Received: 8 April 2021 | Revised: 20 November 2022 | Accepted: 25 November 2022

DOI: 10.1002/zamm.202100127

EDITOR'S CHOICE

Fast optimization of viscosities for frequency-weighted damping of second-order systems

Nevena Jakovčević Stor[1] | **Tim Mitchell**[2,3] | **Zoran Tomljanović**[4] | **Matea Ugrica**[2,4]

[1]Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, University of Split, Split, Croatia

[2]Computational Methods in Systems and Control Theory, Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg, Germany

[3]Department of Computer Science, Queens College, CUNY, Flushing, New York, USA

[4]Department of Mathematics, University Josip Juraj Strossmayer, Osijek, Croatia

Correspondence
Matea Ugrica, Computational Methods in Systems and Control Theory, Max Planck Institute for Dynamics of Complex Technical Systems, Sandtorstr. 1, 39106 Magdeburg, Germany.
Email: ugrica@mpi-magdeburg.mpg.de

Present address
Tim Mitchell, Department of Computer Science, Queens College/CUNY, 65-30 Kissena Boulevard, Flushing, NY 11367, USA.
Matea Ugrica, Computational Methods in Systems and Control Theory, Max Planck Institute for Dynamics of Complex Technical Systems, Sandtorstr. 1, 39106 Magdeburg, Germany.

Funding information
Hrvatska Zaklada za Znanost, Grant/Award Numbers: IP-2019-04-6774, IP-2020-02-2240

We consider frequency-weighted damping optimization for vibrating systems described by a second-order differential equation. The goal is to determine viscosity values such that eigenvalues are kept away from certain undesirable areas on the imaginary axis. To this end, we present two complementary techniques. First, we propose new frameworks using nonsmooth constrained optimization problems, whose solutions both damp undesirable frequency bands and maintain the stability of the system. These frameworks also allow us to weight, which frequency bands are the most important to damp. Second, we also propose a fast new eigensolver for the structured quadratic eigenvalue problems (QEPs) that appear in such vibrating systems. In order to be efficient, our new eigensolver exploits special properties of diagonal-plus-rank-one (DPR1) complex symmetric matrices, which we leverage by showing how each QEP can be transformed into a short sequence of such linear eigenvalue problems. The result is an eigensolver that is substantially faster than standard techniques. By combining this new solver with our new optimization frameworks, we obtain our overall algorithm for fast computation of optimal viscosities. The efficiency and performance of our new approach are verified and illustrated on several numerical examples.

1 | INTRODUCTION

Consider a vibrational mechanical system described by the second-order differential equation

$$M\ddot{q}(t) + C(\upsilon)\dot{q}(t) + Kq(t) = 0, \tag{1.1}$$

where $M, C(\upsilon), K \in \mathbb{R}^{n\times n}$ are all symmetric positive definite matrices, respectively, representing mass, damping, and stiffness, and the damping matrix $C(\upsilon)$ depends on r nonnegative viscosity parameters, that is, $\upsilon \in \mathbb{R}^r_+$, where $\mathbb{R}^r_+$ is the set of r-dimensional vectors with real nonnegative entries. We assume that the number of damping parameters is small, that is, $r \ll n$, as is typical in practice, and that $C(\upsilon)$ has the following form:

$$C(\upsilon) = C_{\text{int}} + G\ \text{diag}(\upsilon_1, \dots, \upsilon_r)G^{\mathsf{T}} = C_{\text{int}} + \sum_{j=1}^{r} \upsilon_j g_j g_j^{\mathsf{T}}, \tag{1.2}$$

where C_{int} represents internal damping, $G \in \mathbb{R}^{n\times r}$ describes the geometry of damping positions, and g_j denotes the jth column of G. Internal damping can be modeled in various ways, for example, Rayleigh (or classical) damping, where $C_{\text{int}} = \alpha M + \beta K$ and $\alpha, \beta \geq 0$. In this paper, we focus on another convention that is widely used, namely that the internal damping is a small multiple of the critical damping, that is,

$$C_{\text{int}} = \alpha M^{\frac{1}{2}} \sqrt{M^{-\frac{1}{2}} K M^{-\frac{1}{2}}}\, M^{\frac{1}{2}}, \tag{1.3}$$

where $\alpha > 0$. In this case, C_{int} is symmetric positive definite. For more details on critical damping, see Refs. [1–3].

The second-order differential equation (1.1) of course is associated with the quadratic eigenvalue problem (QEP)

$$(\lambda(\upsilon)^2 M + \lambda(\upsilon)C(\upsilon) + K)x(\upsilon) = 0. \tag{1.4}$$

For a given vector υ specifying the viscosity parameters, let $\Lambda(\upsilon)$ denote the spectrum of Equation (1.4). Each eigenvalue $\lambda(\upsilon)$ corresponds to a natural frequency of the system (1.1), that is, a frequency on which the system prefers to vibrate. Vibrations can be increased if the system is excited by an external force whose frequencies are close to the natural frequencies. All frequencies that can significantly excite the system are called undesirable frequencies.

One approach to minimizing the impact of external forces is damping optimization, which has been widely studied in the last few decades. In the most general context, given mass and stiffness matrices, the problem is to determine a damping matrix such that unwanted vibrations decay as fast as possible. This requires specifying an objective function to be optimized, and the choice of objective function strongly depends on the application and desired outcome. An overview of different damping optimization criteria can be found in Veselić [1]. For the nonhomogeneous case, where the system is additionally excited, damping optimization has also been studied in Refs. [1, 4]. In Kuzmanović et al. [4], the authors consider energy over arbitrary time, while Veselić [1] considers the case where the excitation function is periodic. For multiple input, multiple output systems, one can also optimize damping in second-order systems by minimizing standard systems norms, such as the $\mathcal{H}_2$ or $\mathcal{H}_\infty$ norms; see Refs. [5–9].

In our setting, Equation (1.1) corresponds to the homogeneous case for which one can consider optimizing the total average energy in various ways; see Refs. [1, 10–12]. One can also use eigenvalue-based criteria to damp resonant frequencies, that is, by directly altering the spectrum of Equation (1.4), as has been considered in Gräbner et al. [13] where the spectral abscissa criterion is minimized. The spectral abscissa of Equation (1.4) is defined

$$\alpha_{\text{MCK}}(\upsilon) = \max_{\lambda(\upsilon)\in\Lambda(\upsilon)} \text{Re}\,\lambda(\upsilon). \tag{1.5}$$

Given some $\upsilon \in \mathbb{R}^r$, the system (1.1) is asymptotically stable if and only if all the eigenvalues of the corresponding eigenvalue problem (1.4) are in the open left half-plane, that is, $\alpha_{\text{MCK}}(\upsilon) < 0$. Note that under our assumption that $M, C(\upsilon), K$ are all symmetric positive definite matrices, and where $\upsilon \in \mathbb{R}^r_+$, the system (1.1) is asymptotically stable; for more details, see Tisseur and Meerbergen [14].

We consider the frequency isolation problem where viscosities are optimized in order to keep eigenvalues away from the certain undesirable areas on the imaginary axis, that is, away from undesirable resonant frequency bands that are known a priori. This variation of the frequency isolation problem has been studied in several works. In Joseph [15], a Newton-type method for structures vibrating at low frequencies was proposed, while a less costly inverse eigenvalue method was presented in Egaña et al. [16], where a target spectrum, which avoids an undesirable resonance band, is fixed in advance. Meanwhile, Moro and Egaña [17] considered the frequency isolation problem for undamped vibrational systems where there is no $C(\upsilon)$ in Equation (1.4), or equivalently, $C(\upsilon)$ is always zero. In damping optimization where $C(\upsilon)$ is present, avoiding undesirable frequency bands can be achieved either by choosing damping positions (by optimizing matrix G) or by viscosity values (by optimizing $\upsilon \in \mathbb{R}^r_+$) or doing both simultaneously. Computing the optimal damping positions is a very challenging problem and there is no efficient algorithm for it, though some heuristics can be found in, for example, Kanno et al. [18]. One approach to determining optimal damping positions is "direct" brute force, where all possible damping configurations are considered and viscosities are optimized for each configuration. In any case, while optimization of damping positions is a challenging and a very important question in and of itself, in this paper, we focus on accelerating this overall process via proposing faster methods for viscosity optimization for fixed damping positions. Therefore, in our algorithms here, we consider that the matrix G specifying the damping positions is fixed, but we have in mind that, in practice, viscosity optimization would be applied over many different configurations of damping positions.

In this paper, we aim to accelerate such damping-based approaches for frequency isolation via new fast techniques for the important subproblem of determining optimal damping viscosities for a given configuration of damping positions. More specifically, given a general system (1.1), where $C(\upsilon)$ has the form given in Equation (1.2), the internal damping matrix C_{int} is given by Equation (1.3), and the matrix G specifying the damping positions is fixed, we consider the problem of optimizing the viscosity parameters $\upsilon \in \mathbb{R}^r_+$ so that the eigenvalues of Equation (1.4) are kept away from undesirable resonant bands as much as possible. Our contribution here consists of two complementary new techniques. First, we propose two related nonsmooth but continuous constrained optimization problems as new models for the frequency isolation problem and show how solutions can be computed via gradient information and recent advances in nonsmooth constrained optimization. When our new problems are solved, their solutions provide viscosity parameters, which maintain stability of the system and damp user-defined undesirable frequency bands. In addition to specifying the number of frequency bands and their respective widths, the undesirable bands can also be weighted in order to prioritize which are most critical to damp. Second, as the cost of our optimization process is actually dominated by solving a sequence of related QEPs, where $C(\upsilon)$ is changing as the viscosity parameters are optimized, we also propose a fast algorithm to solve this sequence of QEPs. Our method, which is many times faster than using standard eigensolvers for QEPs and can be considered an extension of Jakovčević et al. [19] for computing eigenvalues of diagonal-plus-rank-one (DPR1) complex symmetric (DPR1Csym) matrices, works by exploiting the fact that changing the viscosity parameters is equivalent to making a low-rank update to $C(\upsilon)$. Since such structure is not inherent to our problem, we expect that our technique for solving such sequences of QEPs could be quite beneficial in other applications as well.

The paper is organized as follows. In Section 2, we motivate and establish our two new models for the frequency isolation problem, explain their properties, and discuss how to compute solutions of them. Then, in Section 3, we present our new approach for efficiently solving the corresponding sequence of QEPs that arises during optimization (using either of our new models discussed in the previous section). We show how both eigenvalues and eigenvectors of the QEPs can be computed, as both are needed in our optimization-based approach. In Section 4, we present our full algorithm for damping optimization by combining our aforementioned components from Sections 2 and 3. Finally, we validate our new techniques and compare our two models for frequency-weighted damping in Section 5.

2 | NEW FRAMEWORKS FOR FREQUENCY-WEIGHTED DAMPING

Consider how the eigenvalues $\Lambda(\upsilon)$ of Equation (1.4) evolve as the viscosities parameters υ are changed, and suppose $\omega \in \mathbb{R}$ is an undesirable frequency, that is, we wish to keep the spectrum of Equation (1.4) away from $\mathbf{i}\omega$ on the imaginary axis. Since eigenvalues with imaginary parts close to ω can also be undesirable, we thus consider the undesirable frequency band $[\omega - b, \omega + b]$ about ω for some given $b > 0$. In order to minimize the impact of eigenvalues of Equation (1.4) in this

frequency band, it is tempting to consider solving the optimization problem:

$$\begin{aligned} \min_{v\in\mathbb{R}^r} \quad & \max\{\operatorname{Re}\lambda(v) : \lambda(v)\in\Lambda(v) \text{ and } \operatorname{Im}\lambda\in[\omega-b,\omega+b]\} \\ \text{s.t.} \quad & \alpha_{\mathrm{MCK}}(v)\le \mathtt{tol}_{\mathrm{sa}} \quad \text{for some } \mathtt{tol}_{\mathrm{sa}}<0, \\ & v_j\ge 0 \quad \text{for } j=1,\dots,r, \end{aligned} \tag{2.1}$$

which would act to push all the eigenvalues of $\Lambda(v)$ with imaginary parts in $[\omega-b,\omega+b]$ as far to the left as possible while still maintaining asymptotic stability of the system and physically realistic, that is, nonnegative, viscosities. Alternatively, one might consider swapping the objective function and the stability constraint in Equation (2.1), that is, minimize the spectral abscissa as much as possible subject to the constraint that the eigenvalues in the frequency band, that is, those which have imaginary parts in $[\omega-b,\omega+b]$, are all kept at least some fixed distance to the left of the imaginary axis (and again enforcing nonnegative viscosities). However, these two related optimization problems are rather difficult to solve as the function being minimized in Equation (2.1) is actually discontinuous. In general, this function has jump discontinuities whenever a rightmost eigenvalue that attains the maximum leaves the horizontal strip in the complex plane defined by ω and b, or vice versa, when a new eigenvalue enters this region to become a rightmost eigenvalue in this strip, and these discontinuities are typically not uncommon.

To overcome this problem, in this section, we propose two alternatives to Equation (2.1) where continuity is maintained and so our new optimization problems for frequency isolation are much more practical to solve. This allows us to use continuous optimization techniques to compute viscosity values such that eigenvalues are kept away from an undesirable frequency band defined by ω and b. In fact, as we will soon explain, our distance function can be used for different undesirable frequency bands simultaneously.

2.1 | Approach 1

Let the tuple $E=(a,b,c)$ denote the axis-aligned ellipse

$$\frac{(x-\operatorname{Re}c)^2}{a^2}+\frac{(y-\operatorname{Im}c)^2}{b^2}=1, \tag{2.2}$$

where $a,b>0$, respectively, denote the semi-major and -minor axes and $c\in\mathbb{C}$ is the center of the ellipse. Identifying $\mathbb{R}^2$ with $\mathbb{C}$, consider the following algebraic distance $d:\mathbb{C}\mapsto[0,\infty)$ of a point $z\in\mathbb{C}$ to this ellipse, that is,

$$d(z;E) := \frac{(\operatorname{Re}(z-c))^2}{a^2}+\frac{(\operatorname{Im}(z-c))^2}{b^2}. \tag{2.3}$$

Thus, $d(z;E)>1$ when z is outside of the ellipse, $d(z;E)\in[0,1)$ when z is inside the ellipse, and $d(z;E)=1$ when z is on the ellipse, that is, $z=x+\mathbf{i}y$ satisfies Equation (2.2).

Now suppose that $\omega\ge 0$ and $b>0$ specify an undesirable frequency band $[\omega-b,\omega+b]$. Given some $a>0$, a notion of distance between a point in the complex plane and the interval $\mathbf{i}[\omega-b,\omega+b]$ on the imaginary axis is given via $d(z;E)$ for $E=(a,b,\mathbf{i}\omega)$, that is, the ellipse (2.2) centered at $\mathbf{i}\omega$ on the imaginary axis. If z is such that $\operatorname{Im}z\in(\omega-b,\omega+b)$, then the larger we make the value of a, the further z must be to the left or right of the minor axis of the ellipse given by E in order for $d(z;E)>1$ to hold. Thus, as a continuous measure of the distance of the spectrum of Equation (1.4) to the undesirable frequency $[\omega-b,\omega+b]$, we define

$$d_{\Lambda,E}(v) := \min\{d(\lambda(v);E) : \lambda(v)\in\Lambda(v)\}, \tag{2.4}$$

where $E=(a,b,\mathbf{i}\omega)$. Function $d_{\Lambda,E}(v)>1$ when all the eigenvalues of $\Lambda(v)$ are outside the given ellipse, $d_{\Lambda,E}(v)\in[0,1)$ when one or more eigenvalues are inside this ellipse, and $d_{\Lambda,E}(v)=1$ when at least one eigenvalue is on this ellipse and none are inside. The specific value of the semi-major axis a determines the importance of the undesirable frequency band by dictating how far away eigenvalues should be from the interval $\mathbf{i}[\omega-b,\omega+b]$, where eigenvalues with imaginary parts closer to ω are weighted more, that is, must be further away. When multiple undesirable frequency bands are specified

by frequencies $\{\omega_1, \dots, \omega_k\}$ and associated (half) bandwidths $\{b_1, \dots, b_k\}$, their relative importance can be determined by providing different semi-major axis values $\{a_1, \dots, a_k\}$, with $\omega_j \geq 0$ and $a_j, b_j > 0$ for all $j = 1, \dots, k$. Thus, we generalize Equation (2.4) to measuring the distance of the spectrum $\Lambda(v)$ to the k undesirable frequency bands by defining

$$d_{\Lambda,\mathcal{E}}(v) := \min\{d_{\Lambda,E_j}(v) : E_j \in \mathcal{E}\}, \tag{2.5}$$

where $E_j := (a_j, b_j, \mathbf{i}\omega_j)$ is the tuple defining the jth axis-aligned ellipse for the jth undesirable frequency band $[\omega_j - b_j, \omega_j + b_j]$ with relative importance $a_j > 0$ and $\mathcal{E} := \{E_1, \dots, E_k\}$ is the set of k corresponding ellipses.

Using Equation (2.5), we now present our first new model for the frequency isolation problem:

$$\textbf{Model 1}: \quad \begin{aligned} \min_{v \in \mathbb{R}^r} \quad & \alpha_{\mathrm{MCK}}(v) \\ \text{s.t.} \quad & d_{\Lambda,\mathcal{E}}(v) \geq 1, \\ & \alpha_{\mathrm{MCK}}(v) \leq \mathtt{tol}_{\mathrm{sa}} \text{ for some } \mathtt{tol}_{\mathrm{sa}} < 0, \\ & v_j \geq 0 \text{ for } j = 1, \dots, r, \end{aligned} \tag{2.6}$$

that is, minimize the spectral abscissa as much as possible subject to the respective constraints that all the eigenvalues of Equation (1.4) are outside of the ellipses defined by $\mathcal{E}$, the system is asymptotically stable, and the viscosities are nonnegative. Although the spectral abscissa is being minimized in Equation (2.6), note that the additional constraint that the spectral abscissa be negative is necessary. There are multiple reasons for this. First, not all optimization solvers iterate only over the feasible set, and so negative viscosities may be encountered during optimization, which in turn may make $\alpha_{\mathrm{MCK}}(v)$ positive. Second, satisfying $d_{\Lambda,\mathcal{E}}(v) \geq 1$ is not equivalent to satisfying stability, as $d_{\Lambda,\mathcal{E}}(v) \geq 1$ can hold even if all the eigenvalues were to be in the right half-plane. Third, $\alpha_{\mathrm{MCK}}(v)$ may have stationary points where $\alpha_{\mathrm{MCK}}(v) \geq 0$ holds, and so a feasible minimizer of Equation (2.6) without this stability constraint would not necessarily result in an asymptotically stable system.

While $\alpha_{\mathrm{MCK}}(v)$ and $d_{\Lambda,\mathcal{E}}(v)$ in Equation (2.6) are nonsmooth functions, they are at least continuous (unlike the objective function in Equation (2.1)). As there has been significant progress recently in developing effective solvers for nonsmooth constrained optimization, for example, Refs. [20, 21], where the functions are continuous but their nonsmoothness is restricted to a set of measure zero, as is typical, it is reasonable to apply such techniques to compute minimizers of Equation (2.6). We describe the details of how this is done later on and for now make some additional general comments about Equation (2.6). Since $\alpha_{\mathrm{MCK}}(v)$ and $d_{\Lambda,\mathcal{E}}(v)$ will typically be nonconvex and Equation (2.6) may have infeasible stationary points, we cannot necessarily expect to find a globally optimal solution to Equation (2.6), and solvers may also sometimes converge to infeasible points. However, in applications, locally optimal solutions are often sufficient and provide meaningful improvements in performance over nonoptimized configurations. Moreover, both of these problems can typically be mitigated merely by computing multiple solutions to Equation (2.6) via initializing a solver from many different starting points and taking the best of the resulting solutions. Note that the choice of $\mathcal{E}$ depends on the application and is fixed before optimization commences. However, if the a_j values are chosen too aggressively (too large), there is no guarantee that Equation (2.6) will have any feasible solutions. Thus, we now propose a second new model as an alternative and which avoids this issue.

2.2 | Approach 2

Again consider a single axis-aligned ellipse (2.2) defined by tuple $E = (a, b, c)$, where $a, b > 0$ and $c = \eta + \mathbf{i}\omega$ with $\eta, \omega \geq 0$ (so c is not necessarily on the imaginary axis), and suppose that b and c are fixed but a may be varied. Then, given a point $z \in \mathbb{C}$, consider the largest we can make this ellipse, by changing the length of its major axis, such that z is not inside the ellipse. For $z = x + \mathbf{i}y$ and $c = \eta + \mathbf{i}\omega$, solving Equation (2.2) for yields that this largest possible value for the semi-major axis is

$$a(z; E) := \begin{cases} \frac{b|\mathrm{Re}\, z - \eta|}{\sqrt{b^2 - (\mathrm{Im}\, z - \omega)^2}}, & \text{if } \mathrm{Im}\, z \in (\omega - b, \omega + b), \\ \infty & \text{otherwise,} \end{cases} \tag{2.7}$$

where the largest possible semi-major axis value is infinite when the point z is not directly to the left or right of the ellipse, that is, $\operatorname{Im} z \notin (\omega - b, \omega + b)$. Note that this convention is consistent even when $z = \eta + \mathbf{i}(\omega \pm b)$, that is, one of the endpoints of the minor axis, since in this case, z can never be inside the ellipse no matter how large the major axis is. While $a(z; E)$ is determined only by b and c from the tuple E, we continue to use the tuple $E = (a, b, c)$ for notational and conceptual consistency with Section 2.1, but when the value of a is not fixed, we will instead write $E = (\sim, b, c)$.

As a function of z, $a(z; E)$ is real-valued and always nonnegative. Note that $a(z; E)$ is continuous wherever $\operatorname{Re} z \neq \eta$, since then the ratio in Equation (2.7) continuously goes to infinity as $\operatorname{Im} z \in (\omega - b, \omega + b)$ approaches $\omega \pm b$. When $\operatorname{Re} z = \eta$ holds, $a(z; E)$ only has two discontinuities, as in this case, the numerator in $a(z; E)$ is zero, and so $a(z; E)$ has a jump between zero and infinity at $z = \eta + \mathbf{i}(\omega \pm b)$. Relative to Equation (2.1), where the discontinuities can be common and negatively impact solvers, the only two discontinuities of $a(z; E)$ are relatively benign as they typically will not be encountered. Moreover, by a modification, which we will explain momentarily, we can additionally remove these two benign discontinuities from the optimization problem.

Now considering the spectrum $\Lambda(v)$ of Equation (1.4), we can use $a(z; E)$ to determine how the largest value of the semi-major axis of the ellipse $E = (\sim, b, \eta + \mathbf{i}\omega)$, such that none of the eigenvalues are inside it, varies with respect to the viscosities v changing:

$$a_{\Lambda,E}(v) := \min\{a(\lambda(v); E) : \lambda(v) \in \Lambda(v)\}. \tag{2.8}$$

Clearly, function $a_{\Lambda,E}(v)$ inherits the properties of $a(v; E)$ discussed above, but since $\eta \geq 0$, note that $a_{\Lambda,E}(v)$ is continuous at $\hat{v}$ if the system is asymptotically stable for $\hat{v}$. Furthermore, $a_{\Lambda,E}(v)$ is smooth at point $\hat{v}$ if there is only a single eigenvalue (excluding conjugacy) on the ellipse given by $E = (a_{\Lambda,E}(\hat{v}), b, \eta + \mathbf{i}\omega)$ and this eigenvalue is simple. Thus, to damp the frequency band $[\omega - b, \omega + b]$, we could consider solving,

$$\begin{aligned} \max_{v \in \mathbb{R}^r} \quad & a_{\Lambda,E}(v) \\ \text{s.t.} \quad & \alpha_{\mathrm{MCK}}(v) \leq \mathtt{tol}_{\mathrm{sa}} \text{ for some } \mathtt{tol}_{\mathrm{sa}} < 0, \\ & v_j \geq 0 \text{ for } j = 1, \ldots, r. \end{aligned} \tag{2.9}$$

where maximizing $a_{\Lambda,E}(v)$ acts to push all eigenvalues directly to the left of the interval $\mathbf{i}[\omega - b, \omega + b]$ as far to the left as possible. While Equation (2.9) is still discontinuous, encountering the two discontinuities of $a_{\Lambda,E}(v)$ during optimization is unlikely; not only do the discontinuities occur off of the feasible set, that is, when the system is not stable, they require that an eigenvalue passes through $\eta + \mathbf{i}(\omega \pm b)$ *exactly* in order to occur. On the other hand, many optimization solvers do explore the infeasible set during optimization, and even though function $a_{\Lambda,E}(v)$ is otherwise continuous, it can nevertheless have arbitrarily high growth when there exists an eigenvalue $\lambda(v)$ with $\operatorname{Re}\lambda(v) \approx \eta$ and $\operatorname{Im}\lambda(v) \in (\omega - b, \omega + b)$ approaches the endpoints of this interval. As such, before extending Equation (2.9) to the case of multiple ellipses, that is, multiple frequency bands to damp, we first propose modifying Equation (2.9) via a barrier function.

The core idea of introducing a barrier function is to alter Equation (2.9) such that viscosities which cause $\alpha_{\mathrm{MCK}}(v)$ to get close to η will be increasingly penalized, to the point that optimization will never allow a configuration $\hat{v}$ to be accepted as an iterate where $\alpha_{\mathrm{MCK}}(\hat{v}) \geq \eta$ holds. We do this by modifying the objective function such that it goes to negative infinity as $\alpha_{\mathrm{MCK}}(v)$ goes from $\mathtt{tol}_{\mathrm{sa}}$ to η. Since optimization can never accept points where the objective function is infinite, this barrier guarantees that points where $a_{\Lambda,E}(v)$ is discontinuous are never encountered. Furthermore, accepting points where $a_{\Lambda,E}(v)$ is nearly discontinuous will also be heavily discouraged, as the barrier-modified objective function that we are trying to maximize quickly goes to negative infinity as $\alpha_{\mathrm{MCK}}(v)$ increases beyond $\mathtt{tol}_{\mathrm{sa}}$. However, such a barrier function should not introduce any new discontinuities or nonsmooth points of its own, nor should it alter the objective function where $\alpha_{\mathrm{MCK}}(v) \leq \mathrm{tol}_{\mathrm{sa}}$ holds, as all of these things could make optimization more difficult. We construct our barrier function out of a cubic polynomial and log-based function that are specifically crafted to meet these goals.

Given real scalars $y_1 < y_2$ and a continuous function $f : \mathbb{R}^r \mapsto \mathbb{R}$, we define the following generic barrier function:

$$\beta(f(x); y_1, y_2) := \begin{cases} 0 & \text{if } f(x) \le y_1, \\ \tau_1(f(x)-y_1)^3 + \tau_2(f(x)-y_1)^2 & \text{if } f(x) \in (y_1, y], \\ -\log\left(\frac{y_2 - f(x)}{y_2 - y}\right) + h & \text{if } f(x) \in (y, y_2), \\ \infty & \text{otherwise,} \end{cases} \tag{2.10}$$

where $y \in (y_1, y_2)$, $h > 0$, and

$$\tau_1 := \frac{(2h+1)y - y_1 - 2hy_2}{(y_2 - y)(y - y_1)^3} \quad \text{and} \quad \tau_2 := \frac{y_1 + 3hy_2 - (3h+1)y}{(y_2 - y)(y - y_1)^2}. \tag{2.11}$$

Thus, as $f(x)$ goes from y_1 to y_2, our barrier function $\beta(f(x); y_1, y_2)$ goes from zero to infinity. The constants $\tau_1, \tau_2 \in \mathbb{R}$ are specifically chosen so that the value of $\beta(f(x); y_1, y_2)$ always varies continuously and $\nabla\beta(f(x); y_1, y_2)$ is continuous wherever $\nabla f(x)$ is continuous, with $\|\nabla\beta(f(x); y_1, y_2)\| = 0$ if $f(x) = y_1$. The continuity of the gradients can be verified by differentiating the component functions in Equation (2.10), which are shown later in Equation (2.17). In other words, $\beta(f(x); y_1, y_2)$ realizes our goals stated above, as it is a continuous barrier function that can be added to any objective function without introducing any nonsmooth points of its own, that is, points where the gradient is not defined. However, if $\hat{x}$ is a nonsmooth point of $f(x)$ with $f(\hat{x}) \in (y_1, y_2)$, then naturally $\beta(f(x); y_1, y_2)$ must also be nonsmooth at $\hat{x}$.

The values y and h determine exactly where $\beta(f(x); y_1, y_2)$ switches between its second and third cases, that is, where the cubic polynomial meets the log-based function when $f(x) = y$ and $\beta(f(x); y_1, y_2) = h$. While the latter monotonically increases with respect to $f(x)$ increasing, note that this is not necessarily guaranteed for the cubic polynomial. However, this can be enforced with a judicious choice of y. For example, if we choose to set $\tau_2 = 0$, and consider $f(x) = x$ (so $x \in \mathbb{R}$), then the cubic polynomial and its first derivative are always increasing for all $y > y_1$. We can then simply solve for y by setting the numerator of τ_2 in Equation (2.11) equal to zero, which yields $y = y_1 + \frac{3h}{3h+1}\delta$, where $\delta = y_2 - y_1 > 0$.

We now modify and extend Equation (2.9) to, respectively, make it continuous via our barrier function and support damping multiple frequency bands. Suppose multiple undesirable frequency bands are specified by frequencies $\{\omega_1, \dots, \omega_k\}$ and associated (half) bandwidths $\{b_1, \dots, b_k\}$, with their relative importance determined by $\{\phi_1, \dots, \phi_k\}$, where $\omega_j \ge 0$, $b_j > 0$, and $\phi_j \in (0, 1]$ for all $j = 1, \dots, k$. Then given some $\eta \ge 0$ and the corresponding ellipses $\{E_1, \dots, E_k\}$ with $E_j = (\sim, b_j, \eta + \mathbf{i}\omega_j)$, our second model for optimizing viscosities is

$$\textbf{Model 2}: \quad \begin{aligned} \max_{v \in \mathbb{R}^r} \quad & \left(\sum_{j=1}^{k} \phi_j \cdot \min\{a_{\Lambda,E_j}(v), m_j\}\right) - \beta(\alpha_{\text{MCK}}(v); \texttt{tol}_{\text{sa}}, \eta) \\ \text{s.t.} \quad & \alpha_{\text{MCK}}(v) \le \texttt{tol}_{\text{sa}} \text{ for some } \texttt{tol}_{\text{sa}} < 0, \\ & v_j \ge 0 \text{ for } j = 1, \dots, r, \end{aligned} \tag{2.12}$$

where we use $h := 1$ and $y := \texttt{tol}_{\text{sa}} + \frac{3}{4}(\eta - \texttt{tol}_{\text{sa}})$ for our barrier function (2.10) and $m_j > 0$ is a fixed scalar denoting a desired upper bound on the damping of the jth frequency band, that is, a point at which the band can be considered sufficiently damped. The sum in the objective function of Equation (2.12) acts to push all eigenvalues to the left of the intervals $\mathbf{i}[\omega_j - b_j, \omega_j + b_j]$ farther to the left, namely, by trying to increase each of the semi-major axis values of the ellipses (while still having no eigenvalues inside them) as much as possible or until they are at least as large as the respective m_j values. The inclusion of the finite m_j scalars prevent optimization terminating due to one of the $a_{\Lambda,E_j}(v)$ functions becoming infinite, which happens if all the eigenvalues can be moved completely outside of one or more of the specified frequency bands. This can be undesirable because when this happens, the other frequency bands may or may not be well-optimized. Using $\min\{a_{\Lambda,E_j}(v), m_j\}$ prevents this from occurring, and so all the frequency bands will continue to be optimized. Meanwhile, the ϕ_j scalars allow one to balance which frequency bands should be given the most emphasis during optimization. By construction, our barrier function only has an effect when the spectral abscissa stability constraint is violated, and so it does not modify our objective function on the feasible set. As $\beta(\alpha_{\text{MCK}}(v); \texttt{tol}_{\text{sa}}, \eta)$ goes continuously to infinity as the spectral abscissa approaches η, the discontinuities of $a_{\Lambda,E_j}(v)$ functions can never be encountered and

having eigenvalues with real parts close to η is strongly penalized, which helps to avoid regions where $a_{\Lambda,E_j}(v)$ is close to being discontinuous. Compared to our first model in Section 2.1, we have introduced the parameter $\eta \geq 0$ here so that if desired, the distance between being stable to tolerance and the discontinuities of $a_{\Lambda,E_j}(v)$ can be increased by shifting all the ellipses to the right.

2.3 | Solving our optimization problems

A key goal realized by our new constrained optimization problems for frequency-weighted damping, respectively, given in Equations (2.6) and (2.12), is that they are both continuous, unlike the formulation we first considered in Equation (2.1). Consequently, as mentioned earlier, we thus can consider computing solutions to Equations (2.6) and (2.12) using recent gradient-based solvers for continuous nonsmooth constrained optimization, where the nonsmoothness of the functions is limited to a set of zero. This is appealing because such gradient-based nonsmooth solvers not only exhibit good performance in practice but are also easy to use, as they only require that gradients be provided; see section 6 of Curtis et al. [21] for some comparisons. The necessary gradients exist because typically such methods only encounter the nonsmooth manifold in the limit, and so while iterates may be arbitrarily close to nonsmooth points, they are nevertheless not nonsmooth points themselves. Two possible gradient-based solvers for nonsmooth constrained optimization are the open-source software packages SQP-GS [20] and GRANSO: GRadient-based Algorithm for Non-Smooth Optimization [21]. For the purposes of this paper, we use GRANSO[1] to compute solutions to Equations (2.6) and (2.12), partly because the per-iteration cost of GRANSO is much less than that of SQP-GS. We now discuss how to compute the necessary gradients.

For our first approach, given by Equation (2.6), we need the gradient of the spectral abscissa and $d_{\Lambda,\mathcal{E}}(v)$. We begin with the former. Let $\lambda(v)$ be an eigenvalue of Equation (1.4) with associated eigenvector $x(v)$. Since M, $C(v)$, and K are real symmetric matrices, $x(v)$ is also the left eigenvector for $\lambda(v)$. Then given some $\hat{v}$, if $\lambda(\hat{v})$ is a simple eigenvalue with eigenvector by $\hat{x}$, by standard perturbation theory for eigenvalues we have that

$$\left.\frac{\partial \lambda(v)}{\partial v_j}\right|_{v=\hat{v}} = -\frac{\hat{x}^*\left(\lambda(\hat{v})g_j g_j^\top\right)\hat{x}}{\hat{x}^*(2\lambda(\hat{v})M + C(\hat{v}))\hat{x}}. \tag{2.13}$$

Furthermore, if $\lambda(\hat{v})$ is also an eigenvalue that attains the spectral abscissa and there are no other eigenvalues with real part equal to $\operatorname{Re}\lambda(\hat{v})$, that is, there are no ties (excluding conjugacy) for the spectral abscissa, then

$$\left.\frac{\partial \alpha_{\mathrm{MCK}}(v)}{\partial v_j}\right|_{v=\hat{v}} = -\operatorname{Re}\left.\frac{\partial \lambda(v)}{\partial v_j}\right|_{v=\hat{v}}. \tag{2.14}$$

We now turn to $d_{\Lambda,\mathcal{E}}(v)$. Given a single ellipse given by $E = (a, b, c)$, consider $d(z(t); E)$ defined by Equation (2.3), where $z(t)$ is a differentiable path with respect to the real scalar t. Then the derivative of $d(z(t); E)$ is

$$d'(z(t); E) = 2\left(\frac{\operatorname{Re}(z(t) - c) \cdot \operatorname{Re} z'(t)}{a^2} + \frac{\operatorname{Im}(z(t) - c) \cdot \operatorname{Im} z'(t)}{b^2}\right). \tag{2.15}$$

Now given $\hat{v}$, suppose there are no ties for the value of $d_{\Lambda,\mathcal{E}}(\hat{v})$, that is, its value is attained by a single eigenvalue $\lambda(\hat{v})$ and ellipse $E = (a, b, c) \in \mathcal{E}$, with $\lambda(\hat{v})$ being simple. Then the gradient of $d_{\Lambda,\mathcal{E}}(v)$ at $\hat{v}$ exists, and the partial derivative with respect to v_j at $\hat{v}$ can be constructed via Equation (2.15), where $z(t)$ is replaced by $\lambda(\hat{v})$ and $z'(t)$ is replaced by the partial derivative of $\lambda(v)$ at $\hat{v}$ given in Equation (2.13).

For our second approach, given by Equation (2.12), we have shown above how to obtain gradient of the spectral abscissa, which leaves the objective function in Equation (2.12). Given an ellipse defined by $E = (\sim, b, \eta + \mathbf{i}\omega)$, again consider $z(t)$ described above but additional suppose that $\operatorname{Im} z(t) \in (\omega - b, \omega + b)$. Then $a(z(t); E)$ cannot be infinite and its derivative is

$$a'(z(t); E) = \frac{b \operatorname{sgn}(\operatorname{Re} z(t) - \eta) \cdot \operatorname{Re} z'(t)}{(b^2 - (\operatorname{Im} z(t) - \omega)^2)^{1/2}} + \frac{b|\operatorname{Re} z(t) - \eta|(\operatorname{Im} z(t) - \omega) \cdot \operatorname{Im} z'(t)}{(b^2 - (\operatorname{Im} z(t) - \omega)^2)^{3/2}} \tag{2.16}$$

Now consider $a_{\Lambda,E_j}(\upsilon)$, which is differentiable if there is a single eigenvalue (up to conjugacy) on the ellipse specified by E_j and this eigenvalue is simple. If these assumptions hold at $\hat{\upsilon}$ and this eigenvalue is $\lambda(\hat{\upsilon})$, then partial derivative with respect to υ_j of $a_{\Lambda,E_j}(\upsilon)$ at $\hat{\upsilon}$ is given by Equation (2.16) with $z(t)$ and $z'(t)$ are again replaced using $\lambda(\hat{\upsilon})$ and Equation (2.13). For the gradient of our barrier function, it suffices to show the derivative of $\beta(f(x); y_1, y_2)$, where $f : \mathbb{R} \mapsto \mathbb{R}$ is differentiable:

$$\beta'(f(x); y_1, y_2) := \begin{cases} 0 & \text{if } f(x) \le y_1, \\ (3\tau_1(f(x) - y_1)^2 + 2\tau_2(f(x) - y_1)) \cdot f'(x) & \text{if } f(x) \in (y_1, y], \\ \frac{f'(x)}{y_2 - f(x)} & \text{if } f(x) \in (y, y_2), \\ \text{undefined} & \text{otherwise.} \end{cases} \tag{2.17}$$

Then given $\hat{\upsilon}$ and assuming $\alpha_{\text{MCK}}(\upsilon)$ is differentiable at $\hat{\upsilon}$, the partial derivative with respect to υ_j of our barrier function in Equation (2.12) is given by Equation (2.17), where $f(x)$ and $f'(x)$ are, respectively, replaced using $\alpha_{\text{MCK}}(\hat{\upsilon})$ and Equation (2.14), $y_1 = \texttt{tol}_{\text{sa}}$, and $y_2 = \eta$.

3 | FAST SOLUTION OF QEPS WITH LOW-RANK STRUCTURE

The most expensive part of our approaches proposed in Section 2 is successively computing the eigenvalues and eigenvectors of Equation (1.4) as the viscosities are optimized, that is, as $C(\upsilon)$ is changed. One possibility is to use `polyeig` in MATLAB or `quadeig`; see Refs. [14, 22] for more details on these methods. However, using either of these routines would mean that solving each QEP would require roughly the same amount of cubic work, that is, $\mathcal{O}(n^3)$, where we use the usual convention of treating eigenvalue computations as atomic operations. In Ref. [23], Taslaman proposed a much faster eigensolver for QEPs (1.4), where the damping matrix $C(\upsilon)$ is assumed to be low rank. While the overall work complexity of Taslaman's algorithm is still cubic, in experiments (see Section 5 of [23]), it was shown to be many times faster than `quadeig`, and its work can be separated into offline and online components, with the latter only doing $\mathcal{O}(n^2)$ work. Taslaman's algorithm is based on Ehrlich–Aberth iterations, which rely on a good choice of a starting point for each eigenvalue and for which determination of stopping criteria is often heuristic; for more details, see Refs. [24] and [23]. Shortly thereafter, Taslaman's algorithm was extended by Benner and Denißen [25] to systems where $C(\upsilon) = C_{\text{int}} + C_{\text{ext}}(\upsilon)$ may be full rank, but critical damping is used for the internal damping matrix C_{int} and the external damping matrix $C_{\text{ext}}(\upsilon)$ is low rank.

In this section, for the same class of problems considered by Benner and Denißen, we also exploit the fact that changes in $C(\upsilon)$ are only low-rank updates, but we propose a new fast algorithm for efficiently solving such QEPs using a rather different approach. Our new method also does cubic work once in an offline initialization phase and only a quadratic amount of work in the online phase. Since many QEPs will typically be solved during the course of optimizing the viscosities, our approach here can result in a significant speedup. At a high level, we propose computing the eigenvalues and eigenvectors of Equation (1.4) by transforming this QEP into a small sequence of linear eigenvalue problems involving DPR1 matrices. By solving these linear subproblems, we can then recover the eigenvalues and eigenvectors of Equation (1.4). Moreover, as these DPR1 matrices are easily converted to DPR1Csym matrices, we also leverage this special structure in a new fast eigensolver in order to be much more efficient than standard eigenvalue techniques.

3.1 | Efficient eigenvalue computation for DPR1Csym matrices

Let $A \in \mathbb{C}^{2n\times 2n}$ be a DPR1Csym matrix, that is,

$$A = D + \rho z z^{\mathsf{T}}, \tag{3.1}$$

where $D = \text{diag}(d_1, d_2, \ldots, d_{2n}) \in \mathbb{C}^{2n\times 2n}$ is invertible (so $d_i \neq 0\ \forall i$), $z = [z_1\ z_2\ \cdots\ z_{2n}]^{\mathsf{T}} \in \mathbb{C}^{2n}$, and $\rho > 0$. Note that it is not necessary to consider $\rho \le 0$, since if $\rho = 0$, then A is diagonal and so obtaining its eigenvalues is trivial, while if $\rho < 0$, then one can just instead consider $A = -D - \rho z z^{\mathsf{T}}$. Furthermore, we assume that

- A is irreducible, that is, $\forall i, j \in \{1, \dots, 2n\}$, $z_i \neq 0$ and $d_i \neq d_j$ if $i \neq j$, and
- A is diagonalizable.

It is unnecessary to consider reducible A matrices, since d_i is an eigenvalue of $D + \rho zz^\top$, with its corresponding eigenvector being the ith canonical vector, if and only if $z_i = 0$ or $d_i = d_j$ for some $j \neq i$ holds (see, e.g., Xu and Qiao [26]). In other words, such eigenvalues can be easily removed (via exact deflation) to obtain a smaller DPR1Csym matrix that is irreducible. Per the following remark, we will be able to convert eigenvalue problems involving DPR1 matrices into ones involving DPR1Csym matrices.

Remark 1. Note that if $D + \rho uz^\top \in \mathbb{C}^{2n\times 2n}$ is a DPR1 matrix with $u, z \in \mathbb{C}^{2n}$ and $u_i \neq 0$, $z_i \neq 0\ \forall i$, then it can be rewritten as a DPR1Csym matrix with the same eigenvalues. Letting

$$S := \operatorname{diag}\left(\sqrt{\frac{z_1}{u_1}}, \dots, \sqrt{\frac{z_{2n}}{u_{2n}}}\right) \qquad \text{and} \qquad \hat{z} := Su, \tag{3.2}$$

it follows that $u^\top S^2 = z^\top$ holds, and so (λ, x) is an eigenpair of $D + \rho uz^\top$ if and only if (λ, Sx) is an eigenpair of $D + \rho\hat{z}\hat{z}^\top$. While this transformation requires that u and z only have nonzero entries, this is also easily ensured via a preprocessing step. If $u_i = 0$ or $z_i = 0$ for some i, then d_i is an eigenvalue of D and it can be removed via exact deflation. Thus, by first performing a sequence of exact deflations corresponding to the zero entries of u and z, we extract the associated eigenvalues (and eigenvectors) and obtain a smaller DPR1 matrix that can be converted to a DPR1Csym matrix.

Thus, with our assumptions above, we need only consider the case of computing eigenvalues and eigenvectors of DPR1Csym matrices. If we were only to consider DPR1 real symmetric matrices, then fast standard techniques can be used that exploit the DPR1 structure, for example, divide-and-conquer, where the eigenvalues and eigenvectors of a tridiagonal matrix are computed by solving a sequence of eigenvalue problems involving DPR1 real symmetric matrices; see Cuppen [27] and chapter 5.3.3 of Demmel [28]. Of course, the essential properties needed to employ such methods are not present for DPR1Csym matrices, the most important being that the diagonal elements of D and the eigenvalues of A are no longer interlaced for the complex problem, since these values are now in the complex plane as opposed to on the real line. Thus, we instead consider an approach for DPR1Csym matrices that is inspired by a different approach for DPR1 real symmetric matrices [19]. The method of Jakovčević Stor et al. [19] computed eigenpairs using a combination of standard and modified Rayleigh quotient iterations (RQI and MRQI, respectively), but in our setting, the eigenvalues of Equation (3.1) will be complex (and real axis symmetry is not guaranteed), and we have observed that standard RQI often does not converge. Moreover, we have also observed that when eigenvalues are close to each other, the method of Jakovčević Stor et al. [19] often gets stuck oscillating between approximations in such clusters of eigenvalues. To address these shortcomings, we propose two key modifications, namely, to completely forgo using standard RQI and to introduce a new dynamic step-size procedure in order to steer our MRQI-based procedure towards a single eigenvalue in a cluster. We now present our new method in complete detail.

Since A is complex symmetric and diagonalizable, we have the following eigendecomposition:

$$A = W\Lambda W^\top, \tag{3.3}$$

where $\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_{2n})$ and $W = \begin{bmatrix} w_1 & \cdots & w_{2n} \end{bmatrix}$ with $W^\top W = I$ are, respectively, the eigenvalues and eigenvectors of A. Note that the eigenvalues of A are the zeros of the secular function (see e.g., Cuppen [27]):

$$f(\lambda) = 1 + \rho \sum_{i=1}^{2n} \frac{z_i^2}{d_i - \lambda} = 1 + \rho z^\top (D - \lambda I)^{-1} z, \tag{3.4}$$

where for $i \in \{1, \dots, 2n\}$, the eigenvector w_i for eigenvalue λ_i is given by

$$w_i = \frac{x_i}{\|x_i\|_2} \quad \text{with} \quad x_i = (D - \lambda_i I)^{-1} z. \tag{3.5}$$

The zeros of Equation (3.4) can be found using different algorithms, for example, if A is real, the eigenvalues can be efficiently and reliably computed via bisection [29]. If A is a DPR1 matrix, one can use, for example, `mpsolve` from the package MPSolve (see Bini and Robol [30]), but this can be costly since `mpsolve` uses a large amount of extra digits of precision (as opposed to just quad precision). If A is a complex symmetric matrix, one can use MRQI; see Refs. [31, 32]. Regarding the eigenvector formula given in Equation (3.5), this is well known to be numerically unstable, but one option to work around this problem is to use extended precision; for the DPR1 eigensolver of Jakovčević Stor et al. [29], a fraction of the algorithm is implemented in quad precision, and the authors reported that overhead to use this extended precision was very modest, that is, only 55% slower than standard double-precision implementations; see p. 314 of Jakovčević Stor et al. [29],

In our case of A being DPR1Csym, we can consider a modification of the MRQI approach of Jakovčević Stor et al. [19] that both additionally takes advantage of its DPR1 structure for efficiency, and introduces our new step-size procedure to improve the reliability of convergence. Given a starting $x \in \mathbb{C}^{2n}$, repeat

$$\delta \leftarrow \eta \frac{x^{\mathsf{T}} A x}{x^{\mathsf{T}} x}, \quad x \leftarrow (D - \delta I)^{-1} z, \tag{3.6}$$

where $\eta > 0$ is a step size chosen dynamically to enhance convergence to a single eigenvalue. The computation of x comes from the eigenvector formula (3.5). Once δ has converged, it can be deflated from A to obtain a new smaller DPR1Csym matrix; see Pan and Zheng [33]. Without loss of generality, assume eigenvalue λ is computed via shift d_s from the diagonal of D. Then deflating λ from A results in the DPR1Csym matrix $A_{\mathrm{d}} \in \mathbb{C}^{2n-1 \times 2n-1}$, where

$$A_{\mathrm{d}} = D_{\mathrm{d}} + \rho z_{\mathrm{d}} z_{\mathrm{d}}^{\mathsf{T}}, \quad \text{where} \tag{3.7a}$$

$$D_{\mathrm{d}} = \operatorname{diag}(d_1, \dots, d_{s-1}, d_{s+1}, \dots, d_{2n}), \tag{3.7b}$$

$$(z_{\mathrm{d}})_i = z_i \sqrt{\frac{d_i - d_s}{d_i - \lambda}}, \quad i = 1, \dots, s-1, s+1 \dots, 2n. \tag{3.7c}$$

The deflation formula comes from shifted inverse power method and Sherman–Morrison–Woodbury (SMW) formula, and A is always stored implicitly, as two vectors and a scalar. A detailed pseudocode for our new eigensolver for DPR1 matrices is given in Algorithm 1.

3.2 | Efficient eigenvalue computation for QEPs

We now show how to transform our QEP given by Equation (1.4) into multiple connected DPR1 eigenvalue problems. First, since M and K are symmetric positive definite matrices, there exists a matrix Φ which simultaneously diagonalizes M and K, that is,

$$\Phi^{\mathsf{T}} K \Phi = \Omega^2 = \operatorname{diag}(\omega_1^2, \dots, \omega_n^2) \quad \text{and} \quad \Phi^{\mathsf{T}} M \Phi = I, \tag{3.8}$$

where $\omega_1 > \cdots > \omega_n > 0$ are the undamped frequencies. Moreover, it can be shown that Φ also diagonalizes C_{int}, that is, $\Phi^{\mathsf{T}} C_{\mathrm{int}} \Phi = \alpha\Omega$; for more details, see Refs. [1, 10].[2] Thus, we can linearize the QEP given in Equation (1.4) to obtain the standard eigenvalue problem

$$A(v) y(v) = \lambda(v) y(v), \quad \text{where} \tag{3.9a}$$

$$A(v) = \begin{bmatrix} 0 & \Omega \\ -\Omega & -\Phi^{\mathsf{T}} C(v) \Phi \end{bmatrix} = \begin{bmatrix} 0 & \Omega \\ -\Omega & -\alpha\Omega \end{bmatrix} - \begin{bmatrix} 0 \\ \Phi^{\mathsf{T}} G \end{bmatrix} \begin{bmatrix} v_1 & & \\ & \ddots & \\ & & v_r \end{bmatrix} \begin{bmatrix} 0 & G^{\mathsf{T}} \Phi \end{bmatrix}, \tag{3.9b}$$

$$y(v) = \begin{bmatrix} \Omega \Phi^{-1} x(v) \\ \lambda(v) \Phi^{-1} x(v) \end{bmatrix}. \tag{3.9c}$$

ALGORITHM 1 Eigensolver for DPR1 matrices

Input: DPR1 matrix $D + \rho uz^{\mathsf{T}}$ with $\rho > 0$ and vectors $u, z \in \mathbb{C}^{2n}$ with no zero entries.

Output: Eigenvalues $\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_{2n})$ and eigenvectors $W = \begin{bmatrix} w_1 & \cdots & w_{2n} \end{bmatrix}$.

1: $S \leftarrow$ diagonal matrix from Equation (3.2)
2: $\hat{z} \leftarrow Su$
3: **for** $l = 2n, 2n-1, 2n-2, \dots, 2, 1$ **do**
4: $\quad s \leftarrow \operatorname{argmax}_{i \in \{1,\dots m\}} |d_i|$
5: $\quad \widehat{D} \leftarrow D - d_s I$
6: $\quad x \leftarrow e_s, \gamma \leftarrow 0, \eta \leftarrow 1$ (set initial values)
7: $\quad$ **while** $|\frac{\delta}{\gamma}| <$ `tol` (stop MRQI once relative change between steps is small) **do**
8: $\quad\quad$ **if** not converging **then**
9: $\quad\quad\quad \eta \leftarrow \frac{\eta}{2}$ (reduce η and reset other initial values)
10: $\quad\quad\quad x \leftarrow e_s, \gamma \leftarrow 0$
11: $\quad\quad$ **end if**
12: $\quad\quad \delta \leftarrow \eta x^T (\widehat{D} + \rho \hat{z}\hat{z}^{\mathsf{T}}) x / \|x\|^2$
13: $\quad\quad \gamma \leftarrow \gamma + \delta$
14: $\quad\quad x \leftarrow (\widehat{D} - \delta I)^{-1} \hat{z}$
15: $\quad\quad \widehat{D} \leftarrow \widehat{D} - \delta I$
16: $\quad$ **end while**
17: $\quad \lambda_l \leftarrow d_s + \gamma$
18: $\quad [D, \hat{z}] \leftarrow$ Update DPR1Csym matrix by deflating λ_l from $D + \rho \hat{z}\hat{z}^{\mathsf{T}}$ via Equation (3.7)
19: **end for**
20: $W \leftarrow \begin{bmatrix} w_1 & \cdots & w_{2n} \end{bmatrix}$, where w_l is computed by Equation (3.5) using λ_l
21: $W \leftarrow S^{-1} W$

Note: If $u = z$, then $S = I$ in line 1 and so $\hat{z} = z = u$. Since S is diagonal, the operations outside of the for loop amount to $\mathcal{O}(n^2)$ work, while each line inside is at most $\mathcal{O}(n)$ work. Thus, assuming that the number iterations of the while loop is never dependent on n, the total work complexity of Algorithm 1 is $\mathcal{O}(n^2)$.

Let $P \in \mathbb{R}^{2n \times 2n}$ be the perfect shuffle permutation, which splits a set of even cardinality into two sets of equal cardinality and interleaves them, that is, it maps the kth e s:

$$k \mapsto \begin{cases} 2k - 1, & \text{if } k \le n \\ 2(k - n), & \text{if } k > n. \end{cases} \tag{3.10}$$

Now dropping the dependency on υ for brevity and using $PP^{\mathsf{T}} = I$ and $\widehat{A} = P^{\mathsf{T}} A P$, multiplying Equation (3.9a) on the left by P^{T} yields the eigenvalue problem

$$\widehat{A} P^{\mathsf{T}} y = \left(\begin{bmatrix} D_1 & & \\ & \ddots & \\ & & D_n \end{bmatrix} - \widehat{G} \begin{bmatrix} \upsilon_1 & & \\ & \ddots & \\ & & \upsilon_r \end{bmatrix} \widehat{G}^{\mathsf{T}} \right) P^{\mathsf{T}} y = \lambda P^{\mathsf{T}} y, \quad \text{where} \tag{3.11a}$$

$$D_i = \begin{bmatrix} 0 & \omega_i \\ -\omega_i & -\alpha\omega_i \end{bmatrix} \quad \text{and} \quad \widehat{G} = P^{\mathsf{T}} \begin{bmatrix} 0 \\ \Phi^{\mathsf{T}} G \end{bmatrix}. \tag{3.11b}$$

Let Ψ_i be the matrix which diagonalizes matrix D_i and consider the matrices

$$\Psi = \begin{bmatrix} \Psi_1 & & \\ & \ddots & \\ & & \Psi_n \end{bmatrix}, \quad D = \Psi^{-1} \begin{bmatrix} D_1 & & \\ & \ddots & \\ & & D_n \end{bmatrix} \Psi, \quad U = \Psi^{-1} \widehat{G}, \quad \text{and} \quad Z = \Psi^{\mathsf{T}} \widehat{G}, \tag{3.12}$$

noticing that Ψ diagonalizes the block diagonal matrix from Equation (3.11a) into D. Thus, considering $\widetilde{A} = \Psi^{-1} \widehat{A} \Psi$ and multiplying Equation (3.11a) on the left by Ψ^{-1}, we further transform the eigenvalue problem into one involving a diagonal matrix plus a low-rank update

$$\widetilde{A} w = \lambda w, \quad \text{where} \tag{3.13a}$$

$$\widetilde{A} = D - U\begin{bmatrix} \upsilon_1 & & \\ & \ddots & \\ & & \upsilon_r \end{bmatrix} Z^{\mathsf{T}} = D - \sum_{j=1}^{r} \upsilon_j u_j z_j^{\mathsf{T}}, \quad w = \Psi^{-1} P^{\mathsf{T}} y, \tag{3.13b}$$

and u_j and z_j are, respectively, the jth columns of U and Z. Since matrices Φ, Ω, P, D, U, and Z are all independent of υ, obtaining the low-rank structure of $\widetilde{A}$ can be precomputed once in an offline process. In fact, Φ, Ω, P, and D are also independent of the damping positions specified by the matrix G, and so, when optimizing viscosities for multiple configurations of damping positions, these matrices need only be calculated once, while computing U and Z for each configuration is cheap.

We now show how Algorithm 1 can be iteratively applied to portions of $\widetilde{A}$ in order to recover all the eigenvalues and eigenvectors of Equation (3.13a). Let $\widetilde{A}_1 = D - \upsilon_1 u_1 z_1^{\mathsf{T}}$ and suppose it is diagonalizable, that is, it has eigendecomposition $\widetilde{A}_1 = \xi_1 L_1 \xi_1^{-1}$, where matrices ξ_1 and L_1, respectively, contain the eigenvectors and eigenvalues of $\widetilde{A}_1$. Then multiplying Equation (3.13a) on the left by ξ_1^{-1} and separating out $\widetilde{A}_1$, we obtain the transformed eigenvalue problem

$$\xi_1^{-1}\left(\widetilde{A}_1 - \sum_{j=2}^{r} \upsilon_j u_j z_j^{\mathsf{T}}\right) w = \left(L_1 - \sum_{j=2}^{r} \upsilon_j \xi_1^{-1} u_j z_j^{\mathsf{T}} \xi_1\right) \xi_1^{-1} w = \lambda \xi_1^{-1} w. \tag{3.14}$$

If $\widetilde{A}_2 = L_1 - \upsilon_2 \tilde{u}_2 \tilde{z}_2^{\mathsf{T}}$ is also diagonalizable, where $\tilde{u}_2 = \xi_1^{-1} u_2$ and $\tilde{z}_2 = \xi_1^{\mathsf{T}} z_2$, we can again similarly transform the eigenvalue problem via the eigendecomposition $\widetilde{A}_2 = \xi_2 L_2 \xi_2^{-1}$. We keep applying these transformations for $j = 1, \ldots, r$ by computing the eigendecompositions

$$\widetilde{A}_j = \xi_j L_j \xi_j^{-1}, \qquad \text{where} \qquad \widetilde{A}_j = L_{j-1} - \upsilon_j \tilde{u}_j \tilde{z}_j^{\mathsf{T}}, \qquad L_0 = D, \tag{3.15a}$$

$$\tilde{u}_j = \xi_{j-1}^{-1} \cdots \xi_1^{-1} u_j, \qquad \tilde{u}_1 = u_1, \tag{3.15b}$$

$$\tilde{z}_j = \xi_{j-1}^{\mathsf{T}} \cdots \xi_1^{\mathsf{T}} z_j, \qquad \tilde{z}_1 = z_1. \tag{3.15c}$$

Assuming that all the $\widetilde{A}_j$ matrices are indeed diagonalizable, we finally obtain

$$L_r\left(\xi_r^{-1} \cdots \xi_1^{-1} w\right) = \lambda\left(\xi_r^{-1} \cdots \xi_1^{-1} w\right), \tag{3.16}$$

and so we have recovered the eigenvalues of Equation (3.13a) and can reconstruct its eigenvectors as well.

Remark 2. As a final step of our algorithm, note that we also follow a suggestion of Taslaman (see section 4.3 of Taslaman [23]) to refine the accuracy of computed eigenvectors by doing a single step of inverse iteration for each eigenvector; due to the particular structure of $C(\upsilon)$, the SMW formula can be used to do a single-step of inverse iteration in only $\mathcal{O}(n)$ work per eigenvector. Similar applications of the SMW formula in damped systems for efficient computations can be found in Refs. [4, 5, 8, 34, 35].

Pseudocode for our complete QEP eigensolver is given in Algorithm 2. We note that assuming that all matrices $\widetilde{A}_j$ are diagonalizable is quite standard (see, e.g., Refs. [6, 19]), and we have not observed any issues in numerical experiments with this assumption.

We now turn to the work complexity of Algorithm 2. Recall that the work complexity of Algorithm 1 is $\mathcal{O}(n^2)$, and since we assume that the number of dampers r is small, that is, $r \ll n$, we will treat r as a constant. Inside the for loop of Algorithm 2, lines 5 and 6 are potentially more than $\mathcal{O}(n^2)$ work using standard techniques, but since the ξ_j's are Cauchy-like matrices, matrix-vector multiplication and linear solves can be done in approximately linear time, and so the total cost of the loop remains at $\mathcal{O}(n^2)$. Meanwhile, forming Ξ in line 9 is also not more than $\mathcal{O}(n^2)$ work, since P and Ψ are sparse matrices and $\xi_1 \cdots \xi_r$ is a product of linked Cauchy-like matrices, which can also be done in (approximately) linear time; for full details, see [19, 36]. In line 10, applying the inverse of Ω is cheap (recall that Ω is a diagonal matrix), while if Φ is a sparse matrix, then obtaining the full matrix of eigenvectors X is also at most $\mathcal{O}(n^2)$ work. However, note that while evaluating the functions in Equations (2.6) and (2.12) requires that we obtain all of the eigenvalues of Equation (1.4), only a handful of the corresponding eigenvectors are needed to compute the corresponding gradients. For our setting of optimizing viscosities using gradients, in line 10, we can selectively compute the handful of relevant eigenvectors, that

ALGORITHM 2 QEP eigensolver for Equation (1.4)

Input: M and K from Equation (1.4), $\alpha \geq 0$ for C_{int} from Equation (1.3), Φ and Ω from Equation (3.8), Ψ, D, U, and Z from Equation (3.12), and $v \in \mathbb{R}^r$.

Output: Eigenvalues L and eigenvectors X of QEP (1.4)

1: $L_0 \leftarrow D$
2: **for** $j = 1, \ldots, r$ **do**
3: $[\tilde{u}_j, \tilde{z}_j] \leftarrow j$th columns of U and Z, respectively
4: $[L_j, \xi_j] \leftarrow$ eigenvalues and eigenvectors of $L_{j-1} - v_j \tilde{u}_j \tilde{z}_j^{\mathsf{T}}$ computed by Algorithm 1
5: $U \leftarrow \xi_j^{-1} U$
6: $Z \leftarrow \xi_j^{\mathsf{T}} Z$
7: **end for**
8: $L \leftarrow L_r$
9: $\Xi \leftarrow P\Psi\xi_1 \cdots \xi_r$, where $P \in \mathbb{R}^{2n \times 2n}$ is the perfect shuffle permutation
10: $X \leftarrow \Phi\Omega^{-1}\Xi(1 : n, :)$ (Take the first n rows of Ξ)
11: $X \leftarrow$ each column (an eigenvector) of X gets refined according to Remark 2

Note: For simplicity of the pseudocodes in this paper, we assume that vectors $\tilde{u}_j$ and $\tilde{z}_j$ never have zero entries, scalar $v_j \neq 0$, and $L_{j-1} - v_j \tilde{u}_j \tilde{z}_j^{\mathsf{T}}$ is actually given to Algorithm 1 as $L_{j-1} + |v_j|(-\operatorname{sgn}(v_j)\tilde{u}_j)\tilde{z}_j^{\mathsf{T}}$ to adhere to its convention that $\rho > 0$. If $v_j = 0$, obtaining the eigenvalues and eigenvectors is immediate (so Algorithm 1 is not needed), while if either $\tilde{u}_j$ or $\tilde{z}_j$ contain zero entries, then per Remark 1, exact deflation is first used to remove the corresponding eigenvalues, which are on the diagonal of L_{j-1}, and then Algorithm 1 is called on the resulting deflated DPR1 matrix to obtain the remaining eigenvalues and eigenvectors. Finally, note that by design of Algorithms 1 and 2, for $j > 1$ in line 4, Algorithm 1 is warm started by using the eigenvalues of the previous DPR1 eigenvalue problem as shifts for the next DPR1 eigenvalue problem.

is, we apply Φ to the few corresponding columns of $\Omega^{-1}\Xi(1 : n, :)$ in order to not exceed $\mathcal{O}(n^2)$ work when Φ is dense. Finally, per Remark 2, refining the eigenvectors using inverse iteration is only $\mathcal{O}(n)$ work per eigenvector. Hence, the overall work complexity of Algorithm 2 is quadratic.

In terms of constant factors, the total cost of Algorithm 2 is dominated by line 4, that is, the call to Algorithm 1, which invoked a total of r times. As such, provided n is small enough, it is not always critical to implement lines 5, 6, 9, and 10 in Algorithm 2 as discussed above in order to attain the theoretical work complexity result (but those steps should indeed be taken if n is very large). Also, we note that the subsequent calls to Algorithm 1 can be faster than the first one because, by our design of the two algorithms, we are actually warm starting Algorithm 1 by choosing the initial shifts as the eigenvalues of the previous eigenvalue problem. Thus, when two consecutive DPR1 eigenvalue problems have quite similar spectra, which is not uncommon, we end up having excellent initial shift choices for which to accelerate the convergence of Algorithm 1 in line 4 of Algorithm 2.

4 | THE FREQUENCY-WEIGHTED DAMPING OPTIMIZATION ALGORITHM

We are now ready to present our new algorithm for frequency-weighted damping of QEPs. We begin with the offline phase, which simply precomputes the matrices from Equations (3.8) and (3.12) so that Algorithm 2 can be used to evaluate all the eigenvalue-based functions (and their gradients) that appear in Equations (2.6) and (2.12). While this offline phase has a $\mathcal{O}(n^3)$ work complexity, it only needs to be done once. We emphasize again that the simultaneous diagonalization part of the offline phase is also independent of the damping positions, and so it only needs to be performed once for all different damping positions.

For the online phase, by using Algorithm 2, evaluating all the functions (and their gradients described in Section 2.3) in Equation (2.6) or Equation (2.12) for a given vector $v \in \mathbb{R}^r$ is then only $\mathcal{O}(n^2)$ work, as opposed to $\mathcal{O}(n^3)$ via standard eigensolvers. In terms of the overall cost, this is a significant savings as we expect to require many function evaluations before converging to a stationary point of Equation (2.6) or Equation (2.12), particularly since these are nonsmooth optimization problems. To find solutions of Equations (2.6) and (2.12), we use GRANSO; a high-level description of our method is given in Algorithm 3.

ALGORITHM 3 Frequency-weighted damping optimization algorithm

Input: M and K from Equation (1.4), $\alpha \geq 0$ for C_{int} and G from Equation (1.2), set of k ellipses $\mathcal{E}$, weights $[\phi_1, \ldots, \phi_k]$ with each $\phi_j \in (0, 1]$ for ellipse $E_j \in \mathcal{E}$, $\eta \geq 0$, $\texttt{tol}_{\text{sa}} < 0$, initial viscosity values $\upsilon_{\text{init}} \in \mathbb{R}^r_+$, and $\texttt{approach} \in \{1, 2\}$.

Output: Computed for optimized viscosities $\upsilon_{\text{opt}} \in \mathbb{R}^r_+$ for either Equation (2.6) or Equation (2.12)

1: **Offline stage:** (Set up for computing functions and gradients via Algorithm 2)
2: $[\Phi, \Omega] \leftarrow$ matrices from Equation (3.8) (Diagonalization)
3: $[\Psi, D, U, Z] \leftarrow$ matrices from Equation (3.12) (Linearize and construct low-rank structure)
4: **Online stage:** (Optimize viscosities using GRANSO and Algorithm 2)
5: **if** $\texttt{approach} = 1$ **then**
6: $\quad \upsilon_{\text{opt}} \leftarrow$ solution returned by GRANSO for Equation (2.6) initialized at υ_{init}
7: **else**
8: $\quad \upsilon_{\text{opt}} \leftarrow$ solution returned by GRANSO for Equation (2.12) initialized at υ_{init}
9: **end if**

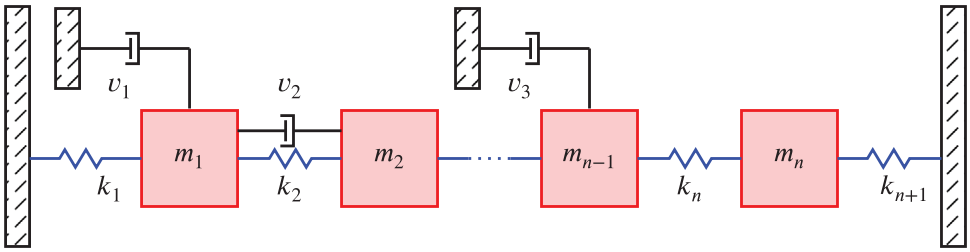


FIGURE 1 Diagram of an n-mass oscillator

5 | NUMERICAL EXPERIMENTS

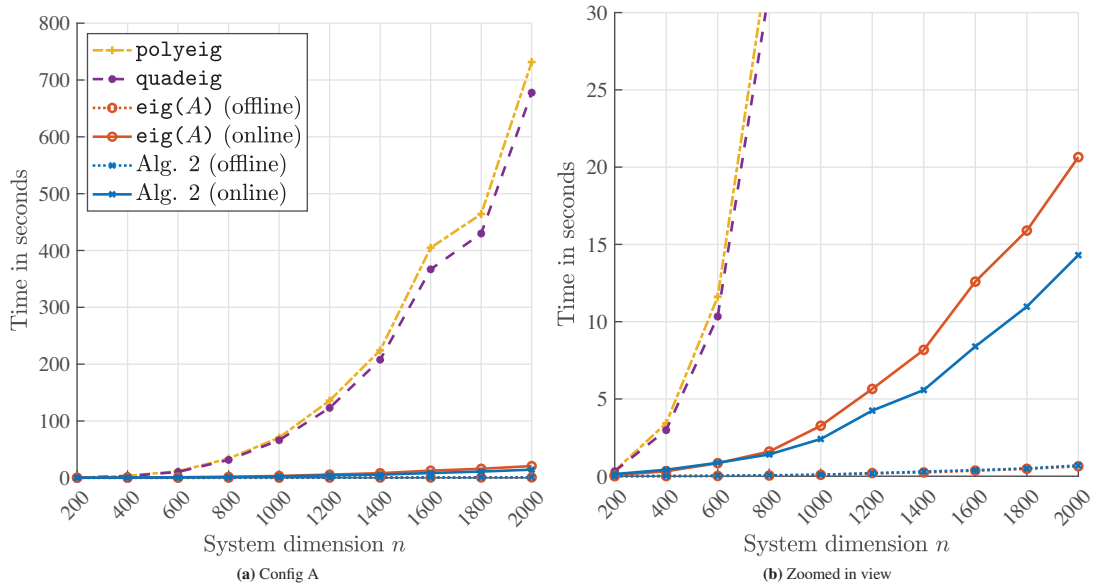
All experiments were done in MATLAB R2021a using a mid-2020 13″ MacBook Pro with an Intel Core i5-1038NG7 CPU (quad core) and 16GB of RAM running macOS 10.15.7. Our code for replicating all experiments reported here is both included as supplementary material with this article and available in a permanent and public archive on Zenodo.[3] For the values of n in our experiments here, it sufficed to implement lines 5, 9, and 10 of Algorithm 2 using standard techniques and compute all the eigenvectors, as opposed to leveraging the Cauchy-like structure and possibly selectively computing eigenvectors. As test problems, we used various instances of an n-mass oscillator; see Figure 1. For this mechanical system, we have the following matrices:

$$M = \operatorname{diag}(m_1, m_2, \ldots, m_n), \tag{5.1a}$$

$$K = \begin{bmatrix} k_1 + k_2 & -k_2 & & \\ -k_2 & \ddots & \ddots & \\ & \ddots & \ddots & -k_n \\ & & -k_n & k_n + k_{n+1} \end{bmatrix}, \tag{5.1b}$$

$$C_{\text{ext}}(\upsilon) = \upsilon_1 e_j e_j^{\mathsf{T}} + \upsilon_2 (e_k - e_{k+1})(e_k - e_{k+1})^{\mathsf{T}} + \upsilon_3 e_l e_l^{\mathsf{T}}, \tag{5.1c}$$

where e_j denotes the jth canonical vector, and $\upsilon_1, \upsilon_2, \upsilon_3 \geq 0$ are the viscosity values. In $C_{\text{ext}}(\upsilon)$, the $e_j e_j^{\mathsf{T}}$ and $e_l e_l^{\mathsf{T}}$ terms, respectively, mean that there are grounded dampers on masses m_k and m_l, while $(e_k - e_{k+1})(e_k - e_{k+1})^{\mathsf{T}}$ indicates that masses m_k and m_{k+1} are connected by a damper. Thus, for Figure 1, $C_{\text{ext}}(\upsilon)$ is defined using $j = k = 1$ and $l = n - 1$. Considering Equation (1.2), we also have that $C_{\text{ext}}(\upsilon) = G \operatorname{diag}(\upsilon_1, \upsilon_2, \upsilon_3) G^{\mathsf{T}}$, where $G = [e_j \quad e_k - e_{k+1} \quad e_l]$.



(a) Config A **(b)** Zoomed in view

FIGURE 2 The overall running times as the system dimension n is increased. For the direct solvers, `polyeig` and `quadeig`, the overall running times to solve instances of Equation (1.4) are shown. For the indirect solvers, `eig(A)` and Algorithm 2, the overall running times are separated into their offline and online parts, where the offline cost for both is computing Φ from Equation (3.8).

5.1 | Validating Algorithm 2

To assess the efficiency and accuracy of our new eigensolver (Algorithm 2) for solving Equation (1.4), we used instances of Equation (5.1) with orders $n = 200, 400, 600, \ldots, 2000$. For each value of n, we defined matrix M using $m_i = 10 + 990(\frac{i-1}{n-1})$ for $i = 1, \ldots, n$ and matrix K using $k_i = 5$ for $i = 1, \ldots, n+1$, and created two problems with different configurations of dampers, by defining two $C(\upsilon) = C_{\text{int}} + C_{\text{ext}}(\upsilon)$ matrices. We used $\alpha = 0.004$ to define C_{int}, while the two versions of $C_{\text{ext}}(\upsilon)$ were defined via choosing j, k, and l as follows:

$$\text{Config A: } (j,k,l) = \left(\frac{n}{10}, \frac{3n}{10}, \frac{5n}{10}\right) \quad \text{and} \quad \text{Config B: } (j,k,l) = \left(\frac{3n}{10}, \frac{7n}{10}, \frac{9n}{10}\right). \tag{5.2}$$

Using randomly generated viscosity values for each n, specifically $\upsilon = 0.1 + \texttt{rand}(3, n)$, we solved the resulting QEPs with Algorithm 2 and other solvers for comparison purposes. For direct QEP solvers, we tested `polyeig` and `quadeig`. We also benchmarked Algorithm 2 against a much simpler version of our algorithm, which also first computes matrix Φ to simultaneously diagonalize M and K, per Equation (3.8), but then forgoes taking any advantage of low-rank structure and instead just computes the eigenvalues of $A(\upsilon)$ defined in Equation (3.9b) via calling `eig` on this standard eigenvalue problem; we refer to this simpler method as `eig(A)`.

In Figure 2, we show the overall running times, recorded using `tic` and `toc`, for all the different eigensolvers as n increases for Config A. As can be seen, the costs of the direct solvers, `polyeig` and `quadeig`, quickly increase as n does. Meanwhile, `eig(A)` and Algorithm 2 are much faster, with their respective costs also increasing at a much slower rate with respect to n. Indeed, already by $n = 400$, `eig(A)` and Algorithm 2 are about an order of magnitude faster than the direct solvers. Moreover, for $n = 2000$, `eig(A)` is about 35 times faster than `polyeig`, while Algorithm 2 is 51 times faster than `polyeig`. Comparing `eig(A)` and Algorithm 2 to each other (see Figure 2), we see the cost of the latter indeed grows more slowly with respect to n, and that hidden constant term in the work complexity for Algorithm 2 is not an issue for overall efficiency in practice. We note that an implementation of Algorithm 2 in a compiled language and that takes advantage of the Cauchy-like structure of the matrices ξ_j should be many times faster than our prototype implementation that we have used here, which recall, is coded in MATLAB and does not yet take advantage of Cauchy-like structure. We also performed the same scaling experiment for Config B, which resulted in plots very similar to those shown in Figure 2; as such, we omit these additional plots here.

In order to show the accuracy of Algorithm 2, we compared its computed eigenvalues with those computed by `polyeig`, and for each computed eigenvalue-eigenvector pair (λ, x), we computed the spectral norm of the left-hand side of Equa-

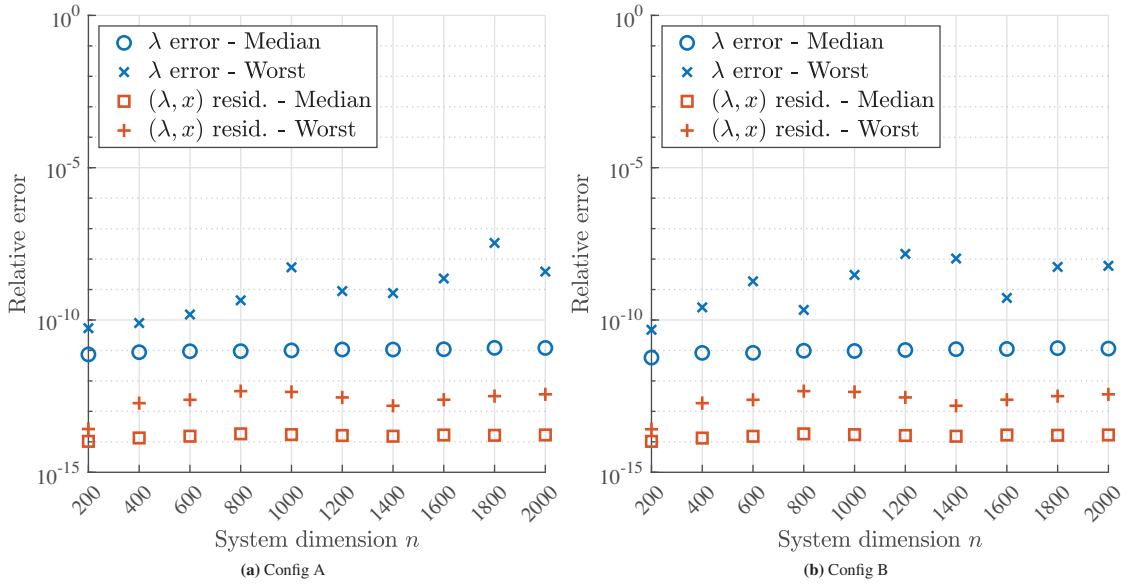


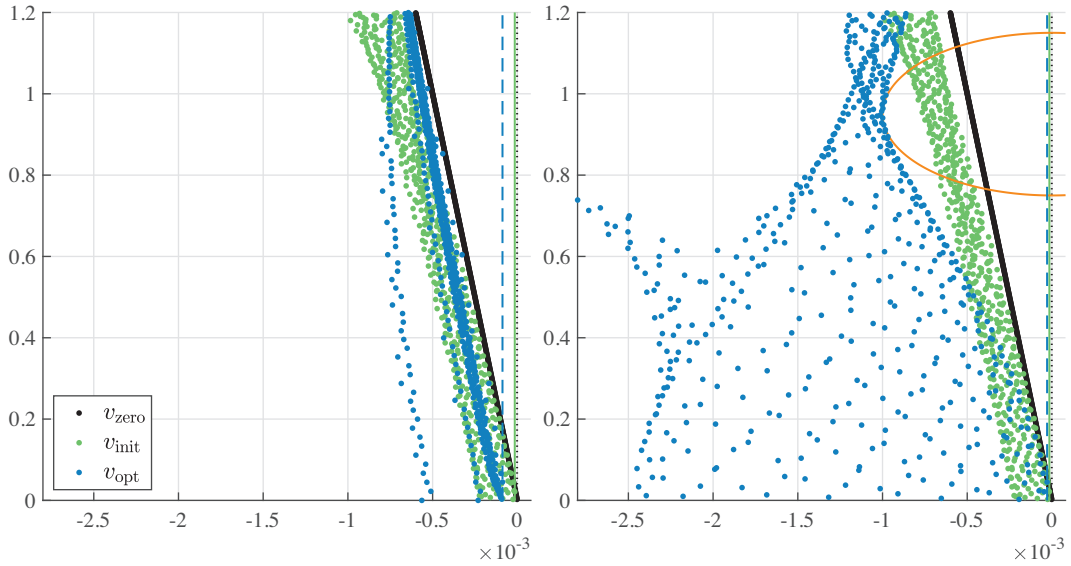
FIGURE 3 The median and worst relative errors of eigenvalues computed by Algorithm 2 with respect to the eigenvalues computed by `polyeig` are denoted by the "λ error" markers, while the median and worst eigenpair residuals are denoted by "(λ, x) resid." markers.

tion (1.4) with this eigenpair plugged into it. For each problem, we paired the two sets of computed eigenvalues greedily, that is, by taking the closest pair of values across the two sets, removing this "match," and then repeating this procedure until all computed eigenvalues were paired. For each matched pair of eigenvalues, we computed the relative errors in the real and imaginary parts separately, which we denote δ_{Re} and δ_{Im}, and then used $\max\{|\delta_{\text{Re}}|, |\delta_{\text{Im}}|\}$ as an overall measure of the error in the computed pair. Then for each problem, we computed the median and worst (largest) of these error measurements over the entire computed spectrum. Similarly, we computed the median and worst (largest) eigenvalue/eigenvectors errors over all the computed eigenpairs (λ, x), where the error of an eigenpair is given by taking the norm of the left-hand side of Equation (1.4). We show the resulting error measurements in Figure 3 for both Config A and Config B across all values of n tested. As can be seen, the results are essentially the same for both configurations. For comparing the accuracy of the eigenvalues with respect to those computed by `polyeig`, the median error was always about 10^{-11}, while the worst error rose from about 10^{-10} to a bit over 10^{-8} as n increased from 200 to 2000; we saw very similar eigenvalue errors when comparing Algorithm 2 to `quadeig` and even when comparing `polyeig` to `quadeig`. Meanwhile, the eigenpairs residuals were in the worst case still under 10^{-12} with the median error being about 10^{-14}, thus demonstrating that Algorithm 2 is indeed computing eigenvalues and eigenvectors to good accuracy. Note that our MATLAB implementation of Algorithm 2 only uses double precision and that implementing the key parts of Algorithm 1 using quad precision should improve the accuracy of Algorithm 2; in this case, Algorithm 1 and Algorithm 2 would be mixed-precision implementations.

5.2 | Validating Algorithm 3 for approaches 1 and 2

To demonstrate our new approaches for optimizing viscosities via nonsmooth constrained optimization and our new models, Approaches 1 (Fixed ellipses) and 2 (Variable ellipses), we used additional instances of the three-damper n-mass oscillator defined by the matrices in Equation (5.1). For these experiments, we used $n = 1000$ and defined M and K via, respectively, setting $m_i = m_{n+1-i} = \frac{2n-i}{200}$ for $i = 1, \ldots, \frac{n}{2}$ and $k_i = 5$ for $i = 1, \ldots, n+1$. To define $C(\upsilon)$, we used $(j, k, l) = (100, 400, 900)$ to specify the configuration of dampers given by matrix G in $C_{\text{ext}}(\upsilon)$ and used various values of α (to be reported momentarily) for C_{int}.

For the online optimization phase of Algorithm 3, we used GRANSO's default parameters except we set `opts.maxit=100`, always initialized GRANSO from υ_{init} = `ones(3,1)`, and set `opts.mu0=10000`. This last change, which multiplies the objective function by 10,000, was simply done for rescaling reasons, that is, so that the value of the objective function at υ_{init} was about one for all of our test problems; in practice, `opts.mu0` can be easily determined



(a) Spectral abscissa minimization only **(b)** Approach 1 (Fixed ellipses)

FIGURE 4 The eigenvalues (depicted as dots) are shown for v_{zero} (no external damping), v_{init} (the damping at the initial point), and v_{opt} (the optimized viscosities computed by GRANSO). The spectral abscissa for each set of viscosities is depicted via a vertical line: v_{zero} (dotted), v_{init} (solid), and v_{opt} (dashed). For Approach 1 (right), the ellipse defining our frequency-weighting constraint $d_{\Lambda,\mathcal{E}}(v) \geq 1$, which requires that none of the eigenvalues are inside the ellipse defined by $E = (0.001, 0.2, 0.95\mathbf{i})$, is also shown.

from the specific problem or one can use GRANSO's automatic pre-scaling feature. Since Equations (2.6) and (2.12) are generally nonconvex and thus may have multiple minimizers (of various quality), for best results in practice, one should initialize GRANSO from multiple starting points and take the best of the resulting computed solutions. Finally, for all problems and approaches 1 and 2, we set $\texttt{tol}_{sa} = 0.9 \cdot \min\{\alpha_{MCK}(v_{init}), \alpha_{MCK}(v_{zero})\}$, where $v_{zero} = \texttt{zeros(3,1)}$,

We begin with approach 1, where we used $\alpha = 0.001$ for C_{int} and defined constraint $d_{\Lambda,\mathcal{E}}(v) \geq 1$ from Equation (2.6) using a single ellipse, specifically $E = (0.001, 0.2, 0.95\mathbf{i})$. For comparison purposes, we also ran Algorithm 3 a second time on this same problem but without our ellipse constraint. In Figure 4, we show the different eigenvalue configurations before and after optimization. When only minimizing the spectral abscissa, GRANSO ran for seven iterations, while for Approach 1, GRANSO ran for 16 iterations. The solutions returned by GRANSO were, respectively,

$$v_{opt} = \begin{bmatrix} 238.7 \\ 101.2 \\ 132.6 \end{bmatrix} \quad \text{and} \quad v_{opt} = \begin{bmatrix} 8.117 \\ 8.187 \\ 0.001467 \end{bmatrix}. \tag{5.3}$$

From Figure 4, we clearly see that both of these solutions are close to the nonsmooth manifold, with the former resulting in several eigenvalues being close to attaining the spectral abscissa (the left pane) and the latter resulting in many more eigenvalues being exceptionally close to the boundary of our specified ellipse (the right pane). Moreover, we see that while the addition of constraint $d_{\Lambda,\mathcal{E}}(v) \geq 1$ in approach 1 causes the spectral abscissa to be minimized less, approach 1 did in fact move all of the eigenvalues at the initial viscosities v_{init} out of our ellipse region. In other words, approach 1 successfully computed a feasible set of viscosities that both selectively and significantly damped the desired frequency band.

Turning to approach 2, we used three ellipses to define our objective function in Equation (2.12), specifically $E_1 = (\sim, 0.05, 0.1\mathbf{i})$, $E_2 = (\sim, 0.05, 0.6\mathbf{i})$, and $E_3 = (\sim, 0.05, 1.1\mathbf{i})$, with respective weightings $\phi_1 = 1$, $\phi_2 = 0.2$, and $\phi_3 = 0.1$, and $\eta = 0$. We then ran Algorithm 3 using this instance of approach 2 on the same n-mass oscillator example that we used to test approach 1, except that now we used $\alpha = 0.004$ and $\alpha = 0.0004$. The configurations of eigenvalues before and after optimization are shown in Figure 5. For $\alpha = 0.004$ and $\alpha = 0.0004$, GRANSO, respectively, ran for 32 and 27 iterations

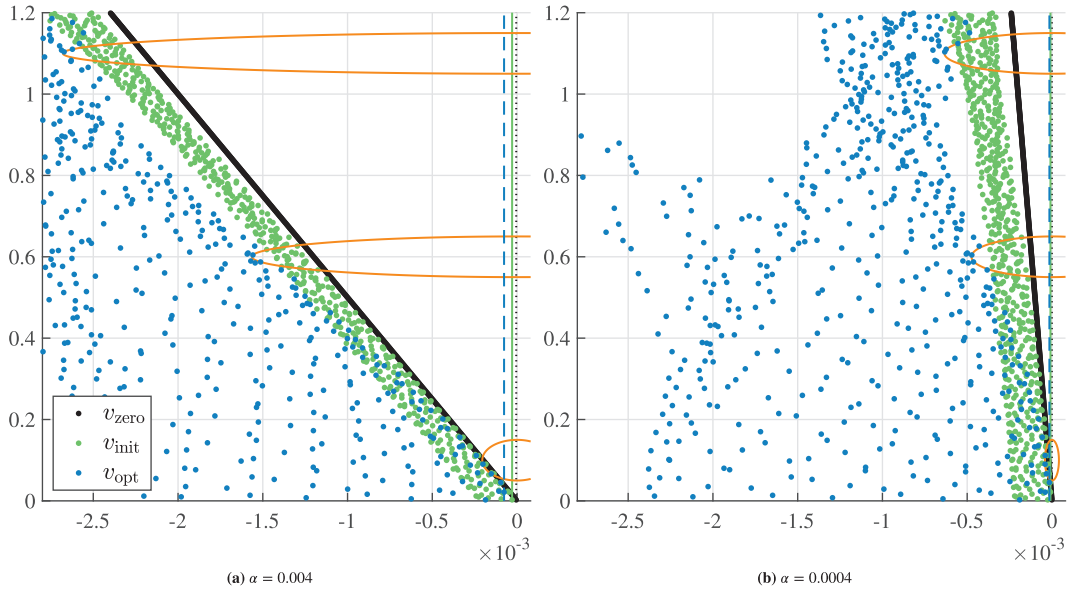


FIGURE 5 The resulting spectrum configurations and maximized ellipses computed by Approach 2 (Variable ellipses) are shown for two different choices of α determining the internal damping matrix C_{int}. See the caption of Figure 4 for the description of the legend.

before halting and, respectively, returned

$$v_{\text{opt}} = \begin{bmatrix} 8.138 \\ 7.147 \\ 1.789 \end{bmatrix} \quad \text{and} \quad v_{\text{opt}} = \begin{bmatrix} 8.295 \\ 7.767 \\ 1.673 \end{bmatrix}. \tag{5.4}$$

for the optimized viscosity values. We again see that the solutions returned by GRANSO are very close to the nonsmooth manifold. In the left pane of Figure 5, we see that each ellipse is essentially touching at least two eigenvalues, while in the right pane, the three ellipses are very close to touching three, four, and three eigenvalues, respectively, from top to bottom. Furthermore, the resulting eigenvalue configurations in Figure 5 confirm that approach 2 is indeed able to perform the desired frequency-weighted damping, as specified by the semi-minor axis values and centers of ellipses E_1, E_2, and E_3.

ACKNOWLEDGMENT

This work of the first author has been partially supported by Croatian Science Foundation under the project "Matrix Algorithms in Noncommutative Associative Algebras" (IP-2020-02-2240), while the work of the third and fourth authors has been partially supported by Croatian Science Foundation under the project "Vibration Reduction in Mechanical Systems" (IP-2019-04-6774).

Open Access funding enabled and organized by Projekt DEAL.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest.

ORCID

Nevena Jakovčević Stor https://orcid.org/0000-0002-9037-7631
Tim Mitchell https://orcid.org/0000-0002-8426-0242
Zoran Tomljanović https://orcid.org/0000-0002-3239-760X
Matea Ugrica https://orcid.org/0000-0002-8585-8967

ENDNOTES

[1] Available at https://gitlab.com/timmitchell/GRANSO/.

[2] While the eigensolver we propose in this section assumes that Φ diagonalizes C_{int}, note that our choice to use critical damping, that is, Equation (1.3), is not required. In particular, our approach can be applied to any internal damping that corresponds to a modally damped system, which is a usual assumption when vibrational mechanical systems are considered.

[3] Downloadable from https://doi.org/10.5281/zenodo.7338121.

REFERENCES

[1] Veselić, K.: Damped Oscillations of Linear Systems—A Mathematical Introduction. Lecture Notes in Math, vol. 2023. Springer-Verlag, Berlin (2011)

[2] Kuzmanović, I., Tomljanović, Z., Truhar, N.: Optimization of material with modal damping. Appl. Math. Comput. 218(13), 7326–7338 (2012). https://doi.org/10.1016/j.amc.2012.01.011

[3] Adhikari, S.: Damping modelling using generalized proportional damping. J. Sound Vib. 293(1–2), 156–170 (2006). https://doi.org/10.1016/j.jsv.2005.09.034

[4] Kuzmanović, I., Tomljanović, Z., Truhar, N.: Damping optimization over the arbitrary time of the excited mechanical system. J. Comput. Appl. Math. 304, 120–129 (2016). https://doi.org/10.1016/j.cam.2016.03.005

[5] Beattie, C., Gugercin, S., Tomljanović, Z.: Sampling-free model reduction of systems with low-rank parameterization. Adv. Comput. Math. 46(6), 83 (2020). https://doi.org/10.1007/s10444-020-09825-8

[6] Benner, P., Kürschner, P., Tomljanović, Z., Truhar, N.: Semi-active damping optimization of vibrational systems using the parametric dominant pole algorithm. Z. Angew. Math. Mech. 96(5), 604–619 (2016). https://doi.org/10.1002/zamm.201400158

[7] Blanchini, F., Casagrande, D., Gardonio, P., Miani, S.: Constant and switching gains in semi-active damping of vibrating structures. Int. J. Control 85(12), 1886–1897 (2012). https://doi.org/10.1080/00207179.2012.710915

[8] Tomljanović, Z., Beattie, C., Gugercin, S.: Damping optimization of parameter dependent mechanical systems by rational interpolation. Adv. Comput. Math. 44(6), 1797–1820 (2018). https://doi.org/10.1007/s10444-018-9605-9

[9] Tomljanović, Z., Voigt, M.: Semi-active $\mathcal{H}_\infty$ damping optimization by adaptive interpolation. Numer. Lin. Alg. Appl. 27(4), e2300 (2020). https://doi.org/10.1002/nla.2300

[10] Benner, P., Tomljanović, Z., Truhar, N.: Dimension reduction for damping optimization in linear vibrating systems. Z. Angew. Math. Mech. 91(3), 179–191 (2011). https://doi.org/10.1002/zamm.201000077

[11] Benner, P., Tomljanović, Z., Truhar, N.: Optimal damping of selected eigenfrequencies using dimension reduction. Numer. Lin. Alg. Appl. 20(1), 1–17 (2013). https://doi.org/10.1002/nla.833

[12] Cox, S.J., Nakić, I., Rittmann, A., Veselić, K.: Lyapunov optimization of a damped system. Systems Control Lett. 53(3), 187–194 (2004). https://doi.org/10.1016/j.sysconle.2004.04.004

[13] Gräbner, N., Mehrmann, V., Quraishi, S., Schröder, C., von Wagner, U.: Numerical methods for parametric model reduction in the simulation of disk brake squeal. Z. Angew. Math. Mech. 96(12), 1388–1405 (2016). https://doi.org/10.1002/zamm.201500217

[14] Tisseur, F., Meerbergen, K.: The quadratic eigenvalue problem. SIAM Rev. 43(2), 235–286 (2001)

[15] Joseph, K.T.: Inverse eigenvalue problem in structural design. AIAA J. 30(12), 2890–2896 (1992). https://doi.org/10.2514/3.11634

[16] Egaña, J.C., Kuhl, N.M., Santos, L.C.: An inverse eigenvalue method for frequency isolation in spring-mass systems. Numer. Linear Algebra Appl. 9(1), 65–79 (2002). https://doi.org/10.1002/nla.255

[17] Moro, J., Egaña, J.C.: Directional algorithms for the frequency isolation problem in undamped vibrational systems. Mech. Syst. Signal Process. 75, 11–26 (2016). https://doi.org/10.1016/j.ymssp.2015.12.030

[18] Kanno, Y., Puvača, M., Tomljanović, Z., Truhar, N.: Optimization of damping positions in a mechanical system. Rad Hrvat. Akad. Znan. Umjet. Mat. Znan. 23, 141–157 (2019). https://doi.org/10.21857/y26kec33q9

[19] Jakovčević Stor, N., Slapničar, I., Tomljanović, Z.: Fast computation of optimal damping parameters for linear vibrational systems. Mathematics 10(3), 790 (2022). https://doi.org/10.3390/10050790

[20] Curtis, F.E., Overton, M.L.: A sequential quadratic programming algorithm for nonconvex, nonsmooth constrained optimization. SIAM J. Optim. 22(2), 474–500 (2012). https://doi.org/10.1137/090780201

[21] Curtis, F.E., Mitchell, T., Overton, M.L.: A BFGS-SQP method for nonsmooth, nonconvex, constrained optimization and its evaluation using relative minimization profiles. Optim. Methods Softw. 32(1), 148–181 (2017). https://doi.org/10.1080/10556788.2016.1208749

[22] Hammarling, S., Munro, C.J., Tisseur, F.: An algorithm for the complete solution of quadratic eigenvalue problems. ACM Trans. Math. Software 39(3), 18 (2013). https://doi.org/10.1145/2450153.2450156

[23] Taslaman, L.: An algorithm for quadratic eigenproblems with low rank damping. SIAM J. Matrix Anal. Appl. 36(1), 251–272 (2015). https://doi.org/10.1137/140969099

[24] Bini, D.A., Noferini, V.: Solving polynomial eigenvalue problems by means of the Ehrlich–Aberth method. Linear Algebra Appl. 439(4), 1130–1149 (2013). 17th Conference of the International Linear Algebra Society, Braunschweig, Germany (2011). https://doi.org/10.1016/j.laa.2013.02.024

[25] Benner, P., Denißen, J.: Ehrlich-Aberth iteration for vibrational systems. In: Boltežar, M., Slavič, J., Wiercigroch, M. (eds.) Proceedings of ICoEV 2015, pp. 1540–1548. Ljubljana, Slovenia (2015)

[26] Xu, W., Qiao, S.: A divide-and-conquer method for the Takagi factorization. SIAM J. Matrix Anal. Appl. 30(1), 142–153 (2008). https://doi.org/10.1137/050624558

[27] Cuppen, J.J.M.: A divide and conquer method for the symmetric tridiagonal eigenproblem. Numer. Math. 36(3), 177–195 (1980). https://doi.org/10.1007/BF01396757

[28] Demmel, J.W.: Applied Numerical Linear Algebra. Society for Industrial and Applied Mathematics (SIAM), Philadelphia (1997)
[29] Jakovčević Stor, N., Slapničar, I., Barlow, J.L.: Forward stable eigenvalue decomposition of rank-one modifications of diagonal matrices. Linear Algebra Appl. 487, 301–315 (2015). https://doi.org/10.1016/j.laa.2015.09.025
[30] Bini, D.A., Robol, L.: Solving secular and polynomial equations: a multiprecision algorithm. J. Comput. Appl. Math. 272, 276–292 (2014). https://doi.org/10.1016/j.cam.2013.04.037
[31] Arbenz, P., Hochstenbach, M.E.: A Jacobi–Davidson method for solving complex symmetric eigenvalue problems. SIAM J. Sci. Comput. 25(5), 1655–1673 (2004). https://doi.org/10.1137/S1064827502410992
[32] Parlett, B.N.: The Rayleigh quotient iteration and some generalizations for nonnormal matrices. Math. Comp. 28(127), 679–693 (1974). https://doi.org/10.1090/S0025-5718-1974-0405823-3
[33] Pan, V.Y., Zheng, A.L.: New progress in real and complex polynomial root-finding. Comput. Math. Appl. 61(5), 1305–1334 (2011). https://doi.org/10.1016/j.camwa.2010.12.070
[34] Truhar, N., Veselić, K.: An efficient method for estimating the optimal dampers' viscosity for linear vibrating systems using Lyapunov equation. SIAM J. Matrix Anal. Appl. 31(1), 18–39 (2009). https://doi.org/10.1137/070683052
[35] Truhar, N., Tomljanović, Z., Veselić, K.: Damping optimization in mechanical systems with external force. Appl. Math. Comput. 250, 270–279 (2015). https://doi.org/10.1016/j.amc.2014.10.081
[36] Pan, V.Y., Zheng, A.: Superfast algorithms for Cauchy-like matrix computations and extensions. Linear Algebra Appl. 310(1), 83–108 (2000). https://doi.org/10.1016/S0024-3795(00)00041-0

How to cite this article: Jakovčević Stor, N., Mitchell, T., Tomljanović, Z., Ugrica, M.: Fast optimization of viscosities for frequency-weighted damping of second-order systems. Z Angew Math Mech. 103, e202100127 (2023). https://doi.org/10.1002/zamm.202100127