An efficient algorithm for damper optimization for linear vibrating systems using Lyapunov equation

Ninoslav Truhar

University of Osijek, Faculty of Civil Engineering, 31000 Osijek, Croatia,
email:truhar@gfos.hr

Abstract

We consider a second order damped-vibration equation $M\ddot{x} + D(\varepsilon)\dot{x} + Kx = 0$, where $M, D(\varepsilon), K$ are real, symmetric matrices of order n. The damping matrix $D(\varepsilon)$ is defined by $D(\varepsilon) = C_u + C(\varepsilon)$, where C_u presents internal damping and $\mathrm{rank}(C(\varepsilon)) = r$, where ε is dampers' viscosity.

We present an algorithm which derives a formula for the trace of the solution $\mathbf{X}$ of the Lyapunov equation $\mathbf{A}^T\mathbf{X} + \mathbf{X}\mathbf{A} = -\mathbf{B}$, as a function $\varepsilon \to Tr(\mathbf{Z}\mathbf{X}(\varepsilon))$, where $\mathbf{A} = \mathbf{A}(\varepsilon)$ is a $2n \times 2n$ matrix (obtained from $M, D(\varepsilon), K$) such that the eigenvalue problem $\mathbf{A}\mathbf{y} = \lambda\,\mathbf{y}$ is equivalent with the quadratic eigenvalue problem $(\lambda^2 M + \lambda D(\varepsilon) + K)x = 0$ ($\mathbf{B}$ and $\mathbf{Z}$ are suitably chosen positive semidefinite matrices). Moreover, our algorithm provides the first and the second derivative of the function $\varepsilon \to Tr(\mathbf{Z}\mathbf{X}(\varepsilon))$ almost for free.

The optimal dampers' viscosity is derived as $\varepsilon_{opt} = \mathrm{argmin}\, Tr(\mathbf{Z}\mathbf{X}(\varepsilon))$. If r is small, our algorithm allows a sensibly more efficient optimization, than standard methods based on the Bartels–Stewart's Lyapunov solver.

Key words: Damped-vibration, Lyapunov equation, optimization of dampers' viscosities.

1 Introduction

Dangerous vibrations are a frequent practical problem. For example, in the design of a bridge, one must pay attention to resonances of the bridge with the wind induced oscillatory forces. For the majority of engineering applications, resonance and sustained oscillations are not desirable because they may result in structural damage. The way to reduce resonance is through damping.

In this paper we consider a particular type of the vibration system, for example a mechanical system. The simplest type of this system is the mass spring damper system described by

$$\begin{aligned} m\ddot{x}(t) + d\dot{x}(t) + kx(t) &= 0 \\ x(0) = x_0, \quad \dot{x}(0) &= \dot{x}_0\,, \end{aligned}$$

where $m, d, k > 0$ are the mass, damping and stiffness coefficient, respectively, and $x(t)$ is the displacement from the equilibrium position.

The generalization of the upper system is given by

$$\begin{aligned} M\ddot{x} + D\dot{x} + Kx &= 0\,, \\ x(0) = x_0, \quad \dot{x}(0) &= \dot{x}_0\,, \end{aligned} \tag{1.1}$$

where M, D, K (called mass, damping, stiffness matrix, respectively) are real, symmetric matrices of order n with M, K positive definite and $D = C_u + C$, where C_u is positive definite and presents the internal damping which is usually taken as 2–10 % of the critical damping (see pp. 26, 260 [11]), and C is positive semidefinite.

A very important question arises in considerations of such systems: *for the given mass and stiffness determine the available dampers' viscosities so as to insure an optimal evanescence.*

For such optimization (and also for a more general one which includes optimization of dampers' positions or damping in general) one can use different optimization criteria (see [10]).

One of the frequently used criteria is the so-called spectral abscissa criterion, which requires that a maximal real part of the eigenvalues λ_k be minimal, that is

$$sp := \max_k \mathrm{Re}\lambda_k = \min\,, \tag{1.2}$$

where λ_k are the complex eigenvalues of the system

$$\left(\lambda^2 M + \lambda D + K\right) x = 0\,, \tag{1.3}$$

obtained from (1.1), simply using the substitution $x(t) = e^{\lambda t}x$.

Another criterion, used in [14], is given by requirement of the minimization of the total energy of the system, that is

$$\int_0^\infty E(t)\,dt = \min \tag{1.4}$$

The advantage of this criterion are: (i) its obvious closeness to the total energy of the vibration and (ii) its smoothness as the function of the damping

parameters, which allows standard methods of minimization via gradient or Hessian. Note that this last property is not shared by the spectral penalty function (1.2). On the other hand, Veselić in [17] and [18] has been shown that the solution of the Lyapunov equation provides rigorous bounds to the energy decay of a vibrating system.

Since, criterion (1.4) depends on the initial condition, the simplest way to correct this is to take average of (1.4) over all initial states of the unit total energy and a given frequency range. It can be shown that this average is the trace of the solution of the corresponding Lyapunov equation.

A general algorithm for the optimization of damping does not exist. Available algorithms optimize only viscosities of dampers, not their positions. Currently, the two types of algorithms are in use. The first type are the Newton-type algorithms for higher-dimensional (constrained or unconstrained) problems which use some Lyapunov solvers, and the second type are the algorithms which explicitly calculate the trace of the solution of the corresponding Lyapunov equation.

One algorithm of the second type was presented in [16] for the case when $C_u = 0$ and the rank of the matrix C is one. Moreover, in [16] Veselić has given an efficient algorithm which calculates optimal ε, where $C = \varepsilon cc^*$, and the optimal viscosity is given by the closed formula (see (2.19)).

A certain generalization of the result from [16] has been considered in [13]. It comes out that the case without the internal damping ($C_u = 0$) with $C = \varepsilon cc^*$, where $r \equiv rank(C) > 1$, is much more complicated than the case with the internal damping, thus we will present a sort of generalization of result from [16], with $C_u \neq 0$ and $C = \varepsilon cc^*$, where $r \equiv rank(C) > 1$. Usually, we assume that $r = 2, 3, 4$, which is a common assumption for a high voltage power line cable which usually contains one or two dampers with one or two degrees of freedom. We present an algorithm which derives a formula for the trace of the solution of the corresponding Lyapunov equation as the function of the viscosity ε of dampers. The formula provides the first and the second derivative of the function $\varepsilon \to Tr(\mathbf{ZX}(\varepsilon))$ almost for free.

Our algorithm needs $O(r^3m^3)$ operations, where $m = 2n$ (dimension of the phase space), for calculating the basic quantities in our formula for the trace and $O(r^2m^2)$ operations for calculating the first and the second derivative. This means that if the degrees of freedom of the dampers $r \ll n$ ($r = 2, 3$), our algorithm allows a sensibly more efficient optimization $\varepsilon_{opt} = \operatorname{argmin} Tr(\mathbf{ZX}(\varepsilon))$ ($O(r^3m^3) + n_{iter} \cdot O(r^2m^2)$ operations, where n_{iter} is a number of iterations) then the standard methods based on the Bartels–Stewart's Lyapunov solver ($\sim 30m^3r^2$ per iteration).

We will use the following notation, matrices written in the simple Roman

fonts, M, D or K for example will have $O(n^2)$ entries. Matrices written in the mathematical bold fonts, $\mathbf{A}$, $\mathbf{B}$ will have $O(m^2)$ entries, where $m = 2n$. And finally, matrices written in the Blackboard bold fonts $\mathbb{A}$, or $\mathbb{D}$ will have more than $O(m^3)$ entries.

The paper is organized as follows. The section 2 describes the mathematical model we will use. The section 3 contains the main result, which describes the way how we have derived the trace of the corresponding Lyapunov equation. In the section 4 we present our algorithms for calculating the trace of the Lyapunov equation. In the last section we compare our algorithm with the Newton-type algorithm from [2] based on the use of the Barthels-Stewart's Lyapunov solver, and we present an example which shows that Newton-type algorithms sometimes can fail, that is minimal energy obtained by our algorithm is 30–50 % better than minimal energy obtained by Newton-type algorithms.

2 Mathematical model

We consider a damped linear vibrational system described by the differential equation

$$M\ddot{x} + D\dot{x} + Kx = 0 \tag{2.5}$$

where M, C, K (called mass, damping, stiffness matrix, respectively) are real, symmetric matrices of order n with M, K positive definite and $D = C_u + C$ positive semidefinite, where C_u describes the internal damping. Often the matrix C has a small rank. An example is the so-called *n-mass oscillator* or *oscillator ladder* (Fig. 1) where

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

$$K = \begin{bmatrix} k_0 + k_1 & -k_1 & & & & \\ -k_1 & k_1 + k_2 & -k_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & -k_{n-2} & k_{n-2} + k_{n-1} & -k_{n-1} & \\ & & & -k_{n-1} & k_{n-1} + k_n \end{bmatrix},$$

$$D \equiv C_u + C = C_u + \varepsilon e_1 e_1^T + \varepsilon e_3 e_3^T. \tag{2.6}$$

Here $m_i > 0$ are the masses, $k_i > 0$ the spring constants or stiffnesses, e_i is the i-th canonical basis vector, and ε is the viscosity of the damper applied

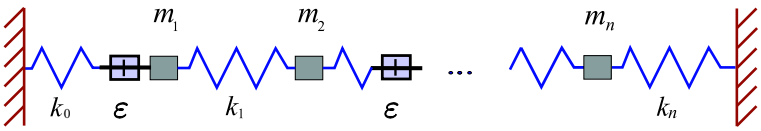

Fig. 1. The n-mass oscillator with two dampers

on the i-th mass. Note that all dampers have the same viscosity and that the rank of the matrix C is two. In this paper we study the system with r equal dampers where we assume that $r \ll n$ (usually $r = 2, 3, 4$).

To (2.5) corresponds the eigenvalue problem

$$(\lambda^2 M + \lambda D + K)x = 0\,. \tag{2.7}$$

Obviously all eigenvalues of (2.7) lie in the left complex plane (see [12, 3.8.1]).

Note that if we use Cholesky decompositions $M = L_M L_M^T$, $D = L_D L_D^T$ and $K = L_K L_K^T$, then (2.7) can be written in equivalent form

$$(\lambda^2 I + \lambda D_E D_E^T + K_E K_E^T)x_E = 0\,,$$

where $D_E = L_M^{-1} L_D$, $K_E = L_M^{-1} L_K$ and $x_E = L_M^T x$.

Now, using the singular value decomposition

$$K_E \equiv L_M^{-1} L_K = \Phi \Omega \Phi_1^T\,, \quad \Phi^T \Phi = I \text{ and } \Phi_1^T \Phi_1 = I$$

where $\Omega = \mathrm{diag}(\omega_1, \ldots, \omega_n)$, $\omega_1 < \ldots < \omega_n$ and setting

$$y_1 = \Omega\, \Phi^T x_E \quad y_2 = \Phi^T \dot{x}_E\,, \tag{2.8}$$

(2.5) can be written as

$$\dot{\mathbf{y}} = \mathbf{A}\mathbf{y}\,, \tag{2.9}$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}\,, \quad \mathbf{A} = \begin{bmatrix} 0 & \Omega \\ -\Omega & -\Phi^T D_E D_E \Phi \end{bmatrix}\,, \tag{2.10}$$

(we are now in the $2n$-dimensional phase space), with the solution

$$\mathbf{y} = e^{\mathbf{A}t}\,\mathbf{y}_0\,, \quad \text{where } \mathbf{y}_0 \text{ is the initial data.} \tag{2.11}$$

Note that, the numbers

$$\omega_1, \omega_2, \ldots, \omega_n\,, \tag{2.12}$$

are the eigenvalues of the corresponding undamped system

$$(\lambda^2 M + K)x = 0\,,$$

and we call them eigenfrequencies of the system.

Also note that the eigenvalue problem $\mathbf{A}\mathbf{y} = \lambda\,\mathbf{y}$ is equivalent to (2.7).

Now, (1.4) can be written as

$$\mathbf{y}_0^T \mathbf{X} \mathbf{y}_0 = \min\,, \tag{2.13}$$

where

$$\mathbf{X} = \int_0^\infty e^{\mathbf{A}^T t}\, e^{\mathbf{A} t} dt \tag{2.14}$$

is the solution of the Lyapunov equation

$$\mathbf{A}^T \mathbf{X} + \mathbf{X}\mathbf{A} = -\mathbf{I}\,. \tag{2.15}$$

The inconvenience in criterion (2.13) is dependence on the initial data $\mathbf{y}_0$. Thus, as in [14] instead of the quantity $\mathbf{y}_0^T \mathbf{X} \mathbf{y}_0$ we are going to take its mean value over all initial data $\mathbf{y}$ with the unit energy $\|\mathbf{y}\|^2$. Therefore, instead of (2.13) we require

$$\int_{\|\mathbf{y}_0\|=1} \mathbf{y}_0^T \mathbf{X} \mathbf{y}_0 \; d\sigma = \min \tag{2.16}$$

where $d\sigma$ is a chosen probability measure on the unit sphere $S^{2n} = \{\mathbf{y}_0 \in \mathbb{R}^{2n}; \|\mathbf{y}_0\| = 1\}$. So, we minimize the average total energy over the set of the initial conditions.

Since by the map

$$\mathbf{X} \mapsto \int_{\|\mathbf{y}_0\|=1} \mathbf{y}_0^T \mathbf{X} \mathbf{y}_0 \; d\sigma$$

is given a linear functional on the space of the symmetric matrices, by Riesz theorem there exists a symmetric matrix $\mathbf{Z}$ such that

$$\mathbf{X} \mapsto \int_{\|\mathbf{y}_0\|=1} \mathbf{y}_0^T \mathbf{X} \mathbf{y}_0 \; d\sigma = Tr(\mathbf{Z}\mathbf{X})\,, \text{ for all symmetric matrices } \mathbf{X}\,.$$

Let $\mathbf{y} \in \mathbb{R}^{2n}$ be arbitrary. Set $X = \mathbf{y}\mathbf{y}^T$. Then

$$0 \le \int_{\|\mathbf{y}_0\|=1} \mathbf{y}_0^T \mathbf{X} \mathbf{y}_0 \; d\sigma = Tr(\mathbf{Z}\mathbf{X}) = Tr(\mathbf{Z}\mathbf{y}\mathbf{y}^T) = Tr(\mathbf{y}^T \mathbf{Z} \mathbf{y})\,,$$

hence $\mathbf{Z}$ is always positive semi-definite.

For the measure σ generated by the Lebesgue measure (i.e. the usual surface measure) on $\mathbb{R}^{2n}$, we obtain $\mathbf{Z} = \frac{1}{2n}\mathbf{I}$. For the convenience of the reader, we give a sketch of the proof:

Recall,

$$\mathbf{Z}_{ij} = \int_S \mathbf{y}_i \mathbf{y}_j \, \sigma(d\mathbf{y})\,.$$

One can easily see using Minkowski formula (see [4]) that

$$\mathbf{Z}_{ij} = \int\limits_S \mathbf{y}_i \mathbf{y}_j \, \sigma(d\mathbf{y}) = \frac{1}{2\varepsilon} \lim_{\varepsilon \to 0} \int\limits_{d(\mathbf{y}, S) \leq \varepsilon} \mathbf{y}_i \mathbf{y}_j \, \sigma(d\mathbf{y}) \, .$$

Obviously, $\mathbf{Z}_{ij} = 0$ for $i \neq j$ and $\mathbf{Z}_{ii} = \mathbf{Z}_{jj}$, for $i, j \in \{1, 2, \ldots, 2n\}$. Since

$$\mathbf{Z}_{11} + \mathbf{Z}_{11} + \ldots \mathbf{Z}_{2n2n} = \lim_{\varepsilon \to 0} \frac{1}{2\varepsilon} \, vol \left(\mathbf{y} \in \mathbb{R}^{2n} : d(\mathbf{y}, S) \leq \varepsilon \right) = 1 \, ,$$

it follows $\mathbf{Z} = \frac{1}{2n}\mathbf{I}$.

We have shown that (2.16) is equivalent to

$$Tr(\mathbf{ZX}) = \min \, . \tag{2.17}$$

where $\mathbf{Z}$ is a symmetric positive semidefinite matrix which may be normalized to have a unit trace.

If one is interested in damping a certain part of the spectrum of the matrix $\mathbf{A}$ (which is very important in applications) then the matrix $\mathbf{Z}$ will have a special structure. For example, let $\sigma = \sigma_1 \times \sigma_2 \times \sigma_1 \times \sigma_2$, where σ_1 is a measure on the frequency subspace determined by $\omega \leq \omega_{\max} \equiv \omega_s$ generated by Lebesgue measure, that is σ_1 is a measure on the frequency subspace which corresponds to the eigenfrequencies (defined by (2.12)) $\omega_1, \ldots \omega_s$ and σ_2 is Dirac measure on the complement. Then we obtain that the corresponding matrix Z has the form

$$\mathbf{Z}_s = \mathbf{Z} = \frac{1}{2s} \begin{bmatrix} I_s & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & I_s & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \, , \tag{2.18}$$

where I_s is the identity matrix of the dimension s which is defined by $\omega_{\max} = \omega_s$. Here $\omega_{\max} = \omega_s$ is critical frequency with the property that the eigenfrequencies from (2.12) greater then ω_s are not dangerous. Hence, we damp first s eigenfrequencies. The construction of $\mathbf{Z}$ from (2.18) is similar to the previous construction of $\mathbf{Z} = \frac{1}{2n}\mathbf{I}$.

In [16] a solution of the problem (2.17) has been given in the case when $C_u = 0$ and $\mathrm{rank}(C) = 1$. In particular,

$$Tr(\mathbf{ZX}) = const + \frac{a}{\varepsilon} + b\varepsilon, \quad a, b > 0 \, , \tag{2.19}$$

which made it possible to find the minimum explicitly by a simple formula. The case $\mathrm{rank}(C) > 1$ seems to be essentially more difficult to handle.

Our approach here is based on the construction of the formula for the trace $Tr(\mathbf{ZX}(\varepsilon))$, and then minimization $\varepsilon_{opt} = \operatorname{argmin} Tr(\mathbf{ZX}(\varepsilon))$ using this formula.

The most expensive part in the calculation of the quantities in our formula is one Hesseberg reduction of a $2rm \times 2rm$ dimensional matrix, which costs $14/3\,(2rm)^3$ operations (see [3]).

Since our algorithm (as it will be shown in the next section) provides the first and the second derivative of the function $\varepsilon \rightarrowtail Tr(\mathbf{ZX}(\varepsilon))$ almost for free ($O(r^2m^2)$ operations) it follows that the whole optimization process costs $14/3\,(2rm)^3 + O(r^2m^2)$ operations.

On the other hand, algorithms of the Newton type based on the Bartels-Stewart solver have to solve $1 + \frac{3r + r^2}{2}$ different Lyapunov equations per iteration. Here 1 stands for deriving the solution $\mathbf{X}$, r for deriving the gradient and $\frac{r + r^2}{2}$ for deriving Hessian (see [2]). This means that these algorithms need $30(1 + \frac{3r + r^2}{2})$ operations per iteration, and if we have a good starting point we need approximately 10 iterations for the Newton process. This shows that our algorithm needs a smaller number of iterations for $r \leq 4$.

Further, one can use one of the methods which do not need any derivative of the function (for example The Golden Section Search for the minimization of a function of one variable or the Nelder-Mead Simplex Method for the minimization of a function of several variables). But then the number of iterations is bigger. For example, the standard MATLAB function `fminbnd` (which is based on the Golden Section search and parabolic interpolation), with the termination tolerance on ε equal to 10^{-8}, needs about 25 iterations (we have to solve the Lyapunov equation 25 times). This means that our algorithm still needs less operations for $r = 2$ and at the same time we obtaine much more accurate solution.

3 The main result

As we have said in the Introduction, our aim is to derive the trace $tr(\mathbf{Z}\,\mathbf{X})$ where $\mathbf{X}$ is the solution of the Lyapunov equation

$$\mathbf{A}^T\mathbf{X} + \mathbf{X}\mathbf{A} = -\mathbf{B}, \tag{3.1}$$

where $\mathbf{Z}$ is defined by (2.18) and $\mathbf{B}$ is symmetric positive semidefinite.

We will assume that the internal damping is between 2–10 % of the critical damping, that is, $C_u = \alpha\Phi\Omega\Phi^T$ where $0.02 \leq \alpha \leq 0.1$ (see [11]), then from

(2.10) it follows that $\mathbf{A}$ can be written as

$$\mathbf{A} \equiv \mathbf{A}_0 - \varepsilon\, \mathbf{D} = \begin{bmatrix} 0 & \Omega \\ -\Omega & -\alpha\, \Omega \end{bmatrix} - \varepsilon \begin{bmatrix} 0 & 0 \\ 0 & C_0 C_0^T \end{bmatrix} . \tag{3.2}$$

where

$$\mathbf{D} = \mathbf{D}_0 \mathbf{D}_0^T, \quad \mathbf{D}_0 = \begin{bmatrix} 0 \\ C_0 \end{bmatrix} \text{ and } C_0 = \Phi^T \left[e_{i_1}, \ldots, e_{i_k} \right] , \tag{3.3}$$

where e_{i_k} is the i_k-th canonical basis vector and r is the number of dampers. We assume that $\Omega = \mathrm{diag}(\omega_1, \ldots, \omega_n)$, where $\omega_1 < \ldots < \omega_n$.

Now, we proceed with solving the equation (3.1). As it is well known, Lyapunov equation (3.1) is equivalent to ([8, Theorem 12.3.1])

$$\Big(\mathbf{I} \otimes (\mathbf{A}_0 - \varepsilon\, \mathbf{D})^T + (\mathbf{A}_0 - \varepsilon\, \mathbf{D})^T \otimes \mathbf{I} \Big) \cdot \mathrm{vec}(\mathbf{X}) = -\mathrm{vec}(\mathbf{B}) , \tag{3.4}$$

where $\mathbf{L} \otimes \mathbf{T}$ denotes the Kronecker product of $\mathbf{L}$ and $\mathbf{T}$, and $\mathrm{vec}(\mathbf{B})$ is the vector formed by "stacking" the columns of $\mathbf{B}$ into one long vector.

Further, we will need the following two $m^2 \times m^2$ matrices defined by

$$\mathbb{A}_0 = \mathbf{I} \otimes \mathbf{A}_0^T + \mathbf{A}_0^T \otimes \mathbf{I} , \quad \mathbb{D} = \mathbf{I} \otimes \mathbf{D}_0 \mathbf{D}_0^T + \mathbf{D}_0 \mathbf{D}_0^T \otimes \mathbf{I} . \tag{3.5}$$

It is easy to show that $\mathbb{D} = \mathbb{D}_F\, \mathbb{D}_F^T$, where

$$\mathbb{D}_F = \Big[\mathbf{I} \otimes \mathbf{D}_0 \;\; \mathbf{D}_0 \otimes \mathbf{I} \Big] . \tag{3.6}$$

Indeed, using [8, Proposition 12.1.2]

$$\begin{aligned} \mathbb{D}_F \mathbb{D}_F^T &= (I_m \otimes \mathbf{D}_0)\, (I_m \otimes \mathbf{D}_0^T) + (\mathbf{D}_0 \otimes I_m)\, (\mathbf{D}_0^T \otimes I_m) \\ &= I_m \otimes \mathbf{D}_0\, \mathbf{D}_0^T + \mathbf{D}_0\, \mathbf{D}_0^T \otimes I_m . \end{aligned}$$

Note that the factor $\mathbb{D}_F$ is an $m^2 \times 2rm$ matrix. Now, using (3.6) and (3.5) it follows that equation (3.4) is equivalent to

$$\Big(\mathbb{A}_0 - \varepsilon \mathbb{D}_F \mathbb{D}_F^T \Big) \cdot \mathrm{vec}(\mathbf{X}) = -\mathrm{vec}(\mathbf{B}) . \tag{3.7}$$

Note that (2.17) means that we need to find a minimum for the function $f(\varepsilon) = tr(\mathbf{Z}\mathbf{X}(\varepsilon))$, where

$$tr(\mathbf{Z}\mathbf{X}(\varepsilon)) = \mathrm{vec}(\mathbf{Z})^T \mathrm{vec}(\mathbf{X}) = -\mathrm{vec}(\mathbf{Z})^T \Big(\mathbb{A}_0 - \varepsilon \mathbb{D}_F \mathbb{D}_F^T \Big)^{-1} \mathrm{vec}(\mathbf{B}) . \tag{3.8}$$

Using the *Sherman-Morrison-Woodbury formula* ([7, (2.1.4),p.51.]), the inverse matrix from (3.8) can be written as

$$(\mathbb{A}_0 - \varepsilon\, \mathbb{D}_F \mathbb{D}_F^T)^{-1} = \mathbb{A}_0^{-1} + \varepsilon \mathbb{A}_0^{-1} \mathbb{D}_F \left(I - \varepsilon \mathbb{D}_F^T \mathbb{A}_0^{-1} \mathbb{D}_F\right)^{-1} \mathbb{D}_F^T \mathbb{A}_0^{-1}\,. \tag{3.9}$$

Note that $\mathbb{A}_0$ is non-singular (for details see [13], [15], [16], [10]). It is well known that sometimes the *Sherman-Morrison-Woodbury formula* runs into numerical difficulties. A nice example about it can be found in [5, Ex. 1, pp. 249-250]. Fortunately, in meany applications, such a "patalogical" example is less natural (for example, if $n = 200$ and $\mathbb{A}_0$ corresponds to mechanical system from the Figure 1, then $\mathrm{cond}(\mathbb{A}_0) \sim 10^5$).

For simplification we introduce the following quantities:

$$x_0 = \mathrm{vec}(\mathbf{Z})^T\, \mathbb{A}_0^{-1}\, \mathrm{vec}(\mathbf{B}) \in \mathbb{R}\,, \tag{3.10}$$
$$a_L = \mathbb{D}_F^T\, \mathbb{A}_0^{-T}\, \mathrm{vec}(\mathbf{Z}) \in \mathbb{R}^{2rm}\,, \tag{3.11}$$
$$a_R = \mathbb{D}_F^T\, \mathbb{A}_0^{-1}\, \mathrm{vec}(\mathbf{B}) \in \mathbb{R}^{2rm}\,, \tag{3.12}$$
$$\mathbf{\Delta} = \mathbb{D}_F^T \mathbb{A}_0^{-1} \mathbb{D}_F \in \mathbb{R}^{2rm \times 2rm}\,. \tag{3.13}$$

Using (3.9)–(3.13) equation (3.8) can be written as

$$tr(\mathbf{Z}\mathbf{X}(\varepsilon)) = -x_0 - \varepsilon a_L^T (I - \varepsilon \mathbf{\Delta})^{-1} a_R\,. \tag{3.14}$$

It follows that we need an efficient algorithm which will calculate the quantities x_0, a_L, a_R and $\mathbf{\Delta}$ from (3.10)–(3.13), respectively, with minimal number of operations as it is possible.

Note that all quantities defined in (3.10)–(3.13) contains the inverse matrix $\mathbb{A}_0^{-1}$, which can not be derived directly. Thus, the main part of our algorithm contains a routine which derives a vector $\widehat{x} = \mathbb{A}_0^{-1}\widehat{y}$, for a given vector $\widehat{y}$.

The following section contains a mathematical background for the basic steps in our algorithm.

4 Description of the algorithm

As we have mentioned in the last section we need to construct an algorithm for deriving the vector $\widehat{x} = \mathbb{A}_0^{-1}\widehat{y}$, where $x, y \in \mathbb{R}^{m^2}$, and $\mathbb{A}_0$ is defined in (3.5). In fact $\widehat{x}$ is the solution of a linear system

$$\mathbb{A}_0 \widehat{x} = \widehat{y}\,, \tag{4.1}$$

which is equivalent to the Lyapunov equation

$$\mathbf{A}_0^T\widehat{\mathbf{X}} + \widehat{\mathbf{X}}\mathbf{A}_0 = \widehat{\mathbf{Y}}\,, \tag{4.2}$$

where $\widehat{x} = \mathrm{vec}(\widehat{\mathbf{X}})$, $\widehat{y} = \mathrm{vec}(\widehat{\mathbf{Y}})$. We will derive a general solution, that is we will not assume any structure on $\widehat{\mathbf{Y}}$, because we will need this kind of the solution in the construction of the matrix Δ.

Since $\mathbf{A}_0$ has a special structure, we can solve equation (4.2) directly with $O(m^2)$ operations, that is, we will not need to use the standard Lyapunov solvers which need $O(m^3)$ operations (like Bartels–Stewart for example).

Let

$$\widehat{\mathbf{X}} = \begin{bmatrix} \widehat{X}_{11} & \widehat{X}_{12} \\ \widehat{X}_{21} & \widehat{X}_{22} \end{bmatrix} \qquad \widehat{\mathbf{Y}} = \begin{bmatrix} \widehat{Y}_{11} & \widehat{Y}_{12} \\ \widehat{Y}_{21} & \widehat{Y}_{22} \end{bmatrix}$$

be the solution and the right-hand side of (4.2), respectively, where all blocks have the same dimension n. Now using this block representations, (4.2) is equivalent to

$$\begin{aligned} -\Omega\widehat{X}_{21} - \widehat{X}_{12}\Omega &= \widehat{Y}_{11} \\ (\widehat{X}_{11} - \alpha\widehat{X}_{12})\Omega - \Omega\widehat{X}_{22} &= \widehat{Y}_{12} \\ \Omega(\widehat{X}_{11} - \alpha\widehat{X}_{21}) - \widehat{X}_{22}\Omega &= \widehat{Y}_{21} \\ \Omega(\widehat{X}_{12} - \alpha\widehat{X}_{22}) + (\widehat{X}_{21} - \alpha\widehat{X}_{22})\Omega &= \widehat{Y}_{22}\,. \end{aligned} \tag{4.3}$$

If we denote

$$\widehat{X}_{11} = (\eta_{ij})\,, \quad \widehat{X}_{12} = (\xi_{ij})\,, \quad \widehat{X}_{21} = (\zeta_{ij})\,, \quad \widehat{X}_{22} = (\mu_{ij})\,,$$

then (4.3) can be written as

$$\begin{aligned} -\omega_j\xi_{ij} \quad -\omega_i\zeta_{ij} \qquad\qquad\quad &= (\widehat{Y}_{11})_{ij} \\ \omega_j\eta_{ij} \; -\alpha\omega_j\xi_{ij} \qquad -\omega_i\mu_{ij} \quad &= (\widehat{Y}_{12})_{ij} \\ \omega_i\eta_{ij} \qquad -\alpha\omega_i\zeta_{ij} \quad -\omega_j\mu_{ij} \quad &= (\widehat{Y}_{21})_{ij} \\ \omega_i\xi_{ij} \quad +\omega_j\zeta_{ij} \; -\alpha(\omega_i+\omega_j)\mu_{ij} &= (\widehat{Y}_{22})_{ij}\,. \end{aligned} \tag{4.4}$$

The solutions of system (4.4) are

$$
\begin{aligned}
\mu_{ij} &= \frac{(\widehat{Y}_{12})_{ij}\,\omega_i(\omega_j-\omega_i) - \omega_j\left(((\widehat{Y}_{11})_{ij} + (\widehat{Y}_{22})_{ij})\,\alpha\,\omega_i + (\widehat{Y}_{21})_{ij}\,(\omega_j-\omega_i)\right)}{(\omega_i+\omega_j)\left(\omega_i^2 + (\alpha^2-2)\,\omega_i\omega_j + \omega_j^2\right)} \\
\eta_{ij} &= \frac{(\widehat{Y}_{12})_{ij} + (\widehat{Y}_{21})_{ij} - \alpha\,(\widehat{Y}_{11})_{ij}}{\omega_i+\omega_j} + \mu_{ij} \\
\xi_{ij} &= \frac{\omega_j\,\eta_{ij} - \omega_i\mu_{ij} - (\widehat{Y}_{12})_{ij}}{\alpha\,\omega_j} \\
\zeta_{ij} &= \frac{-(\widehat{Y}_{11})_{ij} - \omega_j\,\xi_{ij}}{\omega_i}\,.
\end{aligned}
\tag{4.5}
$$

Thus, using (4.5) it is easy to construct an algorithm which calculates the vector $\widehat{x} = \mathbb{A}_0^{-1}\widehat{y}$, for any given vector $\widehat{y}$.

Note that for $\widehat{\mathbf{X}}(i,j)$, we need 32 operations which means that the whole solution $\widehat{\mathbf{X}}$ is obtained by $O(m^2)$ operations.

Now, we can proceed with calculating quantities x_0, a_L, a_R and $\mathbf{\Delta}$ from (3.10)–(3.13), respectively. First, note that from (3.10) it follows that for x_0 we need $O(m^2)$ operations (simple vector multiplication of the vectors $\mathrm{vec}(\mathbf{Z})$ and $\mathbb{A}_0^{-1}\,\mathrm{vec}(\mathbf{B})$).

We continue by deriving a_R from (3.12). If we write $\mathrm{vec}(\mathbf{X}_R) = \mathbb{A}_0^{-1}\mathrm{vec}(\mathbf{B})$, then using the equality ([7, p.180])

$$
\mathrm{vec}(CXB^T) = (B\otimes C)\mathrm{vec}(X)\,, \tag{4.6}
$$

from (3.12) it follows that

$$
a_R = \begin{bmatrix} I\otimes\mathbf{D}_0^T \\ \mathbf{D}_0^T\otimes I \end{bmatrix}\mathrm{vec}(\mathbf{X}_R) = \begin{bmatrix} \mathrm{vec}(\mathbf{D}_0^T\mathbf{X}_R) \\ \mathrm{vec}(\mathbf{X}_R\mathbf{D}_0) \end{bmatrix}\,.
$$

This means that for a_R we need additional $O(rm^2)$ operations.

The above described algorithm as the MATLAB function

```
function [x0, aL, aR] = solljapspec(omega, d0, alpha, B, Z)
```

is available from the author upon request.

From (3.11) it follows that the vector a_L can be obtained similarly. Note that $\mathrm{vec}(\mathbf{X}_L) = \mathbb{A}_0^{-T}\mathrm{vec}(Z)$, means that $\mathbf{X}_L$ is the solution of the equation $\mathbf{A}_0\mathbf{X}_L + \mathbf{X}_L\mathbf{A}_0^T = \mathbf{Z}$ which can be easily obtained using a slightly modified Algorithm 4.

Finally, we derive Δ from (3.13). Since matrices $\mathbb{D}_F$ and $\mathbb{A}_0^{-1}$ have $O(m^3)$ and $O(m^4)$ entries, respectively, we cannot multiply them directly, because we cannot put them in the computer memory. Thus, we need an algorithm which will derive the matrix Δ without constructing matrices $\mathbb{D}_F$ and $\mathbb{A}_0^{-1}$. For that purpose, let $\mathbf{\Theta}_i$, be an $m \times m$ matrix $i = 1, 2, \ldots, 2mr$, defined by

$$\mathrm{vec}(\mathbf{\Theta}_i) = \mathbb{D}_F(:, i), \quad i = 1, 2, \ldots, 2mr\,,$$

then

$$\mathbb{A}^{-1}\mathbb{D}_F = \Big[\mathrm{vec}(\mathbf{X}_1),\ \mathrm{vec}(\mathbf{X}_2),\ \ldots, \mathrm{vec}(\mathbf{X}_{2rm})\Big]\,,$$

where $\mathbf{X}_i$ is the i-th solution of the corresponding Lyapunov equation

$$\mathbf{A}_0^T\mathbf{X}_i + \mathbf{X}_i\mathbf{A}_0 = \mathbf{\Theta}_i,\ i = 1, 2, \ldots, 2mr\,. \tag{4.7}$$

Recall that $\mathbb{D}_F$ is defined in (3.6) as

$$\mathbb{D}_F = \Big[I \otimes \mathbf{D}_0\ \mathbf{D}_0 \otimes I\Big]\,,$$

which means that columns $\mathbb{D}_F(:, (i-1)\,r + 1), \ldots, \mathbb{D}_F(:, i \cdot r)$, $i = 1, \ldots, m$, contain n non-zero elements in rows $(2\,i - 1)\,n + 1, \ldots, 2\,i \cdot n$. On the other hand $\mathbb{D}_F(:, k)$, $k = r \cdot m + 1, \ldots, (r + 1)\,m$, contains n non-zero elements in rows $k + m, k + 2\,m, \ldots, k + n \cdot m$ and $\mathbb{D}_F(:, k)$, $k = (r + 1)\,m + 1, \ldots, 2\,r \cdot m$, contains n non-zero elements in rows $k, k + m, \ldots, k + (n - 1)\,m$.

This structure of the right-hand side in (4.7) allows us to calculate the solution $\mathbf{X}_i$ in $O(m)$ operations. Using this structure and (4.5) one can easily show that the solutions $\mathbf{X}_i$, for $i = 1, \ldots rm$, contain non-zero elements only in the k-th and the $n+k$-th column, where $k = \mathrm{rem}(\mathrm{floor}((i-1)/r), n)+1$. Here $\mathrm{rem}(x, y)$ and $\mathrm{floor}(a)$ are MATLAB functions, $\mathrm{rem}(x, y)$ is the remainder after division and $\mathrm{floor}(x)$ rounds the elements of x to the nearest integers. On the other hand, the solutions $\mathbf{X}_i$, for $i = rm+1, \ldots 2rm$, contain non-zero elements only in the k-th and the $n + k$-th row, where $k = \mathrm{rem}(i - 1, n) + 1$. This means that we can derive the matrix $\mathbb{A}^{-1}\mathbb{D}_F$ in $O(4rm^2)$ operations.

Using (4.6) it follows that the i-th column of the matrix Δ can be derived as

$$\mathbf{\Delta}(:, i) = \begin{bmatrix} \mathrm{vec}(\mathbf{D}_0^T\mathbf{X}_i) \\ \mathrm{vec}(\mathbf{X}_i\mathbf{D}_0) \end{bmatrix}\,,$$

which will cost us the additional $O(r^2m^2)$ operations. Here we assume that all matrix multiplications $\mathbf{D}_0^T\mathbf{X}_i$ are done considering the special structure of the above described matrices $\mathbf{X}_i$.

The above described algorithm as the MATLAB function

```
function [Delta ] = caldelt(omega, d0, alpha, Z, B, n, m, r)
```

is available from the author upon request.

Finally, for deriving the function $tr(\mathbf{ZX}(\varepsilon))$ from (3.14) we have to derive $a_L^T(I-\varepsilon\mathbf{\Delta})^{-1}a_R$. For this purpose we derive the upper Hessenberg form of the matrix Δ, that is

$$\Delta = \mathbf{U}_s\mathbf{H}_s\mathbf{U}_s^T$$

which together with (3.14) gives

$$tr(\mathbf{ZX}(\varepsilon)) = -x_0 - \varepsilon b_L^T(I-\varepsilon\mathbf{H}_s)^{-1}b_R\,, \tag{4.8}$$

where $b_R = \mathbf{U}_s^T a_R$ and $b_L = \mathbf{U}_s^T a_L$. This is the most expensive part of our Algorithm and costs us (see [3]) $\frac{112}{3}\,r^3m^3 + O(r^2m^2)$.

Now from (4.8) it follows that for any given $\varepsilon > 0$ a value of the function $tr(\mathbf{ZX}(\varepsilon))$ is obtained by solving the linear system $(I-\varepsilon\mathbf{H}_s)^{-1}b_R$ with the upper Hessenberg matrix which can be done in $O(r^2m^2)$ operations.

This means that we can evaluate the function $tr(\mathbf{ZX}(\varepsilon))$ at a certain number of points with additional $O(r^2m^2)$ operations.

Further, we will show how one can easily find the first and the second derivative of the function $f(\varepsilon) = tr(\mathbf{ZX}(\varepsilon))$ from (4.8). If we write

$$g_R(\varepsilon) = (I-\varepsilon\mathbf{H}_s)^{-1}b_R\,, \quad w_R(\varepsilon) = (I-\varepsilon\mathbf{H}_s)^{-1}(\mathbf{H}_s\,g_R(\varepsilon))\,, \tag{4.9}$$

the first and the second derivative, respectively, are given by

$$f'(\varepsilon) = -b_L^T\,g_R(\varepsilon) - \varepsilon b_L^T\,w_R(\varepsilon)\,, \tag{4.10}$$

$$f''(\varepsilon) = -2b_L^T w_R(\varepsilon) - 2\varepsilon b_L^T\,(I-\varepsilon\mathbf{H}_s)^{-1}\,(H_s\,w_R(\varepsilon))\,. \tag{4.11}$$

Note that vectors $g_R(\varepsilon)$, $w_R(\varepsilon)$ from (4.9) and the derivatives (4.10) and (4.11) are obtained by solving the linear systems with the same upper Hessenberg matrix $(I-\varepsilon\mathbf{H}_s)$ which can be done in $O(r^2m^2)$ operations.

For the optimization method we will use the Newton method for solving $f'(\varepsilon) = 0$, with the starting point close to zero which comes out as reasonable choice due to properties of the function $f(\varepsilon)$ (for details see [13]). Unfortunately, the choice of the starting point is still an open question, but for our problem it is not of great importance, because this part of our program is not expensive, so we can try two or three different starting points.

For example, one possible choice for the starting point is $\varepsilon_0 = (\sum_i^n \omega_i)/n$. This choice is connected with a global minimum for the optimization problem in the case when the damping matrix C_0 from (3.3) is non-singular ($r = n$, that

is, the number of dampers is equal to the dimension of the problem, for more details see [10], [2]).

5 Comparison

In this section we compare our algorithm with the one presented in [2]. As it has been said in the Introduction, in [2] has been presented the Newton-type algorithm based on the usage of the Lyapunov solvers (Bartels-Stewart) for a higher dimensional minimization problem.

As it was described in the last section, our algorithm costs $37.33r^3m^3 + O(r^2m^2)$ operations, while the standard routines of the Newton type based on Bartels-Stewart solver (see [2]) has to solve a $(3r + r^2)/2 + 1$ Lyapunov equation in one iteration, that is, $30((3r + r^2)/2 + 1)m^3$ operations without calculating the starting point (additional $O(m^3)$ operations). This means that for $r = 2$ with the assumption that the algorithm from [2] needs 7 iterations for convergence (which is true in most of the cases, but in general the number of iterations varies between $5 - 40$) our algorithm needs three times fewer operations, for $r = 3$ our algorithm needs one third operations fewer and for $r = 4$ the both algorithms have a similar number of operation (precisely our algorithm has 3.4 % more operations) .

In the case when $r > 4$ the matrix Δ defined in (3.13) is too big to handle which makes our algorithm inapplicable.

As the first example we consider the following family of the optimization problems: Let

$$\mathbf{A} \equiv \mathbf{A}_0 - \varepsilon\, \mathbf{D} = \begin{bmatrix} 0 & \Omega \\ -\Omega & -\alpha\,\Omega \end{bmatrix} - \varepsilon \begin{bmatrix} 0 & 0 \\ 0 & C_0 C_0^T \end{bmatrix},$$

be a matrix as in (3.2). We generate a different "damping matrices"

$$C_0 = \Phi^T \left[e_{i_1}, \ldots, e_{i_k} \right],$$

for a different choice of i_k, here Φ is a randomly chosen orthogonal matrix.

We have to optimize

$$Tr(\mathbf{Z}\mathbf{X}(\varepsilon)) = \min,$$

as a function of ε, where $\mathbf{Z}$ is a symmetric positive semidefinite matrix defined by (2.18) (we have used $s = 20$), and $\mathbf{X}$ is a solution of the Lyapunov equation

$$\mathbf{A}^T\mathbf{X} + \mathbf{X}\mathbf{A} = -\mathbf{I}.$$

Both algorithms are written in MATLAB, and they are not optimized for usage, but as an illustration we point out that in case with $r = 2$ our algorithm is faster between 2 and 30 times (this depends on the number of iterations needed in the algorithm from [2]), while for $r = 3$ our algorithm is faster between 1.2 and 8 times. For this class of problems we use the starting point $\varepsilon_0 = 1/(n \sum_i \omega_i)$.

We perform optimization of the trace $Tr(\mathbf{ZX}(\varepsilon))$, 50 times for $n = 100$ ($m = 200$) and 50 times for $n = 150$ ($m = 300$), with $r = 2$ and $r = 3$. In most of the cases our algorithm and the algorithm from [2] obtain a similar minimal value for the trace. Precisely, in 90 % of our experiments the algorithm from [2] obtains between 1 – 5 % smaller minimal traces, but in 10 % we obtain better minimal traces (between 1 – 30 %).

As the second example we consider the following family of the optimization problems: Let $M = 10 \cdot I_n$ be the mass matrix and let $K = 2I_n - diag(diag(I_{n-1})), 1) - diag(diag(I_{n-1})), -1)$ be the stiffness matrix, with $n = 100$. Further, let $D = C_u + \varepsilon(e_i e_i^T + e_j e_j^T)$, be a damping matrix with $i = 1 : 20$ and $j = i + 1 : 30$. This means that we optimize the trace (4.8) for 390 different positions with two dampers. The obtained results are similar to the results from the first optimizations. The trace obtained by our algorithm (here for the starting point we used $\varepsilon_0 = \sum_i \omega_i/n$) is between 1 – 3 % larger than the trace obtained by the algorithm from [2], while the number of operations needed for algorithm from [2] is between 5 and 10 times larger than the number of operations needed for our algorithm.

Finally, if we take diagonal matrix $M = diag(1, 3, 5, \ldots, m)$ with the same K and C_u, as in the above optimization problem, then for a certain position of dampers we obtain 30 – 50 % better minimal trace. But if we take viscosity obtained by our algorithm as the starting point for the algorithm from [2], then this algorithm attains 1 – 3 % better minimal trace. Taking all of this into consideration, we can draw a conclusion that our algorithm can be used for many different problems, such as optimization of dampers' viscosity for a small number of equal dampers, or deriving the starting point for the Newton-type algorithms based on the use of Lyapunov solvers.

Acknowledgements. I like to thank the referee for very careful reading of the manuscript and valuable comments. I am very grateful to Professor K. Veselić who introduced me to this problem, for his valuable comments and helpful suggestions. I would also like to thank I. Nakić for his careful reading of the manuscript and useful suggestions.

References

[1] R.H. Bartels and G.W. Stewart, A solution of the matrix equation AX + XB = C. *Comm. ACM* **15** (1972) 820–826.

[2] K. Brabender, Optimale Dämpfung von Linearen Schwingungssystemen. Ph. D. thesis Fernuniversität, Hagen, 1998.

[3] J.W. Demmel, *Applied Numerical Linear Algebra*. SIAM, Philadelphia, PA, 1997.

[4] Herbert Federer, *Geometric Measure Theory*. Springer-Verlag, New York Inc., New York, 1969.

[5] S. Fine and K. Scheinberg, Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, (2):243-264, 2001.

[6] S.K. Godunov, *Modern Aspects of Linear Algera*. AMS, Translations of mathematical monographs, v. 175.,Providence, Rhode Island, 1998.

[7] G.H. Golub and Ch.F. van Loan, *Matrix Computations*. J. Hopkins University Press, Baltimore, 1989.

[8] P. Lancaster and M. Tismenetsky, *The Theory of Matrices*. Academic Press, New York 1985.

[9] P.C. Müller and W.O. Schiehlen, *Lineare Schwingungen*. Akademische Verlagsgesellschaft Wiesbaden 1976.

[10] I. Nakić, Optimal damping of vibrational systems. Ph. D. thesis Fernuniversität, Hagen, 2002.

[11] Paz, M., *Structural dynamics, theory and computation*, Van Nostrand Reinhold, New York, 1991.

[12] F. Tisseur and K. Meerbergen, The quadratic eigenvalue problem. *SIAM Rev.*, 43(2):235-286, 2001.

[13] N. Truhar and K. Veselić, Optimizing the solution of the Ljapunov equation, technical report, Fernuniversität, Hagen, 2001.

[14] K. Veselić, K. Brabender and K. Delinić, Passive control of linear systems. *Applied Mathematics and Computation, M. Rogina et al. Eds. Dept. of Math. Univ. Zagreb*, 2001, 39-68.

[15] K. Veselić, On linear vibrational systems with one dimensional damping. *Appl. Anal.* **29** (1988) 1-18.

[16] K. Veselić, On linear vibrational systems with one dimensional damping II. *Integral Eq. Operator Th.* **13** (1990) 883-897.

[17] K. Veselić, Exponential decay of semigroups in Hilbert space. *Semigruop Forum.* **55** (1997) 325-331.

[18] K. Veselić, Estimating the operator exponential. *Linear Algebra Appl.*. **280** (1998) 241-244.