutions. In the present work we consider the objective function in (7), which is the usual interpretation of the nearness requirement considered in the literature (see, for example, [28] and the references therein).

Finally, a remark on the hyperplane constraint in (7) is also in order. Note that $\lambda_i \in \mathbb{C}$ and $x_i \in \mathbb{C}^n$ for $i = 1, \ldots, p$. Therefore, to deal with regular nonlinear programming solvers, that handle problems in the real (non-complex) space, it would be adequate to re-write the constraint as:

$$\begin{aligned} \Re(MX\Lambda^2 + \tilde{D}X\Lambda + \tilde{K}X) &= 0, \\ \Im(MX\Lambda^2 + \tilde{D}X\Lambda + \tilde{K}X) &= 0, \end{aligned}$$

where $\Re(c)$ and $\Im(c)$ represent the real part a and the imaginary part b of a complex number $c = a + bi$, arriving to the formulation

$$\begin{array}{lrcl} \text{Minimize} & \|\tilde{D} - D\|_F^2 + \|\tilde{K} - K\|_F^2 & & \\ \text{subject to} & \Re(MX\Lambda^2 + \tilde{D}X\Lambda + \tilde{K}X) & = & 0, \\ & \Im(MX\Lambda^2 + \tilde{D}X\Lambda + \tilde{K}X) & = & 0, \\ & -U \leq \tilde{d}_{ij} \leq U \text{ for all } (i,j) & \in & I_D, \\ & -U \leq \tilde{k}_{ij} \leq U \text{ for all } (i,j) & \in & I_K. \end{array} \tag{9}$$

Note that (9) is a continuous and differentiable nonlinear programming problem for which any off-the-shelf nonlinear programming method may be applied.

3 Nonlinear cuts and algorithmic framework

In this section, we consider the extra requirement of avoiding that spurious modes (eigenvectors) appear in the updated quadratic eigenpencil (3). An eigenvalue of (3) is called *unstable* if its corresponding eigenvector is a spurious mode. For simplicity of exposition, we will focus on the frequent situation in which $\hat{\lambda} \in \mathbb{R}$ is given (the reader can think of a "small" and negative value for $\hat{\lambda}$ but any value is possible) and there is a constraint that says that all eigenvalues of the updated eigenpencil (3) must have their real part less than or equal to $\hat{\lambda}$.

Let us assume that $(\mu_1, y_1), \ldots, (\mu_{2n}, y_{2n})$ are the (unknown) eigenpairs of the original quadratic eigenpencil (2) and that $1 \leq p < 2n$ and $(\lambda_1, x_1), \ldots, (\lambda_p, x_p)$ are the desired eigenpairs. In addition, let us assume that the p eigenpairs that must be substituted *are known* and that (without loss of generality) those eigenpairs are the first p eigenpairs of (2), i.e., $(\mu_1, y_1), \ldots, (\mu_p, y_p)$. In what follows we will also assume that

(a) the p desired eigenpairs are such that $\Re(\lambda_i) \leq \hat{\lambda}$ for $i = 1, \ldots, p$,

(b) the p eigenpairs that will be substituted are such that $\Re(\mu_i) > \hat{\lambda}$ for $i = 1, \ldots, p$,

(c) the remaining $2n - p$ eigenpairs are such that $\Re(\mu_i) \leq \hat{\lambda}$ for $i = p + 1, \ldots, 2n$.

The requirement of preserving the *unknown* $2n - p$ eigenpairs $(\mu_{p+1}, y_{p+1}), \ldots, (\mu_{2n}, y_{2n})$ of the original quadratic eigenpencil (2) as eigenpairs of the updated quadratic eigenpencil (3) is known as no spill-over constraint. The methodologies, already developed, that impose the no spill-over constraint certainly avoids the advent of spurious modes in the updated quadratic eigenpencil, at the price of solving several large-scale Lyapunov, Sylvester, and block linear systems per iteration.

In the present work, we address the extra requirement of avoiding that spurious modes appear in the updated quadratic eigenpencil (3) using a different approach. Solving the mathematical programming problem (9), we find matrices $\tilde{D}$ and $\tilde{K}$ such that the updated quadratic eigenpencil (3) has the desired eigenpairs $(\lambda_1, x_1), \ldots, (\lambda_p, x_p)$ for $i = 1, \ldots, p$, i.e., such that (4) holds. To achieve this goal, none of the eigenpairs $(\mu_1, y_1), \ldots, (\mu_{2n}, y_{2n})$ of the original quadratic eigenpencil (2) are assumed to be known. It is the nearness requirement, expressed in the minimization of the distance between $(\tilde{D}, \tilde{K})$ and (D, K), that helps to safeguard as much as possible the eigenpairs of the original quadratic eigenpencil. Then, an eigenvalue with largest real part of the updated quadratic eigenpencil (3) is computed. If its real part is larger than $\hat{\lambda}$, then a (normalized) associated eigenvector $\tilde{x}$ is computed; a constraint, that depends on $\tilde{x}$, is added to the mathematical programming model (9) with the attempt of avoiding the detected unstable eigenvalue; and a new nonlinear programming problem is solved. This iterative process is repeated until the updated quadratic eigenpencil has no unstable eigenvalues. The nature of the constraint or cut that is added to the mathematical programming model is discussed below. The idea of generating additional constraints or cuts from standard (non-quadratic) eigenvectors was used to solve Lyapunov equations by Geromel [29] and to solve constrained least-squares matrix problems by Hu [31].

Assume that an eigenvalue $\tau \in \mathbb{C}$ with largest real part of the updated eigenpencil (3) has been computed and that $\Re(\tau) > \hat{\lambda}$. Then, a normalized associated eigenvector $\tilde{x} \in \mathbb{C}^n$ is computed. From the Galerkin condition (see [30, 40]), it follows that τ must be one of the two

solutions of the quadratic equation

$$q(\theta) = \theta^2(\tilde{x}^* M \tilde{x}) + \theta(\tilde{x}^* \tilde{D} \tilde{x}) + (\tilde{x}^* \tilde{K} \tilde{x}),$$

that are given by

$$\theta_1(\tilde{D}, \tilde{K}, \tilde{x}) = \frac{-(\tilde{x}^* \tilde{D} \tilde{x}) + \sqrt{(\tilde{x}^* \tilde{D} \tilde{x})^2 - 4(\tilde{x}^* M \tilde{x})(\tilde{x}^* \tilde{K} \tilde{x})}}{2(\tilde{x}^* M \tilde{x})}$$

and

$$\theta_2(\tilde{D}, \tilde{K}, \tilde{x}) = \frac{-(\tilde{x}^* \tilde{D} \tilde{x}) - \sqrt{(\tilde{x}^* \tilde{D} \tilde{x})^2 - 4(\tilde{x}^* M \tilde{x})(\tilde{x}^* \tilde{K} \tilde{x})}}{2(\tilde{x}^* M \tilde{x})}.$$

If $\tau = \theta_1(\tilde{D}, \tilde{K}, \tilde{x})$ then the new constraint is given by

$$\Re\left(\theta_1(\tilde{D}, \tilde{K}, \tilde{x})\right) \leq \hat{\lambda} - \varepsilon, \tag{10}$$

where $\varepsilon > 0$ is a given small tolerance, which is subtracted from $\hat{\lambda}$ to slightly overstate what we want to achieve. Otherwise, we have that $\tau = \theta_2(\tilde{D}, \tilde{K}, \tilde{x})$ and then the new constraint is given by

$$\Re\left(\theta_2(\tilde{D}, \tilde{K}, \tilde{x})\right) \leq \hat{\lambda} - \varepsilon. \tag{11}$$

Recall that when solving problem (9) with the additional constraint (10) or the additional constraint (11), we have that the vector $\tilde{x}$ is given, as well as the matrix M, and so the variables are the elements $\tilde{d}_{ij}$ of $\tilde{D}$ with $(i,j) \in I_D$ and the elements $\tilde{k}_{ij}$ of $\tilde{K}$ with $(i,j) \in I_K$. The complete iterative procedure is described below.

Algorithm 3.1. Let $M, D, K \in \mathbb{R}^{n \times n}$ be given matrices such that M is positive definite and $D = (d_{ij})$ and $K = (k_{ij})$ are symmetric. Let $1 \leq p < 2n$ and let $(\lambda_i, x_i) \in \mathbb{C} \times C^n$ for $i = 1, \ldots, p$ be the desired quadratic eigenpairs. Let $\hat{\lambda} \in \mathbb{R}$ be a parameter that describes the forbidden region for the quadratic eigenvalues of the updated quadratic eigenpencil. Let $U > 0$ be a given large real number and let $\varepsilon > 0$ be a given small tolerance. Set $\kappa \leftarrow 0$.

Step 1. *Set the sparsity patterns*

Compute I_D and I_K given by (5) and (6), respectively.

Step 2. *Optimization step*

By solving the nonlinear programming problem, with $n_{\text{nz}} = |I_D| + |I_K|$ variables, given by (9) plus

$$\begin{cases} \Re\left(\theta_1(\tilde{D}, \tilde{K}, u_j)\right) \leq \hat{\lambda} - \varepsilon \;\text{ if } s_j = 1, \text{ for } j = 0, \ldots, \kappa - 1, \\ \Re\left(\theta_2(\tilde{D}, \tilde{K}, u_j)\right) \leq \hat{\lambda} - \varepsilon \;\text{ if } s_j = -1, \text{ for } j = 0, \ldots, \kappa - 1, \end{cases} \tag{12}$$

where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_p) \in \mathbb{C}^{p\times p}$ and $X \in \mathbb{C}^{n\times p}$ has columns $x_1, \ldots, x_p$, find matrices $\tilde{D}_\kappa$ and $\tilde{K}_\kappa$ such that

$$(\lambda_i^2 M + \lambda_i \tilde{D}_\kappa + \tilde{K}_\kappa)x_i = 0,\ i = 1, \ldots, p.$$

Step 3. *Check for spurious quadratic eigenvalues*

Step 3.1. Compute a quadratic eigenvalue τ with largest real part of the updated quadratic eigenpencil

$$\tilde{Q}(\lambda) = \lambda^2 M + \lambda \tilde{D}_\kappa + \tilde{K}_\kappa.$$

Step 3.2. If $\Re(\tau) \leq \hat{\lambda}$ then stop returning $\tilde{D}_\kappa$ and $\tilde{K}_\kappa$.

Step 4. *Add a new cut and iterate*

Step 4.1. Compute a quadratic eigenvector u_κ associated with τ (such that $\|u_\kappa\|_2 = 1$).

Step 4.2. If $\tau = \theta_1(\tilde{D}_\kappa, \tilde{K}_\kappa, u_\kappa)$, set $s_\kappa = 1$. Otherwise, if $\tau = \theta_2(\tilde{D}_\kappa, \tilde{K}_\kappa, u_\kappa)$, set $s_\kappa = -1$.

Step 4.3. Set $\kappa \leftarrow \kappa + 1$ and go to Step 2.

Remark. Notice that, for any iteration κ, the feasible region of the nonlinear programming problem solved at Step 2 is a closed subset of the closed and bounded set $B_D \times B_K$, where

$$B_D = \{D \in \mathbb{R}^{n\times n} |\ D = D^T, -U \leq d_{ij} \leq U \ \text{ if } \ (i,j) \in I_D, \text{ and } d_{ij} = 0 \ \text{ if } \ (i,j) \notin I_D\}$$

and

$$B_K = \{K \in \mathbb{R}^{n\times n} |\ K = K^T, -U \leq k_{ij} \leq U \ \text{ if } \ (i,j) \in I_K, \text{ and } k_{ij} = 0 \ \text{ if } \ (i,j) \notin I_K\}.$$

Notice also that at each iteration $\kappa \geq 1$ all the cuts in (12) from previous iterations are kept as constraints for the current nonlinear programming problem. If all these cuts were not kept in the current optimization problem, the algorithm could explore subsets of the original feasible region that were already explored, and as a consequence it could cycle indefinitely without converging to the solution of the problem. In other words, keeping them around guarantees that the feasible region of the nonlinear programming problem is monotonically reduced when κ increases. Our next theorem establishes that Algorithm 3.1 terminates in a finite number of iterations.

Theorem 3.1 *Let $\hat{\lambda} \in \mathbb{R}$ and $\varepsilon > 0$ be given. If Algorithm 3.1 is applied to solve the optimization problem (9) with the additional constraint that all eigenvalues of the updated eigenpencil (3) must have their real part less than or equal to $\hat{\lambda}$, then it terminates in a finite number of iterations.*

Proof. Let us consider the functions $\widehat{\theta_1} : B_D \times B_K \times S_u \to \mathbb{R}$ and $\widehat{\theta_2} : B_D \times B_K \times S_u \to \mathbb{R}$, where $S_u = \{u \in \mathbb{C}^n |\ \|u\|_2 = 1\}$ is the unit sphere in $\mathbb{C}^n$, that define the possible cuts in (12)

$$\widehat{\theta_1}(\tilde{D}, \tilde{K}, u) = \Re\left(\theta_1(\tilde{D}, \tilde{K}, u)\right) \quad \text{and} \quad \widehat{\theta_2}(\tilde{D}, \tilde{K}, u) = \Re\left(\theta_2(\tilde{D}, \tilde{K}, u)\right).$$

Since M is symmetric and positive definite, and $\|u\|_2 = 1$, then $u^*Mu \geq \lambda_{\min}(M) > 0$, where $\lambda_{\min}(M) > 0$ is the smallest eigenvalue of M. Hence, in $\theta_1(\tilde{D}, \tilde{K}, u)$ and $\theta_2(\tilde{D}, \tilde{K}, u)$ the denominator is uniformly bounded away from zero, and as a consequence the functions $\widehat{\theta}_1$ and $\widehat{\theta}_2$ are continuous on $B_D \times B_K \times S_u$. Moreover, since $B_D \times B_K \times S_u$ is a compact set, then the functions $\widehat{\theta}_1$ and $\widehat{\theta}_2$ are uniformly continuous on $B_D \times B_K \times S_u$. Therefore, for the given $\varepsilon > 0$, there exists $\delta_1 > 0$ such that

$$\|[\tilde{D}' : \tilde{K}' : u'] - [\tilde{D} : \tilde{K} : u]\|_F < \delta_1 \tag{13}$$

implies that

$$|\widehat{\theta}_1(\tilde{D}', \tilde{K}', u') - \widehat{\theta}_1(\tilde{D}, \tilde{K}, u)| < \varepsilon, \tag{14}$$

for all $(\tilde{D}', \tilde{K}', u')$ and $(\tilde{D}, \tilde{K}, u)$ in $B_D \times B_K \times S_u$. Similarly, for the given $\varepsilon > 0$ there exists $\delta_2 > 0$ such that the same implication involving (13) and (14) is obtained, but now using $\delta_2 > 0$ and $\widehat{\theta}_2$ instead of $\delta_1 > 0$ and $\widehat{\theta}_1$. In here, $[\tilde{D} : \tilde{K} : u]$ denotes a block matrix which is built stacking the matrices $\tilde{D}$, $\tilde{K}$, and the vector u, i.e., it is a matrix with $2n + 1$ columns and n rows.

For each iteration κ, if the algorithm does not stop at Step 3.2, it sets $s_\kappa = 1$ or $s_\kappa = -1$. It means that exactly one of the two functions $\widehat{\theta}_1$ or $\widehat{\theta}_2$ is chosen to build a new cut at Step 4. Hence, Algorithm 3.1 generates two subsequences of iterations identified with $J_{\widehat{\theta}_1} \subset \mathbb{N}$ and $J_{\widehat{\theta}_2} \subset \mathbb{N}$, which are the sets of indices κ associated with the iterations for which $\widehat{\theta}_1$ is chosen, and the indices for which $\widehat{\theta}_2$ is chosen, respectively. Hence, once κ iterations have been performed (numbered from 0 to $\kappa - 1$), it follows that $J_{\widehat{\theta}_1} \cap J_{\widehat{\theta}_2} = \emptyset$ and $J_{\widehat{\theta}_1} \cup J_{\widehat{\theta}_2} = \{0, 1, \ldots, \kappa - 1\}$.

Let us suppose, by way of contradiction, that $J_{\widehat{\theta}_1}$ is an infinite set of indices. In that case, Algorithm 3.1 generates a sequence $\{u_\kappa\}$ in S_u for $\kappa \in J_{\widehat{\theta}_1}$. Since S_u is compact then there exists an accumulation point of that sequence in S_u. Hence, for $\delta_1 > 0$ there exist u_{i_1} and u_{i_2} with $i_1 < i_2$ and $i_1, i_2 \in J_{\widehat{\theta}_1}$, satisfying

$$\|[\tilde{D}_{i_2} : \tilde{K}_{i_2} : u_{i_1}] - [\tilde{D}_{i_2} : \tilde{K}_{i_2} : u_{i_2}]\|_F = \|u_{i_1} - u_{i_2}\|_2 < \delta_1. \tag{15}$$

Now, since $i_1 < i_2$, matrices $\tilde{D}_{i_2}$ and $\tilde{K}_{i_2}$ computed at iteration i_2 satisfy the constraint in (12) with $j = i_1$ given by

$$\widehat{\theta}_1(\tilde{D}_{i_2}, \tilde{K}_{i_2}, u_{i_1}) \leq \hat{\lambda} - \varepsilon. \tag{16}$$

However, since Algorithm 3.1 does not stop at iteration i_2 (Step 3.2) and $i_2 \in J_{\widehat{\theta}_1}$, this means that u_{i_2} is such that

$$\widehat{\theta}_1(\tilde{D}_{i_2}, \tilde{K}_{i_2}, u_{i_2}) > \hat{\lambda}. \tag{17}$$

Clearly, (15), (16), and (17) contradict (13) and (14), the uniform continuity of $\widehat{\theta}_1$ on $B_D \times B_K \times S_u$. Therefore the number of indices in the set $J_{\widehat{\theta}_1}$ is finite, say N_1.

Let us now suppose, by way of contradiction, that $J_{\widehat{\theta}_2}$ is an infinite set of indices. Repeating the same sequence of arguments as before but now using $\delta_2 > 0$ and $\widehat{\theta}_2$ instead of $\delta_1 > 0$ and $\widehat{\theta}_1$, we conclude that the number of indices in the set $J_{\widehat{\theta}_2}$ is also finite, say N_2. Thus, Algorithm 3.1 terminates in a finite number of iterations $\widehat{N} = N_1 + N_2$, and the result is established. □

Notice that when Algorithm 3.1 terminates, at Step 3.2, matrices $\tilde{D}_{\widehat{N}}$ and $\tilde{K}_{\widehat{N}}$ satisfy all the constraints in (9) plus (12). Moreover, the real part of all the eigenvalues λ of the quadratic eigenpencil $\lambda^2 M + \lambda \tilde{D}_{\widehat{N}} + \tilde{K}_{\widehat{N}}$ are less than or equal to $\hat{\lambda}$. Therefore, Algorithm 3.1 terminates at a feasible solution of (9) that also satisfies the extra requirement of not having spurious modes, which is optimal for problem (9,12).

4 Numerical experiments

To give further insight into the new approach of dynamically adding quadratic Rayleigh quotient cuts to the nonlinear programming setting, for solving the QFEMUP, we present the results of some numerical experiments. We implemented Algorithm 3.1 in Fortran 90. All tests were conducted on a computer with 4 Intel Core i7-3417U 1.9GHz processors and 4GB of RAM memory, running GNU/Linux operating system (Ubuntu 4.8.2-19ubuntu1, kernel 3.13.0-32). Codes were compiled by the GFortran Fortran compiler of GCC (version 4.8.2) with the -O3 optimization directive enabled.

At Step 2 of Algorithm 3.1, the optimization subproblems given by (9,12) were solved using the nonlinear programming solver Algencan [2, 3, 11]. Algencan version 3.0.0, available for download at the TANGO Project web page (`http://www.ime.usp.br/~egbirgin/tango/`) was considered. All parameters were used with their default values, while feasibility and optimality tolerances were both set to 10^{-4}. Algencan is an augmented Lagrangian method for nonlinear programming that solves the bound-constrained augmented Lagrangian subproblems using Gencan [9, 10, 4], an active-set method for bound-constrained minimization. The initial guess for the (iterative process of solving the) nonlinear programming subproblems is always given by the original matrices D and K.

At Step 3, a quadratic eigenvalue τ with largest real part and, when required, an associated eigenvector, are computed using subroutines `dnaupd` and `dneupd` from ARPACK [35]. ARPACK subroutines, based on implicitly restarted Arnoldi methods, are applied to compute an eigenvalue with largest real part of the linearization given by applying the substitution $v = \lambda x$ in $(\lambda^2 M + \lambda \tilde{D} + \tilde{K})x = 0$, that yields the generalized eigenvalue problem

$$\begin{pmatrix} 0 & I \\ -\tilde{K} & -\tilde{D} \end{pmatrix} \begin{pmatrix} x \\ v \end{pmatrix} - \lambda \begin{pmatrix} I & 0 \\ 0 & M \end{pmatrix} \begin{pmatrix} x \\ v \end{pmatrix} = 0. \tag{18}$$

Other linearizations are possible and the most adequate choice depends on the nonsingularity of M and $\tilde{K}$ and, in the large-scale case, on the sparsity structure of the matrices. See [40, pp. 252–253] for details.

Concerning the computational cost per iteration of Algorithm 3.1, notice that evaluating at Step 2 the constraints, and also the objective function and its gradient, requires $O(n_{\mathrm{nz}})$ flops. Since the considered internal numerical optimization scheme in Algencan is gradient related, the iterative process that is applied at Step 2 uses $O(n_{\mathrm{nz}})$ flops per iteration. In addition, at Step 3, a few matrix-vector products involving the matrices in (18) are required. The cost of these matrix-vector products is $O(n_{\mathrm{nz}})$. Note that, in the large-scale case, M, D, and K are expected to be sparse and, therefore, n_{nz} to be a small multiple of n (the dimension of the space), i.e., $n_{\mathrm{nz}} = O(n)$. If, instead of using a gradient related scheme at Step 2, a quasi-Newton scheme

(e.g., inverse BFGS) or a Newton scheme is used, then the cost per iteration increases from $O(n_{\rm nz})$ to $O(n_{\rm nz}^2)$ or to $O(n_{\rm nz}^3)$ flops but the expected number of iteration required to solve the optimization problem at Step 2 could be drastically reduced; see [11].

The performance of Algorithm 3.1 will be illustrated by analyzing its behavior on three small and one medium-sized randomly generated numerical test examples. In all examples, we set $U = 10^{20}$ and $\varepsilon = 2 \times 10^{-4}$. The three small examples were solved in a fraction of a second, and so elapsed CPU times required to solve them are not reported.

4.1 Example 1.

In the first example, we considered the quadratic eigenpencil (2) with matrices

$$M = \begin{pmatrix} 0.7110 & 0.0212 & -0.5813 \\ 0.0212 & 0.8509 & 0.4498 \\ -0.5813 & 0.4498 & 1.7045 \end{pmatrix}, \quad D = \begin{pmatrix} 0.1167 & 0.3240 & 0.0237 \\ 0.3240 & 0.2774 & 0.6079 \\ 0.0237 & 0.6079 & 2.0967 \end{pmatrix},$$

and

$$K = \begin{pmatrix} 0.3521 & 0.0222 & 0.2350 \\ 0.0222 & -0.0007 & 0.0544 \\ 0.2350 & 0.0544 & 1.0708 \end{pmatrix}.$$

The objective is to find symmetric matrices $\tilde{D}$ and $\tilde{K} \in \mathbb{R}^{3\times 3}$ such that the modified eigenpencil (3) has $\lambda_1 = -0.1$ as quadratic eigenvalue associated with the quadratic eigenvector $x_1 = (0.09, -1.00, 0.07)^T$. In addition, we would like that all quadratic eigenvalues of the modified eigenpencil have its real part not larger that $\hat{\lambda} = -0.1$. Matrices D and K are dense and so there is no sparsity pattern to be preserved and the nonlinear subproblemas at Step 2 have $n_{\rm nz} = 12$ variables.

We now describe the application of Algorithm 3.1 to this example. By solving the nonlinear programming subproblem (9) at Step 2 (note that there are no constraints of type (12) in this first iteration), we obtain matrices $\tilde{D}_0$ and $\tilde{K}_0$ such that $\|D - \tilde{D}_0\|_F^2 = 0.0002$ and $\|K - \tilde{K}_0\|_F^2 = 0.0205$. At Step 3, we obtain that $\tau = -0.0712 \not\leq -0.1 = \hat{\lambda}$. Therefore, at Step 4, we compute a normalized eigenvector u_0 associated with τ given by $u_0 = (0.1106, -0.9909, 0.0762)^T$. In addition, by the test given at Step 4.2, we set $s_0 = 1$. The values u_0 and s_0 define a cut of type (12). In the next iteration ($\kappa = 1$), by solving subproblem (9,12), we obtain matrices

$$\tilde{D}_1 = \begin{pmatrix} 0.1181 & 0.3151 & 0.0251 \\ 0.3151 & 0.3053 & 0.5974 \\ 0.0251 & 0.5974 & 2.0980 \end{pmatrix} \text{ and } \tilde{K}_1 = \begin{pmatrix} 0.3420 & 0.0767 & 0.2237 \\ 0.0767 & 0.0317 & 0.1353 \\ 0.2237 & 0.1353 & 1.0593 \end{pmatrix},$$

such that $\|D - \tilde{D}_1\|_F^2 = 0.0012$ and $\|K - \tilde{K}_1\|_F^2 = 0.0206$. This time, when computing a quadratic eigenvalue τ with largest real part, we verify that $\tau = -0.1 = \hat{\lambda}$. This means that we have obtained the desired modified eigenpencil with no unstable quadratic eigenvalues.

4.2 Example 2.

In the second example, we considered the quadratic eigenpencil (2) with matrices

$$M = \begin{pmatrix} 1.6312 & -0.2473 & -1.0380 & 0.4628 \\ -0.2473 & 0.9275 & -0.0052 & 0.2589 \\ -1.0380 & -0.0052 & 2.1554 & 0.1102 \\ 0.4628 & 0.2589 & 0.1102 & 0.8301 \end{pmatrix}, D = \begin{pmatrix} 1.4794 & -1.1102 & 0 & -0.2222 \\ -1.1102 & 0.3455 & 0.1237 & 0 \\ 0 & 0.1237 & 2.4643 & -0.1004 \\ -0.2222 & 0 & -0.1004 & 1.0838 \end{pmatrix},$$

and

$$K = \begin{pmatrix} 0.5875 & -0.1668 & 0 & 0 \\ -0.1668 & 0.1831 & 0.0456 & 0 \\ 0 & 0.0456 & 1.0749 & 0.3803 \\ 0 & 0 & 0.3803 & 0.5624 \end{pmatrix}.$$

Notice that D and K are sparse and hence the nonlinear programming subproblems have $n_{\rm nz} = 15$ variables. The objective is to find matrices $\tilde{D}$ and $\tilde{K}$ as near as possible to D and K, respectively, and such that the updated eigenpencil (3) has the two desired eigenpairs (λ_1, x_1) and (λ_2, x_2), where $\lambda_1 = -0.1 + 0.3398i$, $x_1 = (0.5 + 0.04i, 0.8, -0.04 + 0.1i, 0.04 - 0.1i)^T$, and (λ_2, x_2) corresponds to the conjugate eigenpair. In addition, we would also like the updated eigenpencil to have all its quadratic eigenvalues with their real part not larger than $\hat{\lambda} = -0.1$.

By solving the first nonlinear programming subproblem at Step 2 (with $\kappa = 0$, i.e., with no cut constraints), we obtain matrices $\tilde{D}_0$ and $\tilde{K}_0$ such that $\|D - \tilde{D}_0\|_F^2 = 0.4284$ and $\|K - \tilde{K}_0\|_F^2 = 0.0557$ (the sum being equal to 0.4841). When computing an eigenvalue τ with largest real part we find that $\tau = -0.0798 \not\leq -0.1 = \hat{\lambda}$. Therefore, we compute an associated normalized eigenvector given by $u_0 = (0.0008, 0.1063, -0.5433, 0.8328)^T$, and, by the test at Step 4.2, set $s_0 = 1$. Then, we solve a new nonlinear programming subproblem at Step 2 (this time with a single cut of type (12) given by u_0 and s_0) and find matrices

$$\tilde{D}_1 = \begin{pmatrix} 1.5936 & -0.9073 & 0 & -0.0807 \\ -0.9073 & 0.7109 & -0.0392 & 0 \\ 0 & -0.0392 & 2.4946 & -0.2534 \\ -0.0807 & 0 & -0.2534 & 1.3588 \end{pmatrix}$$

and

$$\tilde{K}_1 = \begin{pmatrix} 0.5318 & -0.1968 & 0 & 0 \\ -0.1968 & 0.2081 & 0.0312 & 0 \\ 0 & 0.0312 & 0.9915 & 0.4758 \\ 0 & 0 & 0.4758 & 0.4612 \end{pmatrix}.$$

This time, the quadratic eigenvalue τ with largest real part corresponds to $\tau = -0.1 \pm 0.3398i$ and, therefore, since $\Re(\tau) = -0.1 = \hat{\lambda}$, we are done. Matrices $\tilde{D}_1$ and $\tilde{K}_1$ are such that $\|D - \tilde{D}_1\|_F^2 = 0.4454$ and $\|K - \tilde{K}_1\|_F^2 = 0.0414$. Since the sum of both squared distances is equal to 0.4868, it means that we have obtained the desired updated eigenpencil, with no unstable eigenvalues, at the price of a "very small" increase ($\approx 0.5\%$) in the objective function value. Observe that, as required, $\tilde{D}_1$ and $\tilde{K}_1$ are symmetric and preserve the desired sparsity pattern.

4.3 Example 3.

In the third example, we considered the quadratic eigenpencil (2) given by matrices

$$M = \begin{pmatrix} 1.9979 & 0.3890 & -0.3500 & 0.5459 \\ 0.3890 & 1.5993 & 0.2906 & -0.8680 \\ -0.3500 & 0.2906 & 1.1656 & -0.5510 \\ 0.5459 & -0.8680 & -0.5510 & 1.8281 \end{pmatrix}, D = \begin{pmatrix} 0.9727 & 0.7667 & -0.1444 & 0.3118 \\ 0.7667 & 0.0000 & 0.1213 & -0.0389 \\ -0.1444 & 0.1213 & 0.7190 & 0.3321 \\ 0.3118 & -0.0389 & 0.3321 & 1.3145 \end{pmatrix},$$

and

$$K = \begin{pmatrix} 0.4018 & 0.4055 & 0.1019 & 0.3685 \\ 0.4055 & 0.5521 & 0.2048 & 0.0112 \\ 0.1019 & 0.2048 & 0.2443 & 0.0941 \\ 0.3685 & 0.0112 & 0.0941 & 0.8133 \end{pmatrix}.$$

The desired quadratic eigenpair is given by $\lambda_1 = -0.1$ and $x_1 = (0.6, -0.6, 0.4, -0.5)^T$. In addition, we would like the modified eigenpencil (3) to have all its eigenvalues with their real part not larger than $\hat{\lambda} = -0.1$.

Since matrices D and K are dense, the nonlinear programming subproblems at Step 2 have $n_{\text{nz}} = 20$ variables. When solving the first nonlinear programming subproblem with no cuts, we obtained matrices D_0 and K_0 such that $\|D - \tilde{D}_0\|_F^2 = 0.0006$ and $\|K - \tilde{K}_0\|_F^2 = 0.0646$. When computing a quadratic eigenvalue τ with largest real part of the associated modified eigenpencil, we observed that $\tau = 0.626 \not\leq -0.1 = \hat{\lambda}$. Therefore, we computed an associated normalized eigenvector $u_0 = (-0.5199, 0.715, -0.3242, 0.3368)^T$, and, by the test at Step 4.2, set $s_0 = 1$. When solving the second nonlinear programming subproblem, this time with the addition of the cut given by u_0 and s_0, we obtained matrices D_1 and K_1 such that $\|D - \tilde{D}_1\|_F^2 = 0.2703$ and $\|K - \tilde{K}_1\|_F^2 = 0.0655$. When computing a quadratic eigenvalue $\tau = -0.047 \pm 0.4178i$ with largest real part, we observed that $\Re(\tau) = -0.047 \not\leq -0.1 = \hat{\lambda}$. A normalized eigenvector associated with the eigenvalue $-0.047 + 0.4178i$ is given by $u_1 = (-0.1964 - 0.5258i, -0.153 + 0.7568i, -0.1265 - 0.0316i, 0.2661 - 0.0305i)^T$. By the test at Step 4.2, we set $s_1 = -1$. This means that the next nonlinear programming subproblem to be solved had two cut constraints. By solving it, we obtained matrices

$$\tilde{D}_2 = \begin{pmatrix} 1.1721 & 0.5134 & -0.1034 & 0.2673 \\ 0.5134 & 0.4118 & 0.0858 & -0.0337 \\ -0.1034 & 0.0858 & 0.7428 & 0.2943 \\ 0.2673 & -0.0337 & 0.2943 & 1.3818 \end{pmatrix}$$

and

$$\tilde{K}_2 = \begin{pmatrix} 0.5662 & 0.3403 & 0.1600 & 0.3661 \\ 0.3403 & 0.5181 & 0.2128 & -0.0690 \\ 0.1600 & 0.2128 & 0.2487 & 0.1354 \\ 0.3661 & -0.0690 & 0.1354 & 0.7032 \end{pmatrix}$$

such that $\|D - \tilde{D}_2\|_F^2 = 0.3555$ and $\|K - \tilde{K}_2\|_F^2 = 0.072$. When computing an eigenvalue τ with largest real part associated with the modified eigenpencil, we obtained that $\tau = -0.1 \leq \hat{\lambda}$ and, therefore, the algorithm stopped.

It is worth noticing that the final sum of the squared distances is 0.4275, which is relatively larger than the sum of the squared distances obtained after the first iteration of the algorithm, whose value is 0.0652. This is because the matrices $\tilde{D}_0$ and $\tilde{K}_0$ have, as well as the original matrices, a quadratic eigenvalue $\lambda \approx 0.6$ that is relatively far from the desired upper limit $\hat{\lambda} = -0.1$. This justifies the "relatively large modification" of the matrices, reflected in the relatively large final squared distances, when compared to the distances obtained after the first iteration of the algorithm that did not include any cut.

4.4 Example 4.

In the fourth example, we considered the quadratic eigenpencil (2) given by sparse matrices $M, D, K \in \mathbb{R}^{100\times 100}$, such that M is diagonal and positive definite and D and K are both tridiagonal and symmetric. Matrices M, D, and K can be found in [5]. The two desired quadratic eigenvalues are given by $\lambda_1 = -0.3 + 0.4713i$ and its conjugate $\lambda_2 = -0.3 - 0.4713i$. The desired eigenvectors x_1 and x_2 are such that x_1 has its non-zero elements given by

$$\begin{array}{rcr} [x_1]_1 & = & 0.0010 + 0.0001i \\ [x_1]_2 & = & -0.0010 + 0.0001i \\ [x_1]_{16} & = & -0.0020 + 0.0000i \\ [x_1]_{17} & = & -0.0020 + 0.0020i \\ [x_1]_{18} & = & -0.0100 + 0.0050i \\ [x_1]_{19} & = & 0.8000 + 0.0000i \\ [x_1]_{20} & = & -0.0050 - 0.0020i \\ [x_1]_{100} & = & 0.0010 + 0.0010i \end{array}$$

and x_2 is the conjugate of x_1. In addition, we would like the modified eigenpencil (3) to have all its eigenvalues with their real part not larger than $\hat{\lambda} = -0.3$. Matrices D and K are tridiagonal, so we have $I_D = I_K = \{(i,i) \mid 1 \leq i \leq 100\} \cup \{(i,i+1) \mid 1 \leq i \leq 99\}$ and, thus, the nonlinear programming subproblems at Step 2 have $n_{\text{nz}} = 398 \approx 4n$ variables.

Algorithm 3.1 required four iterations to solve this problem. The computed eigenvalue with largest real part at each iteration, as well as the squared distances of the obtained matrices, are reported in Table 1. Intermediate matrices ($\tilde{D}_0$, $\tilde{D}_1$, $\tilde{D}_2$, $\tilde{K}_0$, $\tilde{K}_1$, and $\tilde{K}_2$) and final matrices $\tilde{D}_3$ and $\tilde{K}_3$ are reported in [5]. In the considered computational environment, the elapsed CPU time required to solve this problem was 6.96 seconds.

κ	$\|D - \tilde{D}_\kappa\|_F^2$	$\|K - \tilde{K}_\kappa\|_F^2$	τ
0	0.5305	0.1308	$-0.2275 \pm 0.4486i$
1	0.5876	0.1367	$-0.2910 \pm 0.4472i$
2	0.5886	0.1369	$-0.2989 \pm 0.4406i$
3	0.5891	0.1370	$-0.3000 \pm 0.4713i$

Table 1: Details of the progress of Algorithm 3.1 when applied to Example 4.

5 Final remarks

We introduced a new optimization approach for solving the QFEMUP problem, that combines the use of standard nonlinear programming solvers with the dynamical inclusion of additional constraints to avoid that spurious modes appear in the frequency range of interest. These additional constraints or cuts are based on an extension of the Rayleigh quotient for quadratic eigenvalue problems. For this combination of ideas, we have presented an iterative algorithm for which we have established finite termination. A key feature, observed on some preliminary numerical experiments, is that our new machinery finds an optimal and feasible solution computing only a few eigenvalues and eigenvectors of the associated quadratic matrix pencil. For our experiments we have combined the packages Algencan for solving the constrained optimization problems, and the package ARPACK for the eigenvalue-eigenvector calculations. Both packages are easy to obtain and easy to use, however, it is worth mentioning that any other packages can be used as inner-solvers in our iterative scheme.

In our approach, the structure of the feasible region played a key role concerning practical and also theoretical issues. However, the objective function of the sequence of nonlinear programming problems was not referred in the analysis of the algorithm. Hence, our iterative combined scheme can be extended to solve some other related model updating and eigenvalue assignment problems; see e.g., [6, 12, 20, 26, 34]. Moreover, the symmetry of the involved matrices M, D, and K, does not play any special role in our optimization machinery, besides halving the required storage, and so our approach can also be adapted to solve some other problems for which these structural constraints are not required.

Acknowledgements. We thank an anonymous referee for helpful suggestions.

References

[1] M. O. Abdalla, K. M. Grigoriadis, and D. C. Zimmerman, Enhanced structural damage detection using alternating projection methods, *AIAA Journal* 36, pp. 1305–1311, 1998.

[2] R. Andreani, E. G. Birgin, J. M. Martínez, and M. L. Schuverdt, On Augmented Lagrangian methods with general lower-level constraints, *SIAM Journal on Optimization* 18, pp. 1286–1309, 2007.

[3] R. Andreani, E. G. Birgin, J. M. Martínez, and M. L. Schuverdt, Augmented Lagrangian methods under the Constant Positive Linear Dependence constraint qualification, *Mathematical Programming* 111, pp. 5–32, 2008.

[4] M. Andretta, E. G. Birgin and J. M. Martínez, Practical active-set Euclidean trust-region method with spectral projected gradients for bound-constrained minimization, *Optimization* 54, pp. 305–325, 2005.

[5] M. Andretta, E. G. Birgin, and M. Raydan, An inner-outer nonlinear programming approach for constrained quadratic matrix model updating, Technical Report MCDO130814

(see `http://www.ime.usp.br/~egbirgin/`), Department of Computer Science, Institute of Mathematics and Statistics, University of São Paulo, Brazil.

[6] Z.-J. Bai, B. N. Datta, and J. Wang, Robust and minimum norm partial quadratic eigenvalue assignment in vibrating systems: A new optimization approach, *Mechanical Systems and Signal Processing* 24, pp. 766–783, 2010.

[7] M. Baruch, Optimization procedure to correct stiffness and flexibility matrices using vibration data, *AIAA Journal* 16, pp. 1208–1210, 1978.

[8] C. A. Beattie and S. W. Smith, Optimal matrix approximants in structural identification, *Journal of Optimization Theory and Applications* 74, pp. 23–56, 1992.

[9] E. G. Birgin and J. M. Martínez, A box-constrained optimization algorithm with negative curvature directions and spectral projected gradients, *Computing [Suppl]* 15, pp. 49–60, 2001.

[10] E. G. Birgin and J. M. Martínez, Large-scale active-set box-constrained optimization method with spectral projected gradients, *Computational Optimization and Applications* 23, pp. 101–125, 2002.

[11] E. G. Birgin and J. M. Martínez, *Practical Augmented Lagrangian Methods for Constrained Optimization*, Society for Industrial and Applied Mathematics, Philadelphia, 2014.

[12] S. Brahma and B. N. Datta, An optimization approach for minimum norm and robust partial quadratic eigenvalue assignment problems for vibrating structures, *Journal of Sound and Vibration* 324, pp. 471–489, 2009.

[13] J. B. Carvalho, B. N. Datta, A. Gupta, and M. Lagadapati, A direct method for model updating with incomplete measured data and without spurious modes, *Mechanical Systems and Signal Processing* 21, pp. 2715–2731, 2007.

[14] J. B. Carvalho, B. N. Datta, W.-W. Lin, and C.-S. Wang, Symmetry preserving eigenvalue embedding in finite element model updating of vibrating structures, *Journal of Sound and Vibration* 290, pp. 839–864, 2006.

[15] M. T. Chu, B. N. Datta, W.-W. Lin, and S.-F. Xu, Spillover phenomenon in quadratic model updating, *AIAA Journal* 46, pp. 420–428, 2008.

[16] M. T. Chu, W.-W. Lin, and S.-F. Xu, Updating quadratic models with no spill-over effect on unmeasured spectral data, *Inverse Problems* 23, pp. 243–256, 2007.

[17] B. N. Datta, Finite element model updating, eigenstructure assignment and eigenvalue embedding techniques for vibrating systems, *Mechanical Systems and Signal Processing* 16, pp. 83–96, 2002.

[18] B. N. Datta, *Numerical Linear Algebra and Applications*, second edition, SIAM, Philadelphia, 2010.

[19] B. N. Datta, *Numerical methods for linear control systems*, Elsevier Academic Press, San Diego, CA, 2004.

[20] B. N. Datta, S. Deng, V. O. Sokolov, and D. R. Sarkissian, An optimization technique for damped model updating with measured data satisfying quadratic orthogonality constraint, *Mechanical Systems and Signal Processing* 23, pp. 1759–1772, 2009.

[21] B. N. Datta, S. Elhay, Y. M. Ram, and D. R. Sarkissian, Partial eigenstructure assignment for the quadratic pencil, *Journal of Sound and Vibration* 1, pp. 101–110, 2000.

[22] B. N. Datta and D. R. Sarkissian, Multi-input partial eigenvalue assignment for the symmetric quadratic pencil, *Proceedings of the 1999 American Control Conference* 4, pp. 2244–2247, 1999.

[23] B. N. Datta and D. R. Sarkissian, Theory and computations of some inverse eigenvalue problems for the quadratic pencil, *Contemporary Mathematics, Volume Structured Matrices in Operator Theory, Control and Signal an Image Processing* 280, American Mathematical Society, pp. 221–240, 2001.

[24] B. N. Datta and D. R. Sarkissian, A computational method for feedback control in distributed parameter systems, *Proceedings of the 8th IEEE International Conference on Methods and Models in Robotics*, pp. 139–144, 2002.

[25] B. N. Datta and V. O. Sokolov, Quadratic inverse eigenvalue problems, active vibration control and model updating, *Applied and Computational Mathematics* 8, pp. 170–191, 2009.

[26] B. N. Datta and V. O. Sokolov, A solution of the affine quadratic inverse eigenvalue problem, *Linear Algebra and its Applications* 434, pp. 1745–1760, 2011.

[27] M. A. Diniz-Ehrhardt, M. A. Gomes-Ruggiero, J. M. Martínez, and S. A. Santos, Augmented Lagrangian algorithms based on the spectral projected gradient for solving nonlinear programming problems, *Journal of Optimization Theory and Applications* 123, pp. 497–517, 2004.

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

[29] J. C. Geromel, On the determination of a diagonal solution of the Lyapunov equation, *IEEE Transactions on Automatic Control* AC-30, pp. 404–406, 1985.

[30] M. E. Hochstenbach and H. A. van Der Vorst, Alternatives to the Rayleigh quotient for the quadratic eigenvalue problem, *SIAM Journal on Scientific Computing* 25, pp. 591–603, 2003.

[31] H. Hu, Positive definite constrained least-squares estimation of matrices, *Linear Algebra and its Applications* 229, pp. 167–174, 1995.

[32] D. J. Inman, *Vibrations: with Control, Measurement, and Stability*, Prentice Hall, Englewood Cliffs, NJ, 1989.

[33] D. J. Inman and A. Kress, Eigenstructure assignment using inverse eigenvalue methods, *Journal of Guidance, Control, and Dynamics* 18, pp. 625–627, 1995.

[34] Y.-C. Kuo and B. N. Datta, Quadratic model updating with no spill-over and incomplete measured data: Existence and computation of solution, *Linear Algebra and its Applications* 436, pp. 2480–2493, 2012.

[35] R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*, Society for Industrial and Applied Mathematics, Philadelphia, 1998.

[36] K. Miettinen, *Nonlinear Multiobjective Optimization*, Kluwer academic Publishers, Boston, MA, 1999.

[37] J. Moreno, B. N. Datta, and M. Raydan, A symmetry preserving alternating projection method for matrix model updating, *Mechanical Systems and Signal Processing* 23, pp. 1784–1791, 2009.

[38] N. K. Nichols and J. Kautsky, Robust eigenstructure assignment in quadratic matrix polynomials: nonsingular case, *SIAM Journal on Matrix Annalysis and Applications* 23, pp. 77–102, 2001.

[39] J. Qian and S. Xu, Robust partial eigenvalue assignment problem for the second-order system, *Journal of Sound and Vibration* 282, pp. 937–948, 2005.

[40] F. Tisseur and K. Meerbergen, The quadratic eigenvalue problem, *SIAM Review* 43, pp. 235–286, 2001.