Low-Authority Controller Design via Convex Optimization

To Appear in AIAA J. Guidance, Control, and Dynamics

Arash Hassibi* Jonathan How† Stephen Boyd‡

Information Systems Laboratory
Stanford University
Stanford, CA 94305-9510

Email: `arash@isl.stanford.edu` `howjo@sun-valley.stanford.edu`
`boyd@isl.stanford.edu`

Telephone: (650) 723-9833 Fax: (650) 723-8473

May 20, 1999

*Ph.D. candidate, Dept. of Electrical Engineering, Stanford University. Research supported by Air Force (under F49620-97-1-0459)

†Assistant Professor, Dept. of Aeronautics and Astronautics, Member AIAA, IEEE, Associate Member ASME.

‡Professor, Dept. of Electrical Engineering, Stanford University. Research supported in part by AFOSR (under F49620-95-1-0318), NSF (under ECS-9222391 and EEC-9420565), and MURI (under F49620-95-1-0525).

Abstract

In this paper we address the problem of low-authority controller (LAC) design. The premise is that the actuators have limited authority, and hence cannot significantly shift the eigenvalues of the system. As a result, the closed-loop eigenvalues can be well approximated *analytically* using perturbation theory. These analytical approximations may suffice to predict the behavior of the closed-loop system in practical cases, and will provide at least a very strong rationale for the first step in the design iteration loop. We will show that LAC design can be cast as convex optimization problems that can be solved efficiently in practice using interior-point methods. Also, we will show that by optimizing the ℓ_1 norm of the feedback gains, we can arrive at sparse designs, *i.e.*, designs in which only a small number of the control gains are nonzero. Thus, in effect, we can also solve actuator/sensor placement or controller architecture design problems.

Keywords: Low-authority control, actuator/sensor placement, linear operator perturbation theory, convex optimization, second-order cone programming, semi-definite programming, linear matrix inequality.

1 Introduction

The premise in low-authority control (LAC) is that the actuators have limited authority, and hence cannot significantly shift the eigenvalues of the system [1, 2]. As a result, the closed-loop eigenvalues can be well approximated *analytically* using perturbation theory. These analytical approximations may suffice to predict the behavior of the closed-loop system in practical cases, and will provide at least a very strong rationale for the first step in the design iteration loop.

An important use of LAC is in lightly damped large structures with an infinite number of elastic modes, where LAC is used to provide a small amount of damping in a wide range of modes for maximum robustness. A high-authority controller (HAC) is then used around the LAC to achieve high damping or mode-shape adjustment in a selected number of modes to meet performance requirements.

In this paper we introduce a new method for low-authority controller design, based on convex programming. We formulate the LAC design problem as a nonlinear convex optimization problem, which can then be solved efficiently by interior-point methods. The advantage of formulating the problem as convex is that very large order problems can be solved (globally) in practice. Another advantage of this formulation is that it can handle a very wide variety of specifications and objectives beyond standard eigenvalue-placement. Typical design objectives for the LAC design include increased damping or decay rate for the system response, and typical constraints include limitations on the controller gains and actuator power. We show that by optimizing the ℓ_1 norm of the gains, we can arrive at sparse designs, *i.e.*, designs in which only a small number of the control gains are nonzero. Thus, in effect, we can also solve actuator/sensor placement or controller architecture design problems. Moreover, it is possible to address the *robustness* of the LAC, *i.e.*, closed-loop performance subject to uncertainties or variations in the plant model. Therefore, by combining all these, for example, we can solve the problem of robust actuator/sensor placement and LAC design in *one* step.

Although LAC design has been traditionally used for eigenvalue-placement, by using powerful Lyapunov methods it is possible to extend LAC design to specifications beyond eigenvalue-placement. These include bounds on output energy, quadratic costs on the state and control input, induced $\mathcal{L}_2$ gain, *etc.*

The paper is organized as follows. The next section poses the problem statement, which is followed by a section that presents typical applications of LAC. Section 4 is a brief overview of convex programming, and in particular, linear, second-order cone, and semi-definite programming. Section 5 discusses the first order perturbation formulas for the matrix eigenvalues, and how the design problem can be posed within convex optimization framework. Section 6 discusses the sparsity of the solution, which is important for the control architecture studies. Section 7 addresses robust LAC design, *i.e.*, a LAC design that guarantees performance subject to uncertainties or variations in the plant model. Section 8 introduces an extension to LAC design based on Lyapunov methods, and it is shown how additional performance objectives (other than eigenvalue-placement) can be included in the formulation. Finally, Sections 9 demonstrates the application of the methods on a few example problems.

2 Problem statement

We consider the linear time-invariant system

$$\dot{z} = A(x)z, \quad z(0) = z_0, \tag{1}$$

where $z(t) \in \mathbf{R}^n$ is the state, $x \in \mathbf{R}^q$ is a (design) parameter, and $A(x) \in \mathbf{R}^{n\times n}$ is differentiable at $x = 0$. The goal is to find x so that the system has sufficient damping, or more generally, the eigenvalues of the system are in some desired region of the complex plane. However, it is assumed that there is "limited authority" in designing x so that the eigenvalues of system (1) are only slightly different from the eigenvalues of the *unperturbed* system

$$\dot{z} = A(0)z, \quad z(0) = z_0, \tag{2}$$

i.e., system (1) with $x = 0$. Therefore, first order perturbation methods can be used to predict the eigenvalue locations of system (1) from the eigenvalue locations of system (2). We will refer to (1) and (2) as the *closed-loop* and *open-loop* systems respectively.

In many applications, it is desirable to achieve the required eigenvalue locations (or damping) when x has the minimum number of nonzero elements. In such cases, each nonzero x_i may correspond to a sensor, an actuator, a dissipating mechanism, or a structural component, and therefore, reducing the number of nonzero x_is simplifies the implementation. Hence, we will also address the problem of minimizing the number of nonzero elements of x such that the eigenvalues of system (1) are in some desired region of the complex plane.

In addition, we will consider robust LAC design, *i.e.*, a LAC design with guaranteed closed-loop system performance subject to uncertainties for variations in the system, as well as LAC design for performance measures beyond eigenvalue-placement.

3 Applications of LAC

A key control design methodology for flexible systems with many elastic modes follows the two-level architecture presented in [1, 3, 2, 4]. This architecture consists of a wide-band, low-authority control (LAC) and a narrow-band, high-authority control (HAC). Within this framework, the HAC is designed based on a (low-order) finite-dimensional model of the structure, and provides high damping or mode-shape adjustment in a selected number of modes to meet performance requirements. However, due to spillover, the HAC can destabilize modes not included in the design model, which are usually at high frequency and poorly known. LAC, on the other hand, introduces low damping in a wide range of modes for maximum robustness. LAC is, therefore, necessary to reduce the destabilization problems created by HAC. HAC, for example, could be a linear-quadratic-Gaussian (LQG) controller using a collection of sensors and actuators. LAC, however, is usually implemented using (active or passive) high-energy-dissipating mechanisms [5].

High-energy-dissipating mechanisms are usually incorporated into the structure by using layers of viscoelastic shear damping material. In the simplest case, the force-extension characteristic of viscoelastic material can be modeled as a combination of a linear spring and a dash-pot, where the stiffness and damping is related to the geometry of the dissipating mechanism and the amount of viscoelastic material used. Hence, in the framework (1) the parameter x represents, for example, the amount of viscoelastic material at various locations of the structure. A zero x_i would mean that the dissipating mechanism at the corresponding location is not needed, so in many cases it is desirable to find an x with as many zero components as possible (subject to the control design specifications) to obtain a simple design.

Linear state-feedback LAC design is another example that can be easily cast in the framework (1). We may require the state-feedback gain to satisfy certain constraints (*e.g.*, on the size of its components or its sparsity pattern), or find a state-feedback gain that is sparse (so that a small number of sensors/actuators are needed and the controller has a simple topology). This state-feedback approach is particularly useful for the (collocated) rate-feedback design often used for LAC. Specifically, suppose that

$$\dot{z} = Az + Bu, \quad u = Kz,$$

where $A \in \mathbf{R}^{n\times n}$ and $B \in \mathbf{R}^{n\times m}$ are given and $K \in \mathbf{R}^{m\times n}$ should be found to achieve, say, sufficient damping. The closed-loop system becomes $\dot{z} = (A + BK)z$ and by taking x to be the elements of K this problem falls into the framework (1). A sparse K represents a simple controller topology since sparsity implies that we only need to connect each sensor to a *few* actuators. Moreover, a zero row (column) in K means that the corresponding actuator (sensor) is not required.

More generally, we can also consider *dynamic* LAC design for the open-loop system

$$\dot{z} = Az + Bu, \quad y = Cz,$$

where the controller is parameterized by its state-space system matrices A_{c}, B_{c}, C_{c}, D_{c}, and is given by

$$\dot{z}_{\mathrm{c}} = A_{\mathrm{c}} z_{\mathrm{c}} + B_{\mathrm{c}} y, \quad u = C_{\mathrm{c}} z_{\mathrm{c}} + D_{\mathrm{c}} y.$$

The closed-loop system can be written as

$$\begin{bmatrix} \dot{z} \\ \dot{z}_{\mathrm{c}} \end{bmatrix} = \begin{bmatrix} A + BD_{\mathrm{c}}C & BC_{\mathrm{c}} \\ B_{\mathrm{c}}C & A_{\mathrm{c}} \end{bmatrix} \begin{bmatrix} z \\ z_{\mathrm{c}} \end{bmatrix},$$

which is in the form $\dot{\tilde{z}} = A(x)\tilde{z}$ where $\tilde{z} = [z^T \;\; z_{\mathrm{c}}^T]^T$, and x represents the elements of the controller system matrices A_{c}, B_{c}, C_{c}, and D_{c}. By requiring sparsity for B_{c}, C_{c}, and D_{c} we can find designs that require a small number of actuators and sensors.

Another problem that can be formulated in the LAC framework is that of structural design and optimization [6]. In such a case, x can include various parameters such as beam widths, beam lengths, masses, dampers, *etc.* The best design, for example, is a structure that supports specified loads at fixed points, achieves acceptable dynamic behavior such as sufficient damping, and at the same time, has the simplest topology or minimum weight.

4 Linear, second-order cone, and semi-definite programming

In this section we briefly introduce linear, second-order cone, and semi-definite programs which are families of convex optimization problems that can be efficiently solved (globally) using interior-point methods [7, 8]. In later sections, we will see how LAC design can be cast in terms of linear, second-order cone, or semi-definite programs and hence solved efficiently in practice.

A *linear program* (LP) is an optimization problem with linear objective and linear equality and inequality constraints:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & f_i^T x \leq g_i, \quad i = 1, \ldots, J, \\ & Ax = b, \end{array} \tag{3}$$

where the vector x is the optimization variable and c, f_i, g_i, A, and b are problem parameters. Linear programming has been used in a wide variety of fields. In control, for example, Zadeh and Whalen observed in 1962 that certain minimum-time and minimum-fuel optimal control problems could be (numerically) solved by linear programming [9]. In the late 70s, Richalet [10] developed *model predictive control* (also known as dynamic matrix control or receding horizon control), in which linear or quadratic programs are used to solve an optimal control problem at each time step. Model predictive control is now widely used in the process control industry. Several high quality, efficient implementations of interior-point LP solvers are available (see, *e.g.*, [11, 12, 13]).

A *semi-definite program* (SDP), is an optimization problem which has the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x_1 F_1 + \cdots + x_m F_m \preceq G, \\ & Ax = b \end{array} \tag{4}$$

where F_i and G are symmetric $p \times p$ matrices, and the inequality $\preceq$ denotes matrix inequality, *i.e.*, $X \preceq Y$ means $Y - X$ is positive semi-definite. The constraint $x_1 F_1 + \cdots + x_m F_m \preceq G$ is called a *linear matrix inequality* (LMI). While SDPs look complicated and would appear difficult to solve, new interior-point methods can solve them with great efficiency (see, *e.g.*, [8, 14]) and several SDP codes are now widely available [15, 16, 17, 18, 19, 20]. The ability to numerically solve SDPs with great efficiency is being applied in several fields, *e.g.*, combinatorial optimization and control [21]. SDP is currently a highly active research area.

A *second-order cone program* (SOCP) has the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & \|F_i x + g_i\| \leq c_i^T x + d_i, \quad i = 1, \ldots, L, \\ & Ax = b, \end{array} \tag{5}$$

where $\| \cdot \|$ denotes the Euclidean norm, *i.e.*, $\|z\| = \sqrt{z^T z}$. SOCPs include linear and quadratic programming as special cases, but can also be used to solve a variety of nonlinear, nondifferentiable problems; see, *e.g.*, [22]. Moreover, efficient interior-point software for SOCP is now available [23, 24].

As a final note, it should be mentioned that among the three different class of optimization problems mentioned, SDP is the most general, and includes LP and SOCP as special cases.

5 Eigenvalue-placement LAC design using linear and second-order cone programming

In this section we show that analytic first order perturbation formulas for eigenvalues of a matrix can be used to design low-authority controllers using linear or second-order cone programming for eigenvalue-placement specifications. As mentioned in §4, linear and second-order cone programs can be solved very efficiently, and therefore, this gives an efficient method for LAC design.

5.1 First order perturbation formulas for eigenvalues of a matrix

A typical problem of the perturbation theory for linear operators is to investigate how the eigenvalues of a linear operator $A \in \mathbf{R}^{n\times n}$ change when A is subjected to small perturbation. For example, consider the family of operators $A(x) \in \mathbf{R}^{n\times n}$ where $A(0) = A$ and $x \in \mathbf{R}^q$ is a parameter supposed to be small. A question arises whether the eigenvalues of $A(x)$ can be expressed as a power series in x, *i.e.*, whether they are holomorphic functions of x in the neighborhood of $x = 0$.

In [25] it is shown that if $A(x)$ is k-times continuously differentiable in x on a simply-connected domain $\mathcal{D} \subset \mathbf{R}^q$, and the number of eigenvalues $\lambda_i(x)$ of $A(x)$ corresponding to a Jordan block of size 1 is constant for $x \in \mathcal{D}$, then each $\lambda_i(x)$ is also k-times continuously differentiable. Therefore, the change of these eigenvalues will be of the same order of magnitude as the perturbation for small $\|x\|$. Specifically we have

$$\lambda_i(x) = \lambda_i + \sum_{k=1}^{q} \left(\frac{w_i^* A_k u_i}{w_i^* u_i} \right) x_k + o(\|x\|), \tag{6}$$

where $u_i \in \mathbf{C}^n$, $w_i \in \mathbf{C}^n$ are the left and right eigenvectors of $A(0)$ corresponding to the eigenvalue $\lambda_i \in \mathbf{C}$, and $A_k = \partial A(0)/\partial x_k$ for $k = 1, \ldots, q$. Equation (6) gives the first order expansion formula for the eigenvalues of the perturbed matrix $A(x)$.

Remark. If λ_i is a repeated eigenvalue of $A(0)$ corresponding to a Jordan block of size $p_i > 1$, $\lambda_i(x)$ is no longer given as in (6). In this case $\lambda_i(x)$ is given by a *Puiseux series* such as

$$\lambda_i(x) = \lambda_i + \sum_{k=1}^{q} \alpha_{ik} x_k^{1/p_i} + \sum_{k=1}^{q}\sum_{j=1}^{q} \beta_{ikj} x_k^{1/p_i} x_j^{1/p_i} + \cdots.$$

In other words, the change in the eigenvalue is not of the same order of magnitude as the perturbation of the matrix for small $\|x\|$.

5.2 LAC eigenvalue-placement design using linear or second-order cone programming

Let $\mathcal{D}_i \subset \mathbf{C}$ be the desired region for $\lambda_i(x)$, the ith eigenvalue of $A(x)$. We assume that $\mathcal{D}_i$ is either *polyhedral* (an intersection of J_i half-planes) given by

$$\mathcal{D}_i = \{\ s \in \mathbf{C} \mid a_{ij}\,\mathbf{Re}(s) + b_{ij}\,\mathbf{Im}(s) \le c_{ij},\ \ j = 1, \ldots, J_i\ \}, \tag{7}$$

where $a_{ij} \in \mathbf{R}$, $b_{ij} \in \mathbf{R}$, $c_{ij} \in \mathbf{R}$, or an intersection of *second-order cones* given by

$$\mathcal{D}_i = \{\ s \in \mathbf{C} \mid \left\| F_i \begin{bmatrix} \mathbf{Re}(s) \\ \mathbf{Im}(s) \end{bmatrix} + g_i \right\| \le c_i^T \begin{bmatrix} \mathbf{Re}(s) \\ \mathbf{Im}(s) \end{bmatrix} + d_i\ \}, \tag{8}$$

where $F_i \in \mathbf{R}^{2\times 2}$, $g_i \in \mathbf{R}^2$, $c_i \in \mathbf{R}^2$, $d_i \in \mathbf{R}$, in which $\mathbf{Re}(s)$ and $\mathbf{Im}(s)$ are the real and imaginary parts of $s \in \mathbf{C}$ respectively (examples of these regions will follow shortly).

Under the low-authority control assumption, we can drop the $o(\|x\|)$ term in equation (6) without significant error, and $\lambda_i(x)$ becomes approximately linear in the design variable x. From (6)

$$\begin{aligned} \mathbf{Re}\,(\lambda_i(x)) &\approx \mathbf{Re}\,(\lambda_i) + \textstyle\sum_{k=1}^q \mathbf{Re}\left(\frac{w_i^* A_k u_i}{w_i^* u_i}\right) x_k, \\ \mathbf{Im}\,(\lambda_i(x)) &\approx \mathbf{Im}\,(\lambda_i) + \textstyle\sum_{k=1}^q \mathbf{Im}\left(\frac{w_i^* A_k u_i}{w_i^* u_i}\right) x_k, \end{aligned}$$

and therefore to first order $\lambda_i(x) \in \mathcal{D}_i$ as defined in (7) if and only if for $j = 1, \ldots, J_i$

$$a_{ij}\left(\mathbf{Re}\,(\lambda_i) + \sum_{k=1}^q \mathbf{Re}\left(\frac{w_i^* A_k u_i}{w_i^* u_i}\right) x_k\right) + b_{ij}\left(\mathbf{Im}\,(\lambda_i) + \sum_{k=1}^q \mathbf{Im}\left(\frac{w_i^* A_k u_i}{w_i^* u_i}\right) x_k\right) \le c_{ij},$$

or equivalently

$$\sum_{k=1}^q \left(a_{ij}\,\mathbf{Re}\left(\frac{w_i^* A_k u_i}{w_i^* u_i}\right) + b_{ij}\,\mathbf{Im}\left(\frac{w_i^* A_k u_i}{w_i^* u_i}\right)\right) x_k \le c_{ij} - a_{ij}\,\mathbf{Re}\,(\lambda_i) - b_{ij}\,\mathbf{Im}\,(\lambda_i)\,, \tag{9}$$

which is a linear inequality constraint in the variable $x \in \mathbf{R}^q$.

Similarly, if we require that $\lambda_i(x)$ fall inside the second-order conic region $\mathcal{D}_i$ as in (8), to first order we must have

$$\begin{aligned} &\left\| F_i \begin{bmatrix} \mathbf{Re}\left(\frac{w_i^* A_1 u_i}{w_i^* u_i}\right) & \cdots & \mathbf{Re}\left(\frac{w_i^* A_q u_i}{w_i^* u_i}\right) \\ \mathbf{Im}\left(\frac{w_i^* A_1 u_i}{w_i^* u_i}\right) & \cdots & \mathbf{Im}\left(\frac{w_i^* A_q u_i}{w_i^* u_i}\right) \end{bmatrix} x + F_i \begin{bmatrix} \mathbf{Re}(\lambda_i) \\ \mathbf{Im}(\lambda_i) \end{bmatrix} + g_i \right\| \le \\ &\qquad c_i^T \begin{bmatrix} \mathbf{Re}\left(\frac{w_i^* A_1 u_i}{w_i^* u_i}\right) & \cdots & \mathbf{Re}\left(\frac{w_i^* A_q u_i}{w_i^* u_i}\right) \\ \mathbf{Im}\left(\frac{w_i^* A_1 u_i}{w_i^* u_i}\right) & \cdots & \mathbf{Im}\left(\frac{w_i^* A_q u_i}{w_i^* u_i}\right) \end{bmatrix} x + c_i^T \begin{bmatrix} \mathbf{Re}(\lambda_i) \\ \mathbf{Im}(\lambda_i) \end{bmatrix} + d_i, \end{aligned} \tag{10}$$

which is a second-order cone constraint in $x \in \mathbf{R}^q$.

Suitable objectives are usually ones that require x to be in some sense "small". These include different norms on x such as $\|x\|_1$, $\|x\|_2$, and $\|x\|_\infty$. For example, minimizing $\|x\|_1$ or $\|x\|_\infty$ subject to (9) leads to LPs (after adding slack variables), while minimizing any of these norms subject to (9) or (10) leads to SOCPs. Therefore, the LAC eigenvalue-placement problem can be easily cast as an LP or SOCP that can be solved very efficiently.

Consider a typical example which is to place the eigenvalues of system (1) in the shaded region of Figure 1(a) (damping of at least 0.1, damping ratio of at least 0.2), and the objective is to minimize the sum of the entries of x. In this case, for $i = 1, \ldots, n$

$$\mathbf{Re}\,(\lambda_i(x)) \leq -0.1, \quad \mathbf{Im}\,(\lambda_i(x)) \pm 5\,\mathbf{Re}\,(\lambda_i(x)) \leq 0.$$

Therefore, the optimization problem becomes (to first order)

$$\begin{array}{ll} \text{minimize} & x_1 + x_2 + \cdots + x_q \\ \text{subject to} & \sum_{k=1}^q \mathbf{Re}\left(\frac{w_i^* A_k u_i}{w_i^* u_i}\right) x_k \leq -0.1 - \mathbf{Re}\,(\lambda_i)\,, \\ & \sum_{k=1}^q \left(\mathbf{Im}\left(\frac{w_i^* A_k u_i}{w_i^* u_i}\right) \pm 5\,\mathbf{Re}\left(\frac{w_i^* A_k u_i}{w_i^* u_i}\right)\right) x_k \leq -\,\mathbf{Im}\,(\lambda_i) \mp 5\,\mathbf{Re}\,(\lambda_i)\,, \\ & i = 1, \ldots, n, \end{array}$$

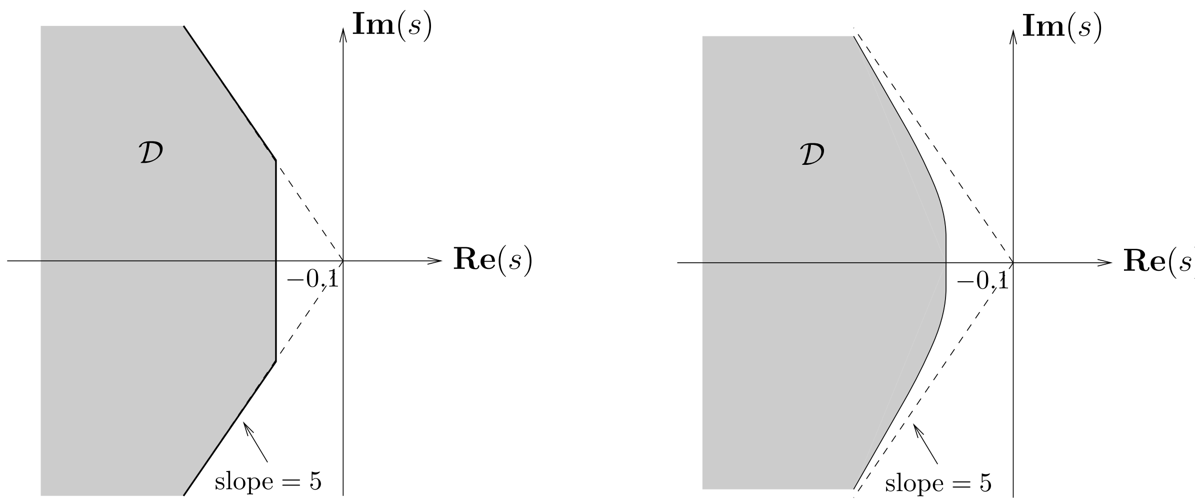
which is an LP in x. (Of course because of conjugate symmetry of the eigenvalues not all of the linear inequality constraints need to be imposed).

As another example, if the eigenvalues are to be placed in the hyperbolic region $\mathcal{D}$ of Figure 1(b), *i.e.*, $\{s \mid (\sqrt{\mathbf{Im}(s)^2} \leq -5\,\mathbf{Re}(s) - 0.5\}$, and the objective is the same as before, according to (10) we get the optimization problem

$$\begin{array}{ll} \text{minimize} & x_1 + x_2 + \cdots + x_q \\ \text{subject to} & \left\| \begin{bmatrix} \mathbf{Im}(\lambda_i) \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{Im}\left(\frac{w_i^* A_1 u_i}{w_i^* u_i}\right) & \cdots & \mathbf{Im}\left(\frac{w_i^* A_q u_i}{w_i^* u_i}\right) \\ 0 & \cdots & 0 \end{bmatrix} x \right\| \leq \\ & \\ & -5\begin{bmatrix} \mathbf{Re}\left(\frac{w_i^* A_1 u_i}{w_i^* u_i}\right) & \cdots & \mathbf{Re}\left(\frac{w_i^* A_q u_i}{w_i^* u_i}\right) \end{bmatrix} x - 5\,\mathbf{Re}(\lambda_i) - 0.5, \quad i = 1, \ldots, n, \end{array}$$

which is an SOCP in x.

Note that we can also mix the linear inequality and second-order cone constraints (9) and (10) with other constraints on x. For example, we may require that $0 \leq x_i \leq x_{i,\max}$ ($x_{i,\max}$ is given) corresponding to, say, physical limitations on the values of x_i. As long as these conditions are linear equality, linear inequality, or second-order cone constraints in x, they can be easily dealt within an efficient optimization program.



(a) Polyhedral region.

(b) Hyperbolic region.

Figure 1: Desired regions for system eigenvalues.

6 Sparse LAC design

In many cases it is desirable to guarantee performance for system (1) using the minimum number of nonzero elements of the vector x. For example, each nonzero element could correspond to a sensor, actuator, damper, or structural component, and a *sparse* x (*i.e.*, one with "many" zero elements) would result in a simpler controller, dissipation mechanism, or structure. As another example, x could denote the entries of a full matrix of feedback gains that indicates which sensors should be connected to which actuators. A sparse x then corresponds to a simpler controller topology. In this section we briefly address the problem of computing a sparse x that satisfies one or more of the constraints in §5 (or §8).

The problem of minimizing the number of nonzero elements of a vector x (subject to some constraints in x) arises in many different fields, but unfortunately, except in very special cases, it is a very difficult problem to solve numerically [26, 27, 28, 29]. However, a relaxation to this problem gives reasonably sparse solutions while being numerically tractable [26, 29]. The method is to minimize the ℓ_1 norm of x instead of minimizing its nonzero entries. The ℓ_1 norm of x is defined as $\|x\|_1 = |x_1| + \cdots + |x_q|$, and therefore, minimizing $\|x\|_1$ subject to, for example, (9) or (10) is an LP or SOCP that can be solved very efficiently.

To see why this method is a relaxation to our original problem, let $\|x\|_0$ be the number of nonzero elements of x. Now consider the following optimization problem:

$$\begin{array}{ll} \text{minimize} & \|x\|_0 \\ \text{subject to} & x \in \mathcal{C} \\ & \|x\|_\infty \leq 1, \end{array} \tag{11}$$

where $\mathcal{C}$ is some compact (convex) subset of $\mathbf{R}^n$. (Note that by scaling variables, without loss of generality, we can assume that $\|x\|_\infty \leq 1$.) The optimization problem (11) can be cast as the mixed optimization problem

$$\begin{array}{ll} \text{minimize} & z_1 + z_2 \cdots + z_q \\ \text{subject to} & x \in \mathcal{C} \\ & |x_i| \leq z_i \\ & z_i \in \{0, 1\}, \quad i = 1, \ldots, q. \end{array}$$

Now if we relax the Boolean constraint $z_i \in \{0, 1\}$ by $0 \leq z_i \leq 1$ we get the optimization problem

$$\begin{array}{ll} \text{minimize} & z_1 + z_2 \cdots + z_q \\ \text{subject to} & x \in \mathcal{C} \\ & |x_i| \leq z_i \\ & 0 \leq z_i \leq 1, \quad i = 1, \ldots, q. \end{array}$$

which is the same as

$$\begin{array}{ll} \text{minimize} & \|x\|_1 \\ \text{subject to} & x \in \mathcal{C} \\ & \|x\|_\infty \leq 1. \end{array} \tag{12}$$

Therefore, we have just shown that (12) is a natural relaxation to (11). (When we do not have the constraint $\|x\|_\infty \leq 1$, the natural relaxation will involve a *weighted* ℓ_1 norm.)

The ℓ_1 norm relaxation method usually tends to give acceptable sparse solutions. However, if we insist on finding the x with the least number of nonzero elements, we need to enumerate all possible sparsity patterns of x and check them for feasibility (*i.e.*, if there exists an x with the given sparsity pattern that satisfy the constraints). Among all feasible sparsity patterns of x, the one with the least number of nonzero elements minimizes $\|x\|_0$.

Since x has q components and each component is either zero or nonzero, the total number of sparsity patterns of x is 2^q. Therefore, in principle, by solving at most 2^q feasibility problems it is possible to find an x minimizing $\|x\|_0$. However, 2^q could be very large for even relatively small values of q, and as a result, finding the optimum x could be cumbersome. Good heuristics on to how and in what order to check the different sparsity patterns of x usually greatly reduce the necessary number of feasibility problems we need to solve. For example, one heuristic is to use the solution to the ℓ_1 relaxation problem as a basis to decide what sparsity patterns should be checked first. The idea is simply that it is "more probable" for the ith component of the optimum solution to be nonzero if the ith component of the solution to the relaxed problem is "large".

7 Robust LAC design

In this section we address the problem of robust LAC design, *i.e.*, a LAC design with guaranteed (closed-loop) system performance subject to *uncertainties* or *variations* in the system model. We show that it is possible to solve the robust LAC design problem using LP and SOCP. Therefore, by combining the methods of this section and that of §6, we can handle low-authority controller design, actuator/sensor placement, and robustness at the same time. Robust actuator/sensor placement and robust controller design are usually performed in two separate stages (and hence non-optimally) because it is numerically intractable to do otherwise (see, *e.g.*, [30] for a thorough overview of robust actuator and damper placement for structural control). It is yet another numerical advantage of LAC design that it is possible to handle both of these problems at one step very efficiently.

We will consider two different approaches for modeling the system uncertainty and will show how to design a robust LAC in each case. The first approach is to consider a parametric uncertainty, and the second approach is to model the uncertainty by a finite number of possible system models. The uncertainty is assumed to be time-invariant in both cases.

7.1 Robust LAC design for systems subject to "small" parametric uncertainties

As a generalization to the setup of §1, we assume that the dynamics of the (closed-loop) system can be described as

$$\dot{z} = A(x, \delta)z \tag{13}$$

where $x \in \mathbf{R}^q$ is the design parameter (as before), and $\delta \in \mathbf{R}^r$ represents the *model uncertainty* satisfying

$$-\delta_{i,\max} \le \delta_i \le \delta_{i,\max} \tag{14}$$

for $i = 1, \ldots, r$ in which $\delta_{i,\max}$ is given. We assume that the low-authority assumption holds and δ is "small" so that the eigenvalues of $A(x, \delta)$ can be well-approximated using (first order) perturbation formulas. The goal is to find x such that for all possible values of δ, the eigenvalues of (13) are in some desired region of the complex plane. Let $\mathcal{D}_i \subset \mathbf{C}$ be the desired region for $\lambda_i(x,\delta)$, the ith eigenvalue of $A(x,\delta)$. We assume that $\mathcal{D}_i$ is polyhedral as in (7).

Using the Farkas Lemma (see, *e.g.*, [6]) it can be shown that (to first order) $\lambda_i(x,\delta) \in \mathcal{D}_i$ for all δ satisfying (14) if and only if there exists $\tau^{(1)}, \tau^{(2)} \in \mathbf{R}^r$ such that

$$\begin{aligned} &\tau_l^{(1)} \geq 0, \quad \tau_l^{(2)} \geq 0, \quad \tau_l^{(1)} - \tau_l^{(2)} = a_{ij}\,\mathbf{Re}\left(\tfrac{w_i^* \bar{A}_l u_i}{w_i^* u_i}\right) + b_{ij}\,\mathbf{Im}\left(\tfrac{w_i^* \bar{A}_l u_i}{w_i^* u_i}\right), \\ &\sum_{k=1}^{q}\left(a_{ij}\,\mathbf{Re}\left(\frac{w_i^* A_k u_i}{w_i^* u_i}\right) + b_{ij}\,\mathbf{Im}\left(\frac{w_i^* A_k u_i}{w_i^* u_i}\right)\right) x_k + \\ &\qquad\qquad \sum_{l=1}^{r}\left(\tau_l^{(1)} + \tau_l^{(2)}\right)\delta_{l,\max} \leq c_{ij} - a_{ij}\,\mathbf{Re}\,(\lambda_i) - b_{ij}\,\mathbf{Im}\,(\lambda_i), \end{aligned} \tag{15}$$

for $l = 1, \ldots, r$ and $j = 1, \ldots, J_i$, which is a set of linear equality and inequality constraints in x, $\tau^{(1)}$, and $\tau^{(2)}$. Hence, by minimizing $\|x\|_1$ subject to (15) for example, it is possible to design robust and sparse LACs for eigenvalue-placement specifications subject to bounded parametric uncertainties in the system model by solving LPs.

Note that using similar methods, it is possible to cast robust LAC design as an LP or SOCP for cases in which $\mathcal{D}_i$ is described as in (8), and/or δ is bound to lie in an ellipsoid. Ellipsoidal (confidence) regions for δ may come from a statistical study of the uncertainties in the system. For example, suppose that it is known that the uncertainty δ lies in the ellipsoid $\{\ \delta \mid \|F\delta + g\| \leq 1\ \}$ where $F \in \mathbf{R}^{r\times r}$ (full rank) and $g \in \mathbf{R}^r$ are known. It is easy to verify that, to first order, the ith eigenvalue of (13) lies in $\mathcal{D}_i$ as defined in (7) if and only if

$$\begin{aligned} &\sum_{k=1}^{q}\left(a_{ij}\,\mathbf{Re}\left(\frac{w_i^* A_k u_i}{w_i^* u_i}\right) + b_{ij}\,\mathbf{Im}\left(\frac{w_i^* A_k u_i}{w_i^* u_i}\right)\right) x_k \leq \\ &\qquad\qquad c_{ij} - a_{ij}\,\mathbf{Re}\,(\lambda_i) - b_{ij}\,\mathbf{Im}\,(\lambda_i) + \|F^{-T} d_{ij}\| + d_{ij}^T F^{-1} g, \end{aligned}$$

for $j = 1, \ldots, J_i$ where

$$d_{ij} = a_{ij}\left[\ \mathbf{Re}\left(\tfrac{w_i^* \bar{A}_1 u_i}{w_i^* u_i}\right) \ \cdots \ \mathbf{Re}\left(\tfrac{w_i^* \bar{A}_r u_i}{w_i^* u_i}\right)\ \right]^T + b_{ij}\left[\ \mathbf{Im}\left(\tfrac{w_i^* \bar{A}_1 u_i}{w_i^* u_i}\right) \ \cdots \ \mathbf{Im}\left(\tfrac{w_i^* \bar{A}_r u_i}{w_i^* u_i}\right)\ \right]^T.$$

Again, these are a set of linear inequalities in x and can therefore be handled by solving LPs.

7.2 Robust LAC design for systems with multiple models

Here we consider a multiple model approach to robust LAC design. This approach relies on the fact that it is possible to adequately model uncertainty or plant variation by a finite number of system models

$$\dot{z} = A^{(l)}(x)z, \quad l = 1, \ldots, \nu. \tag{16}$$

For robust LAC design in this framework, the goal is to find x such that the eigenvalues of each of the system models (16) are in some desired region of the complex plane. This can be easily done by requiring the eigenvalue-placement specifications to hold for each of the models.

For example, if the desired region $\mathcal{D}_i$ of the ith eigenvalue is given as in (7), then using perturbation formulas from §5.1 we require that

$$\sum_{k=1}^{q}\left(a_{ij}\,\mathbf{Re}\left(\frac{w_i^{(l)*}A_k^{(l)}u_i^{(l)}}{w_i^{(l)*}u_i^{(l)}}\right)+b_{ij}\,\mathbf{Im}\left(\frac{w_i^{(l)*}A_k^{(l)}u_i^{(l)}}{w_i^{(l)*}u_i^{(l)}}\right)\right)x_k \le c_{ij}-a_{ij}\,\mathbf{Re}\left(\lambda_i^{(l)}\right)-b_{ij}\,\mathbf{Im}\left(\lambda_i^{(l)}\right), \tag{17}$$

for $l=1,\ldots,\nu$, where $\lambda_i^{(l)}$, $u_i^{(l)}$, and $w_i^{(l)}$ are the ith eigenvalue, right eigenvector, and left eigenvector of $A^{(l)}(0)$ respectively. Therefore, in the robust case, eigenvalue-placement specifications can still be described as LPs that are just ν times larger. Hence, robust LAC design using a multiple model approach can be easily handled as before.

8 Extension: LAC design based on Lyapunov theory

In this section we show how Lyapunov theory can be used to design low-authority controllers. Lyapunov methods are very powerful and enable us to formulate design objectives *beyond* eigenvalue-placement specifications in terms of SDPs, which can then be solved very efficiently (cf. §4). These design objectives can be combined to get, for example, a desired eigenvalue location for the system while providing a bound on output energy, $\mathcal{L}_2$ gain, *etc.*

By combining the results of §6 and §7 with those presented here it is possible to perform robust actuator/sensor placement or controller structure design that are optimum to first order for a variety of control objectives. This would be very useful, for example, for HAC structure design as well, since it will provide a rationale as to where the actuators/sensors should be placed.

The method of LAC design using Lyapunov theory can be summarized as follows. Suppose that the Lyapunov function $V(z)=z^TPz$, $P\succ 0$ proves some level of performance for some property of the unperturbed or open-loop system (2). Then, under the low-authority assumption, it is reasonable to assume that the Lyapunov function $\widehat{V}(z)=z^T(P+\delta P)z$, $P+\delta P\succ 0$ with δP "*small*" is a Lyapunov function candidate for the same property of the closed-loop system (1). Therefore, as a first order approximation, we can neglect second order (cross) terms such as $x_i\delta P$ in the (bilinear) matrix inequality conditions that are equivalent to $\widehat{V}$ being a Lyapunov function proving (a better) level of performance for the closed-loop system. Hence, the matrix inequalities become jointly linear in x and δP, and therefore, can be easily handled by solving SDPs.

In this section we illustrate this method for two different design specifications, but it should be noted that the method is quite powerful, and can also be applied to handle many other design specifications.

8.1 Bound on output energy

Consider the (closed-loop) linear dynamical system with output

$$\dot{z} = A(x)z, \quad y = C(x)z, \quad z(0) = z_0. \tag{18}$$

The goal is to design x to "moderately" reduce the output energy $\int_0^\infty y^T y \, dt$ of the closed-loop system (18) from that of the unperturbed or open-loop system (*i.e.*, system (18) with $x = 0$).

The output energy of the open-loop system is bounded by $z(0)^T P z(0)$ for any $P \succ 0$ satisfying (see, *e.g.*, [21])

$$A(0)^T P + PA(0) + C(0)^T C(0) \preceq 0. \tag{19}$$

(If the inequality in this equation is replaced by equality, $z(0)^T P z(0)$ gives the exact output energy). The output energy of the closed-loop system (18) is bounded by $z(0)^T (P + \delta P) z(0)$ if there exists δP such that $P + \delta P \succ 0$ and

$$A(x)^T (P + \delta P) + (P + \delta P) A(x) + C(x)^T C(x) \preceq 0. \tag{20}$$

Under the low-authority assumption it is reasonable to assume that δP and x_i are "small" and their product is to first order negligible. Hence, by expanding $A(x)$ and $C(x)$ in (20) to their first order (Taylor) approximation, and neglecting the second order terms such as $x_i \delta P$ we get

$$\begin{array}{r} A(0)^T P + PA(0) + C(0)^T C(0) + A(0)^T \delta P + \delta P A(0) + \\ \sum_{k=1}^q x_k \left(A_k^T P + P A_k + C(0)^T C_k + C_k^T C(0) \right) \preceq 0, \end{array} \tag{21}$$

where $A_k \triangleq \partial A(0)/\partial x_k$, and $C_k \triangleq \partial C(0)/\partial x_k$. (21) is an LMI in the variables $\delta P \in \mathbf{R}^{n \times n}$ and $x \in \mathbf{R}^q$ although we will not write out the LMI explicitly in the standard form $x_1 F_1 + \cdots + x_m F_m \preceq G$ of §4 (leaving LMIs in condensed form, in addition to saving notation, may lead to more efficient computation). By adding the constraint $P + \delta P \succeq 0$ (and constraining $\|\delta P\| \leq 0.2 \|P\| I$ for example to ensure that the first order approximations are accurate), a first order condition for an output energy of $z(0)^T(P+\delta P)z(0)$ for the closed-loop system becomes

$$\begin{aligned} &P + \delta P \succeq 0, \quad \begin{bmatrix} 0.2P & \delta P \\ \delta P & 0.2P \end{bmatrix} \succeq 0, \\ &A(0)^T P + PA(0) + C(0)^T C(0) + A(0)^T \delta P + \delta P A(0) + \\ &\qquad \textstyle\sum_{k=1}^q x_k \left(A_k^T P + PA_k + C(0)^T C_k + C_k^T C(0) \right) \preceq 0, \end{aligned} \tag{22}$$

where P is any positive definite matrix satisfying (19). (In this case it seems reasonable to pick P as the unique solution to the Lyapunov equation $A(0)^T P + PA(0) + C(0)^T C(0) = 0$.) By adding (linear) constraints such as $z(0)^T(P + \delta P)z(0) \leq \epsilon$, $\mathbf{Tr}(P + \delta P) \leq \eta$, *etc.*, that require the output energy to be smaller than some prescribed level, and by minimizing for example $\|x\|_1$, LAC design for output energy specifications can be solved using SDP.

8.2 $\mathcal{L}_2$ gain

Consider the (closed-loop) linear dynamical system with input and output

$$\dot{z} = A(x)z + B(x)w, \quad y = C(x)z + D(x)w. \tag{23}$$

Suppose that the induced $\mathcal{L}_2$ gain from input w to output y of the open-loop system, *i.e.*, system (23) with $x = 0$, is less than γ so that [21]

$$\begin{bmatrix} A(0)^T P + PA(0) + C(0)^T C(0) & PB(0) + C(0)^T D(0) \\ B(0)^T P + D(0)^T C(0) & -\gamma^2 I + D(0)^T D(0) \end{bmatrix} \preceq 0. \tag{24}$$

Then using a similar reasoning to §8.1, to first order, the induced $\mathcal{L}_2$ gain from input w to output y of the closed-loop system is less than $\gamma^2 + \delta(\gamma^2)$ if

$$\begin{aligned} &P + \delta P \succeq 0, \quad \begin{bmatrix} 0.2P & \delta P \\ \delta P & 0.2P \end{bmatrix} \succeq 0, \\ &\begin{bmatrix} \begin{pmatrix} A(0)^T P + PA(0) + \\ C(0)^T C(0) + A(0)^T \delta P + \delta P A(0) + \\ \sum_{k=1}^q x_k \left(A_k^T P + PA_k \right) + \\ \sum_{k=1}^q x_k \left(C(0)^T C_k + C_k^T C(0) \right) \end{pmatrix} & \begin{pmatrix} (P + \delta P)B(0) + C(0)^T D(0) + \\ \sum_{k=1}^q x_k (PB_k^T + C_k^T D(0) + C(0)^T D_k) \end{pmatrix} \\ \begin{pmatrix} B(0)^T (P + \delta P) + D(0)^T C(0) + \\ \sum_{k=1}^q x_k (B_k P + D(0)^T C_k + D_k^T C(0)) \end{pmatrix} & \begin{pmatrix} -(\gamma^2 + \delta(\gamma^2))I + D(0)^T D(0) + \\ \sum_{k=1}^q x_k (D_k^T D(0) + D(0) D_k) \end{pmatrix} \end{bmatrix} \preceq 0, \end{aligned} \tag{25}$$

where $A_k \triangleq \partial A(0)/\partial x_k$, $B_k \triangleq \partial B(0)/\partial x_k$, $C_k \triangleq \partial C(0)/\partial x_k$, and $D_k \triangleq \partial D(0)/\partial x_k$. (25) is an LMI in the variables $\delta P = \delta P^T$, x, and $\delta(\gamma^2)$.

Note that the method of this section works for any P that proves some level of performance for the open-loop system (*e.g.*, any P that satisfies (19) or (24)). This observation highlights a potential weakness of this method because it is not clear which P should be used in for example (21) or (25). However, we conjecture that it does not make much difference which P is chosen because the perturbations are assumed to be small and the P can be adjusted using the free variable δP. Our experience indicates that the P with smallest condition number, or the one that minimizes $\log\det P^{-1}$, seems to work well in practice.

As a final remark, it should be noted that the different LAC design constraints in previous sections can be mixed freely. Since these constraints were either linear inequalities, second-order cone constraints, or LMIs, a semi-definite program solver (*e.g.*, [17]) can be used to compute the design parameter x very efficiently in practice.

For example, x can be a vector of feedback gains of different colocated sensor/actuator pairs, and for the closed-loop system we may require a certain minimum amount of damping and damping ratio in the eigenvalues (cf. §5.2), and a bound on the level of induced $\mathcal{L}_2$ norm from an input to an output (cf. §8.2), while the ith feedback gain is absolutely bounded by $x_{i\max}$ ($-x_{i,\max} \leq x_i \leq x_{i,\max}$). By minimizing say $\|x\|_1$ subject to these design specifications we will (hopefully) obtain a sparse x and therefore many of the sensor/actuator pairs will not be needed (cf. §6).

9 Example: LAC design for 39-bar truss structure

The purpose of this section is to design low-authority controllers for the truss structure shown in Figure 2. The structure consists of 39 bars with stiffness and damping connecting 17 masses at the nodes. The dynamics of the structure are written as $\dot{z} = Az$ where $A \in \mathbf{R}^{64\times 64}$, and the state variable z consists of (a linear combination) of the horizontal and vertical displacements, and rates of displacements of each mass, u_i, v_i, $\dot{u}_i$, and $\dot{v}_i$ respectively for $i = 1, \ldots, 17$.

The goal is to design a controller that achieves an overall damping of at least 0.01 and a damping ratio of at least 0.02. The open-loop eigenvalues and the desired region for the closed-loop eigenvalues of the system are shown in Figure 3. We will assume that the low-authority assumption holds and use the method of §5.2 to design controllers that achieve the required damping and damping ratio. The validity of the low-authority assumption will be verified after each design.

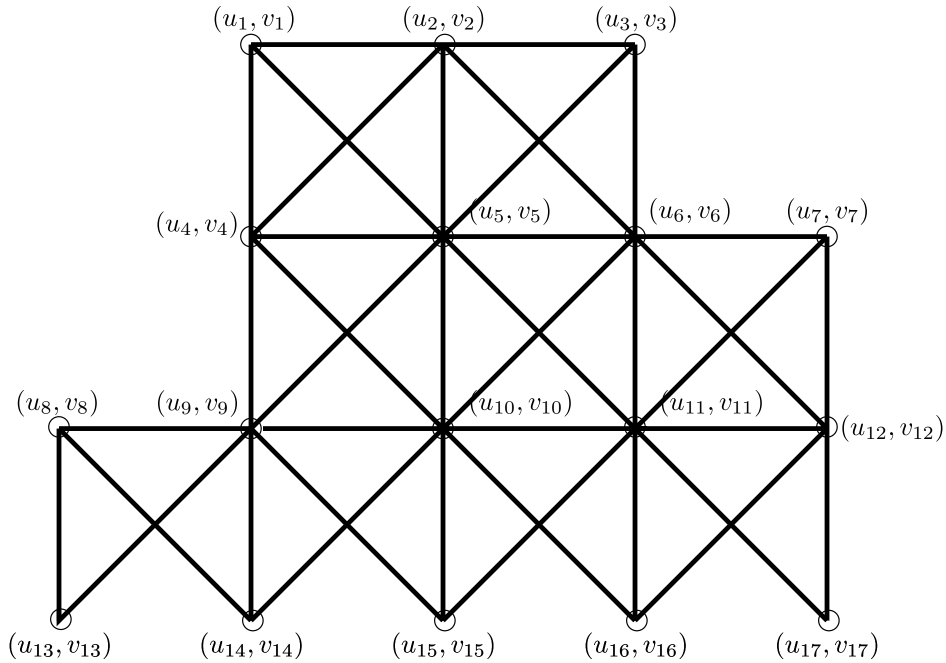


Figure 2: Truss structure consists of 39 bars (stiffness and damping) and 17 nodes (masses).

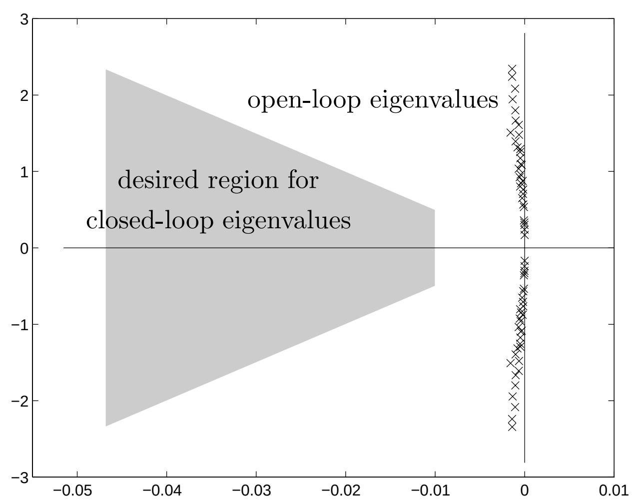


Figure 3: Open-loop eigenvalues of structure and the desired region for closed-loop eigenvalues.

9.1 LAC using dampers along bars

We first consider the case in which we can place a damper of size b_i along each bar to achieve the design specifications. The closed-loop system dynamics are now written as $\dot{z} = A(x)z$ where the design variables are the size of each of the dampers $x = [b_1\ b_2 \cdots b_{39}]^T$ and $A(0) = A$. In this case, $A(x)$ is affine in x.

It is desirable to find a design in which many of the dampings are zero. To achieve such a design we minimize the ℓ_1 norm of x subject to the eigenvalue placement constraints (note that the number of sparsity patterns of x is $2^{39} \approx 10^{12}$ and an exhaustive search method for computing the optimum $\|x\|_0$ is impractical). The resulting LP that must be solved to find the damping design consists of 39 variables and 64 linear inequality constraints. The solution for this problem resulted in 22 out of the 39 possible dampers being zero (this takes several seconds using the LP solver `PCx`* on a typical personal computer). The location of the nonzero dampers are shown in Figure 4. A solid line between two nodes in this figure corresponds to a nonzero damper between those two nodes. The figure shows that, in this case, most of the dampers are on the diagonals of the truss structure. In this case we get $\sum_i b_i = 1.73$ and $\max_i b_i = 0.27$ which are a measure of the total amount of damping material that must be added to the structure and to a single strut.

To verify the low-authority assumption, Figure 5 shows a plot of the actual (not first order approximate) eigenvalues of the closed-loop system. All closed-loop poles satisfy the requirements, or are very close to the boundary, which clearly shows that the low-authority assumption is valid in this case.

9.2 LAC using rate sensors at each node, force actuator along each bar

A more sophisticated design approach is to use active damping. In this case we assume that a rate sensor can be placed at each node (measuring $\dot{u}_i$ and $\dot{v}_i$) and a force actuator can be placed along each bar. We consider an extremely flexible control architecture that allows each sensor to be connected to each actuator via a feedback gain that must be determined. The dynamics of the closed-loop system are written as $\dot{z} = A(x)z$ such that the vector of design variables $x \in \mathbf{R}^{1326}$ represents the elements of the 34×39 matrix of feedback gains from each sensor to each actuator. In this case, $A(x)$ is again affine in x.

The goal is to achieve the eigenvalue placement design specifications with a small number of actuators/sensors and a simple controller topology. This objective is accomplished by

*`PCx` can be downloaded from WWW at URL `http://www-c.mcs.anl.gov/home/otc/Library/PCx/`

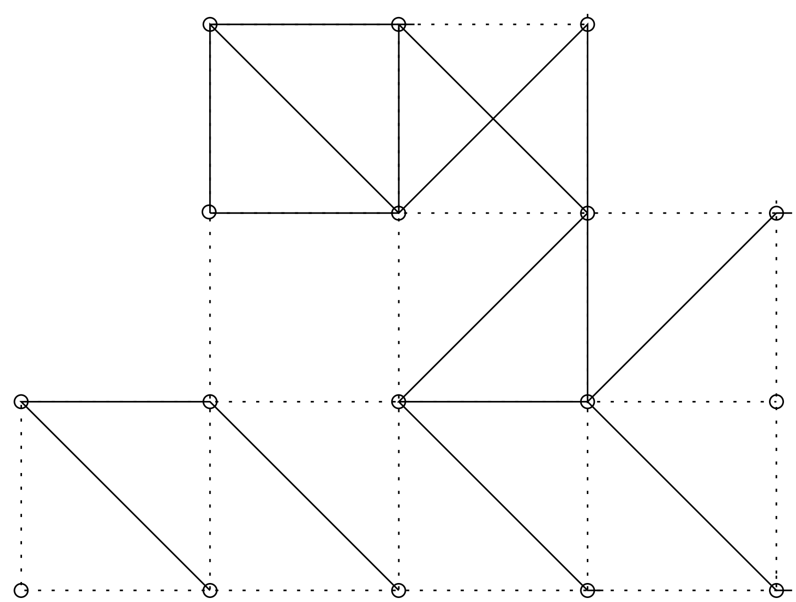


Figure 4: Location of dampers for the LAC design. A solid line between two nodes corresponds to a nonzero damper between those two nodes.

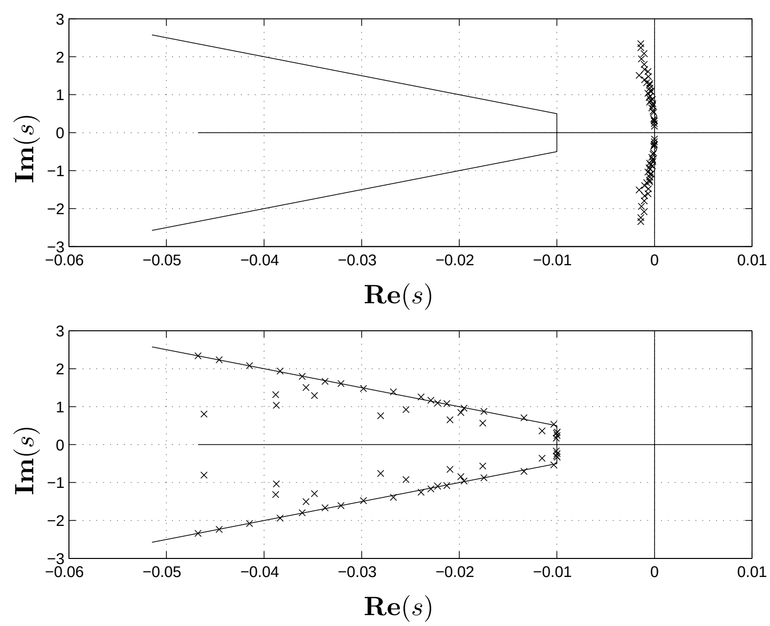


Figure 5: Open-loop and (actual, not approximate) closed-loop eigenvalues.

minimizing the ℓ_1 norm of x subject to the eigenvalue placement specifications, which is an LP with 1326 variables and 64 linear inequality constraints (this takes approximately a minute using PCx on a typical personal computer).

The sparsity pattern of the resulting feedback gain matrix is given in Figure 6. Again, the solution is very sparse and only 16 out of 1326 possible feedback gains are nonzero ($\sum_i |x_i| = 3.44$ and $\max_i |x_i| = 0.56$). Of course, actuators (sensors) that are not connected to a sensor (actuator) can be eliminated. For the solution given here, only 13 (out of 39) actuators, and 15 (out of 34) sensors are required. Figure 7 shows the location of these actuators and sensors.

By examining Figures 6 and 7 it can be seen that the controller is colocated rate feedback, in the sense that there is no feedback path from a sensor to an actuator that does not have the sensor attached to it. It is interesting to note that actuators #6, #21, and #28 use two sensors, while all other actuators use only one sensor. Also, all sensors are connected to only one actuator except for #14 which is connected to two actuators #21 and #22.

Thus, by considering a problem with a very general feedback matrix, the optimization has succeeded in simultaneously performing the sensor/actuator placement problem and the feedback control design. The open-loop and (actual) closed-loop eigenvalues for this design are shown in Figure 8. Again, the closed-loop poles meet or exceed the design specifications, which verifies the low-authority assumption in this design approach.

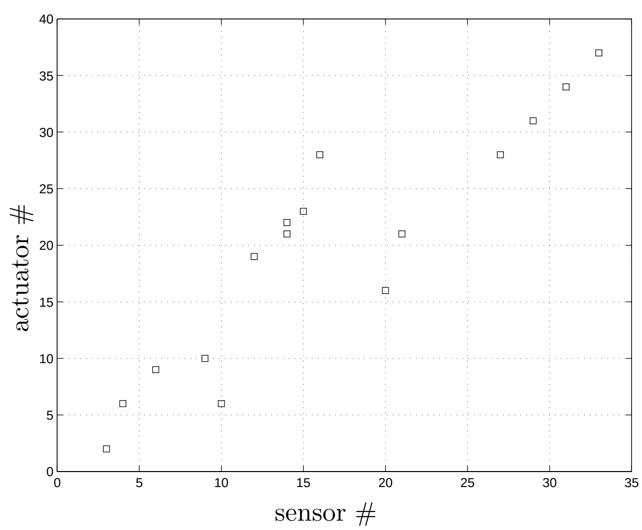


Figure 6: Sparsity pattern of the feedback gain matrix. A nonzero feedback gain from that sensor to actuator is shown by a '□'.

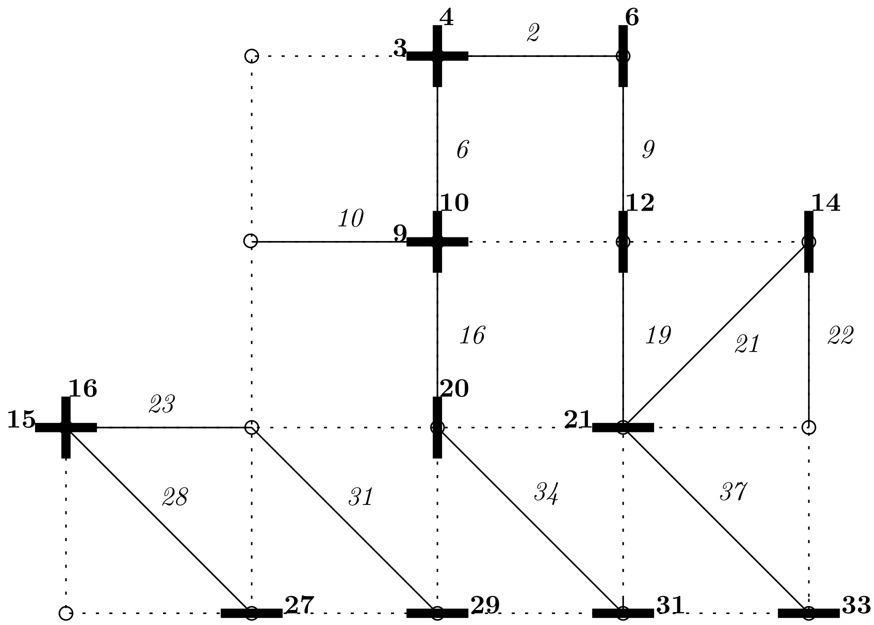


Figure 7: Sensor and actuator locations. A solid line between two nodes corresponds to an actuator between those two nodes. A vertical or horizontal line crossing a node corresponds to a vertical or horizontal rate sensor at that node. The actuator numbers are *italicized* and the sensor numbers are **boldfaced**.

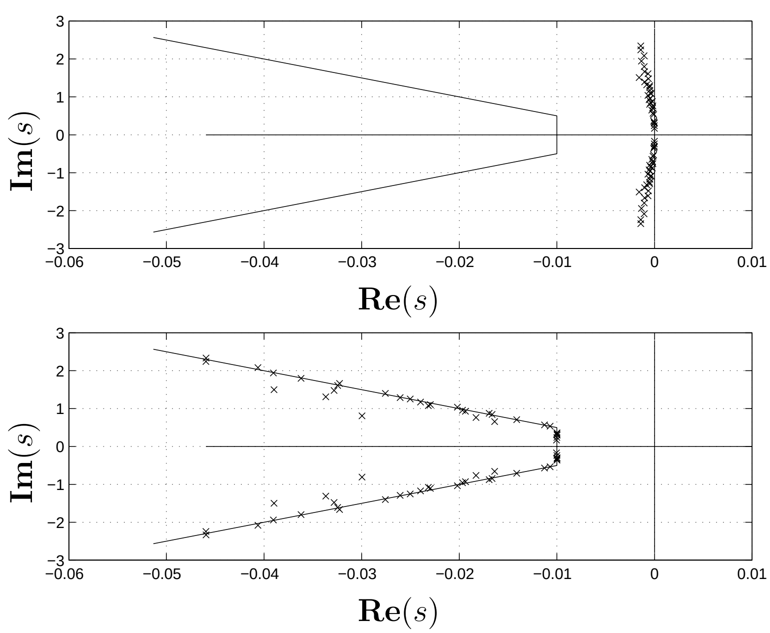


Figure 8: Open-loop and (actual, not approximate) closed-loop eigenvalues.

9.3 Robust LAC example for truss

A key problem with sensor actuator placement problems is the sensitivity of the optimization to the particular system model used [30]. To address this problem, we design a robust LAC based on a multiple model approach as discussed in §7.2. We assume that there are two possible models for the truss: Model 1 is the same truss considered in the previous examples, and Model 2 is the same truss but with the node masses doubled at the 3rd level and halved at the 2nd level. As before, the goal is to find the amount of damping required along each bar to *robustly* place the eigenvalues of the system in the desired region in Figure 3. Once again, we solve this problem by optimizing the ℓ_1 norm of x to find the amount of damping required on each strut.

Robustness is a concern in this problem, because if the dampers are designed solely for Model 1, then the eigenvalues of the closed-loop system corresponding to Model 2 will not satisfy the eigenvalue-placement specifications. This result is shown in Figure 9. The design problem using Model 1 was discussed earlier, and the nonzero damping locations are shown in Figure 4.

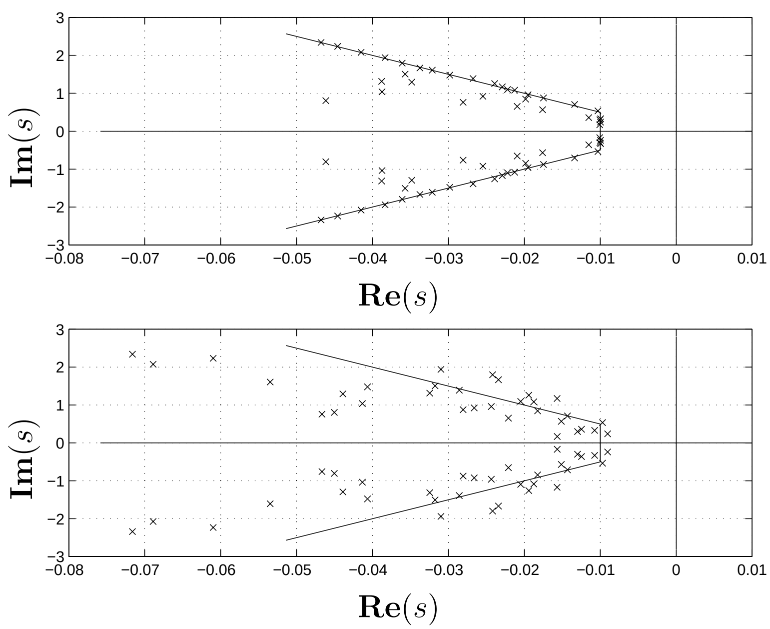


Figure 9: Closed-loop eigenvalues for model 1 (*top*) and model 2 (*bottom*) using dampers designed for model 1.

Figure 10 shows the actual closed-loop eigenvalues for both models for a design using Model 2. As before, if the dampers are solely designed for Model 2, then the closed-loop system corresponding to Model 1 violates the eigenvalue-placement specifications. The nonzero damping locations selected using Model 2 are shown in Figure 11. A comparison of Figures 4 and 11 indicates that there are some similarities in the best locations for the dampers based on these two models (primarily in the first and third level), but the two solutions are very different in the second level.

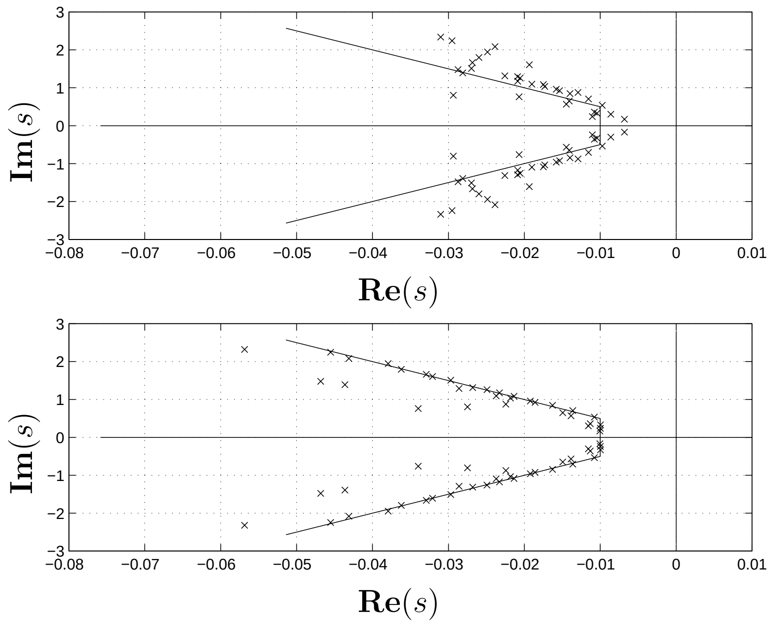


Figure 10: Closed-loop eigenvalues for model 1 (*top*) and Model 2 (*bottom*) using dampers designed for Model 2.

As discussed, we can resolve this difficulty using a robust LAC design that is based on both Model 1 and Model 2. The nonzero damping locations for the robust solution are given in Figure 12. This design is interesting because, as the figure shows, it combines many of the unique placement features of the two non-robust designs. However, for the design based on Model 1, $\sum_i b_i = 1.73$, for the design based on Model 2, $\sum_i b_i = 1.52$, and for the robust design $\sum_i b_i = 1.82$. Thus the robustness is achieved with only slight more damping in the structure. The actual closed-loop eigenvalue locations corresponding to each model is shown in Figure 13. Clearly, the eigenvalue-placement specifications hold for both models.

9.4 Joint HAC/LAC design

This problem investigates a typical LAC application, which is to add damping to a structure to compensate for spillover from a higher authority controller (HAC) that has been designed

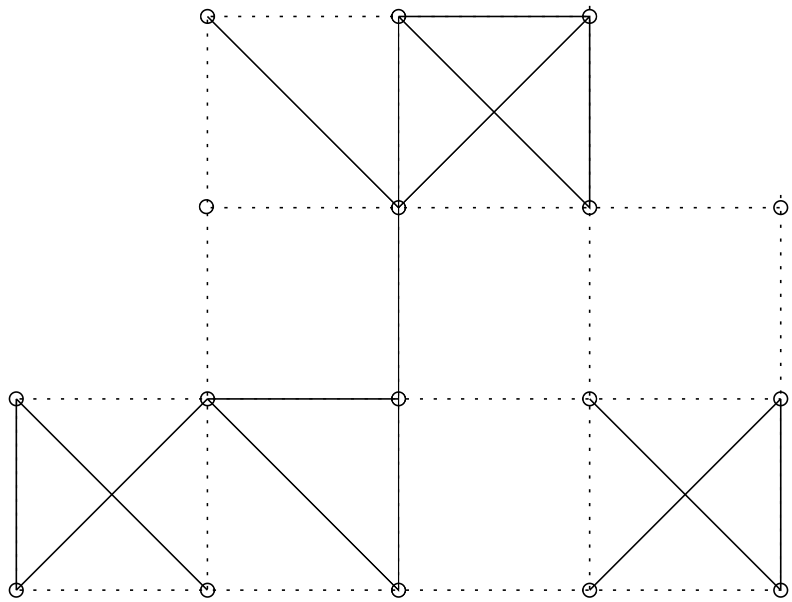


Figure 11: Location of dampers for design based on model 2. A solid line between two nodes corresponds to a nonzero damper between those two nodes.

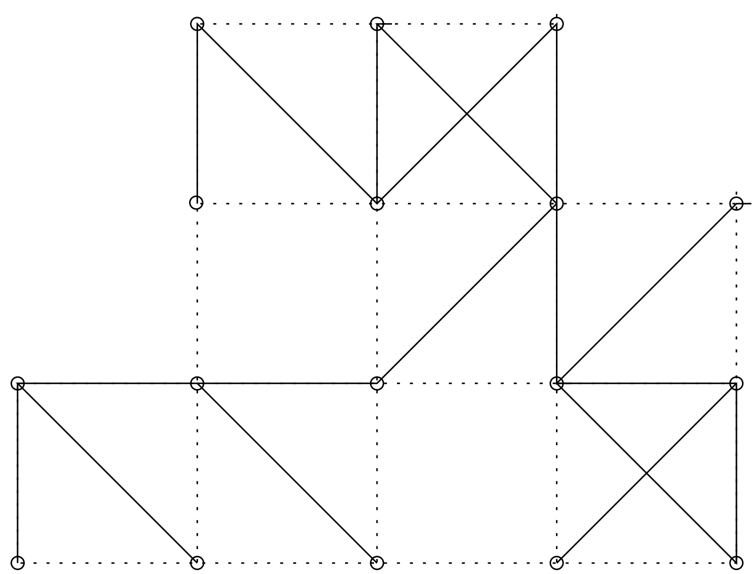


Figure 12: Location of dampers for robust design. A solid line between two nodes corresponds to a nonzero damper between those two nodes.

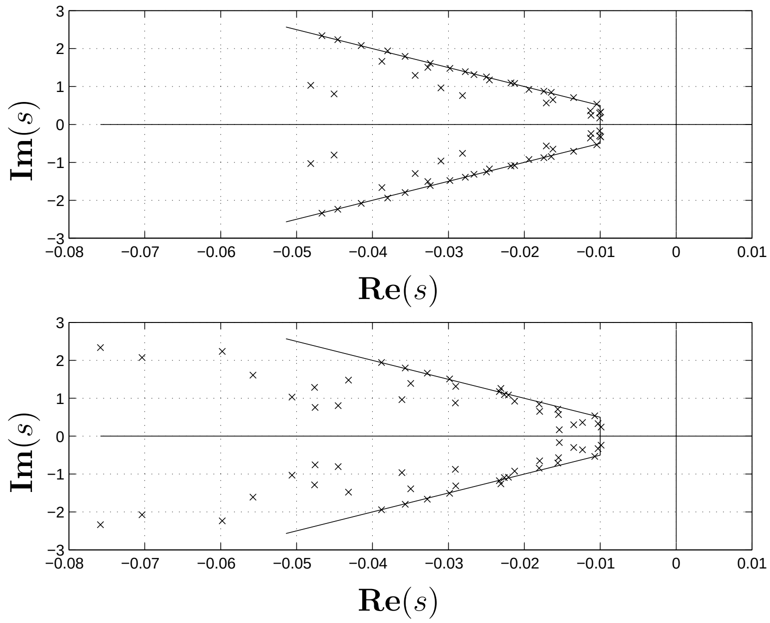


Figure 13: Closed-loop eigenvalues for model 1 (*top*) and model 2 (*bottom*) using robust design.

using a reduced model of the structure. This problem is of interest because it is often difficult to achieve the design specifications using damping alone. For example, if the amount of damping that can be placed along each bar is limited to 0.08, then it is not possible to meet the eigenvalue-placement specifications with dampers alone, even when the maximum amount of damping (*i.e.*, 0.08) is placed along all bars.

The design methodology in this example follows the classic two-step process. We first design a HAC, which is a Linear-Quadratic Gaussian (LQG) controller, to minimize a quadratic cost involving the displacements in the vertical bars (*e.g.*, in a building this corresponds to the interstory drift). Note that the specific design process for the HAC is not important for this paper, and this step could be performed using *eigenvalue-placement*. A key point is that these higher authority designs are typically based on significantly reduced order models of the system to avoid designing a very high order controller [2, 3]. As a result, we would expect a considerable amount of *spillover* of the control authority to the higher frequency modes of the structure. The destabilizing effects of the spillover are addressed during the second step of the design by adding sufficient damping along bars.

More specifically, suppose that the disturbance (exogenous) inputs are left-right and up-down forces at the nodes on the base of the truss, and left-right forces at the nodes on the sides of the truss. The output that needs to be regulated is the displacement of the vertical bars. The actuators for control are force actuators along bars at the first level of the truss as well as force actuators along vertical bars at other levels. The sensors measure the displacement of all bars on the highest level of the truss as well as the displacement of all vertical bars at other levels. (Since the regulating output involves the displacement of vertical bars it is reasonable to have a sensor/actuator pair along each vertical bar, otherwise there is no specific reason in choosing the actuator/sensors as such). The LQG controller is designed based on a reduced-order truss model, using the 5 lowest frequency modes which are the most lightly damped modes and have a good frequency seperation from the remaining 27 modes. The relative weight of the output energy and control input energy in the LQG cost were tuned until the HAC provided reasonable performance for the reduced order model (the output energy to control input energy weighting ratio was 1/15). The eigenvalues of the feedback interconnection of the HAC and the truss are shown in the top of Figure 14 (only eigenvalues near the origin are shown). Note that, as a result of the spillover, the system is actually unstable and the eigenvalue-placement specification is obviously violated.

One way to fix this problem is to find dampers (limited in this case for illustrative purposes to a maximum size of 0.08) along the bars that would achieve the required eigenvalue-placement specifications. In the LAC framework considered in this paper, the open-loop system here is actually the interconnection of the HAC and the truss. The damper design problem can then be posed in exactly the same way as the LP problems discussed earlier. However, with the limited amount of damping allowed in this problem, the eigenvalue-placement specifications still cannot be achieved (Figure 14).

Fortunately it is possible to do much better if the HAC system matrices can also be perturbed (to first order) during the LAC design. In this case the specifications *can* be met. The parameter x in this design now represents the amount of damping added to each strut as well as the perturbations to the controller system matrices (the controller is in modal form). The perturbations to the elements of the controller matrices are limited to 5% of their original values for first order perturbation formulas to be approximately valid. The problem specification is to minimize the sum of the damper values subject to the eigenvalue specifications. The corresponding LP has 419 variables and 483 linear inequality constraints.

The eigenvalue locations for the closed-loop system after adding the dampers and adjusting the HAC are shown in Figure15. The figure clearly shows that this combined controller has sufficiently damped the (unstable) modes of the system. (Note that a couple of eigenvalues slightly violate the damping ratio constraint but we can simply perform another iteration of LAC design to fix this problem.) As shown in Figure 16, the location of the

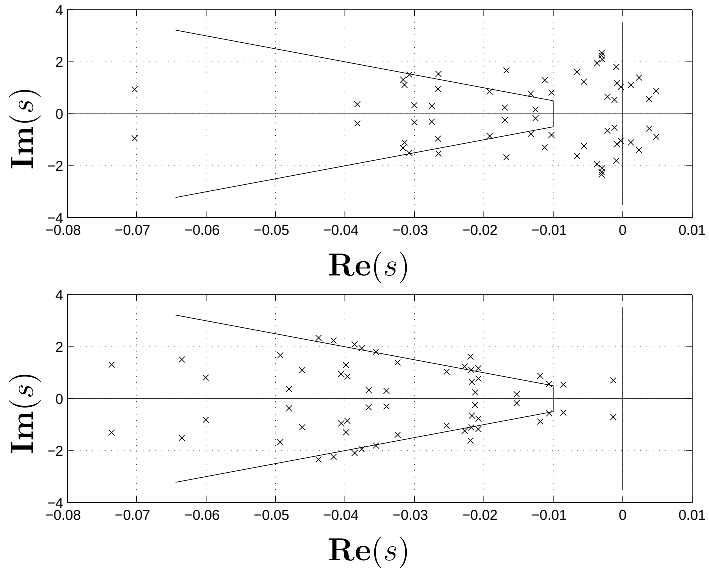


Figure 14: Eigenvalues of HAC around truss (*top*), and eigenvalues of HAC around truss with all dampers set to the maximum value of 0.08 (*bottom*). Clearly, eigenvalue specifications are not satisfied even when the dampings are maximum.

nonzero dampers in this case are quite different than the three designs considered previously. The total amount of damping material added to the structure in these 19 struts is given by $\sum_i b_i = 1.28$.

Finally, Figure 17 shows the Hankel singular values of the HAC before and after adjustment. The plots clearly show that the HAC has been modified as a result of the refinements, but, consistent with the first order assumptions, the change is quite small.

This simple problem shows that there are often key advantages to simultaneously designing the HAC and LAC components of the control architecture. More importantly, however, this example also shows that this entire control design problem can be posed as an LP, which can be solved very efficiently and very quickly on a simple computer.

10 Conclusions and further remarks

In this paper we addressed the problem of robust and sparse LAC design using linear and semidefinite programming. The main points were:

- LP, SOCP and SDP can be used to solve very complex LAC problems, involving complicated cases with substantial spillover. LP, SOCP and SDPs can be solved (globally)

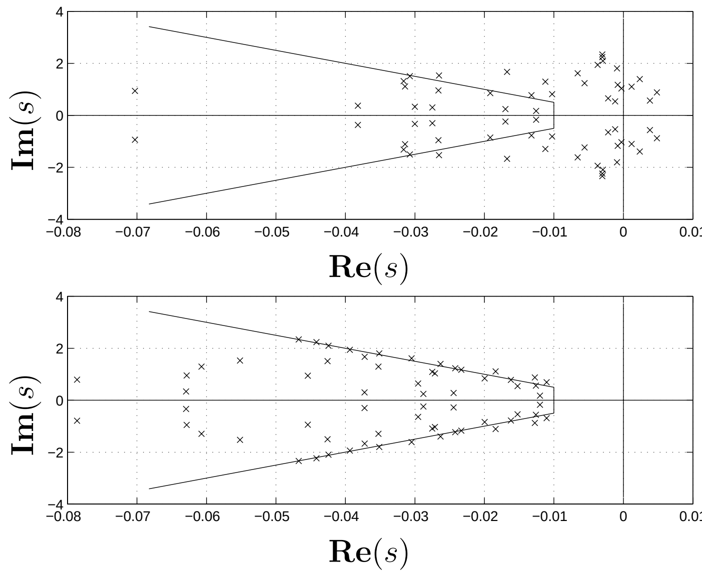


Figure 15: Eigenvalues of the feedback interconnection of the truss and HAC before adding dampers (*top*), and after adding dampers (*bottom*). Here the system matrices of the HAC are also adjusted.

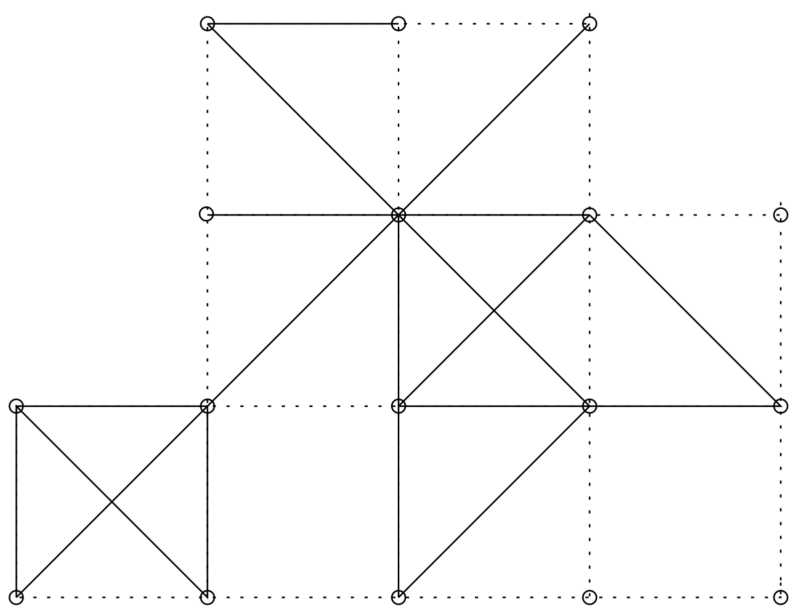


Figure 16: Location of dampers for damper design for the feedback interconnection of the plant and the HAC. A solid line between two nodes corresponds to a nonzero damper between those two nodes.

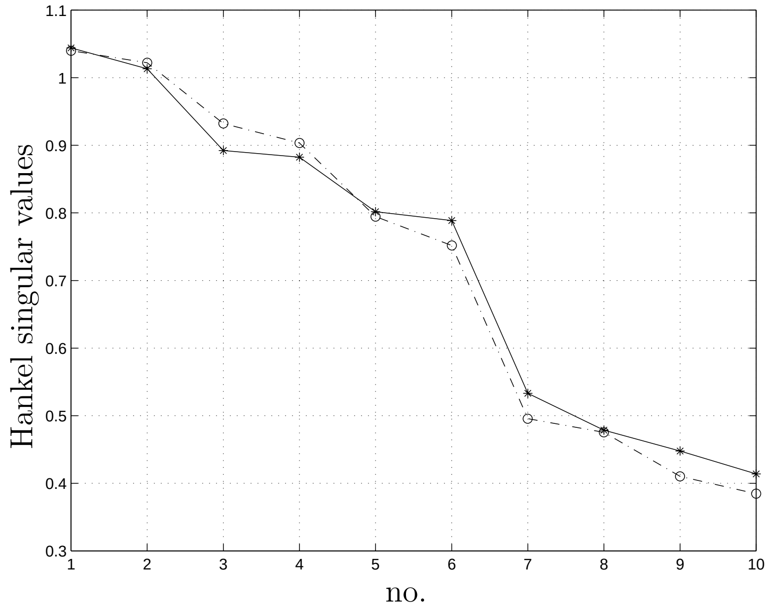


Figure 17: Hankel singular values of HAC before and after LAC adjustment (*dashed* and *solid* curves respectively).

for huge problem sizes.

- In *many* applications it is desirable to compute a sparse x which can be done by an ℓ_1 relaxation method. These applications include actuator/sensor placement and controller topology design.

It was also shown that, Lyapunov theory and LMI formulation can be used to handle a very wide variety of specifications and objectives beyond eigenvalue-placement for LAC design. These include, for example, bounding the output energy of a system, RMS gain, *etc.* Such objectives can be freely mixed with each other, and with the eigenvalue-placement specifications of §5.

A wide variety of control problems, including synthesis with structured uncertainty, fixed-order controller design, decentralized controller synthesis, simultaneous stabilization, multi-objective controller synthesis, *etc.*, can be cast as matrix inequalities that are bilinear in the variables (bilinear matrix inequalities or BMIs). See, for example, Safonov, Goh, and others [31, 32, 33, 34, 35]. BMIs are hard to solve directly so people have been looking at a variety of iterative schemes to solve them. The idea of linearizing matrix inequalities of §8 can be used as a path-following (homotopy) method for solving (locally) BMIs. This approach offers another alternative which allows us to approach the overall design objective by iteratively solving a sequence of linearized problems. Starting from the initial (open-loop) system, the idea is to design better and better controllers by slowly improving the design objective (*e.g.*, given a reduced-order, decentralized, or fixed architecture controller could iteratively design for lower values of induced $\mathcal{L}_2$ norm). Since the design objective in consecutive problems are close, at each step, we can linearize the bilinear matrix inequality to accurately design a controller that is slightly better than the previous one by solving an SDP.

References

[1] J. H. Wykes. Structural dynamic stability augmentation and gust alleviation of flexible aircraft. In *AIAA Paper 68-1067, AIAA Anual Meeting*, October 1968.

[2] J. N. Aubrun. Theory of control of structures by low-authority controllers. *Journal of Guidance and Control*, 3(5):444–451, September 1980.

[3] J. N. Aubrun, N. K. Gupta, and M. G. Lyons. Large space structures control: An integrated approach. In *Proceedings of AIAA Guidance and Control Conference*, Boulder, Colorado, August 1979.

[4] T. Williams. Transmission zeros and high-authority/low-authority control of flexible space structures. *Journal of Guidance, Control, and Dynamics*, 17(1):170–4, January 1994.

[5] C. T. Sun and Y. P. Lu. *Vibration Damping of Structural Elements*. Prentice Hall, 1995.

[6] R. J. Vanderbei. *Linear Programming: Foundations and Extensions*. Kluwer Academic Publishers, 1997.

[7] Yu. Nesterov and A. Nemirovsky. *Interior-point polynomial methods in convex programming*, volume 13 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, 1994.

[8] L. Vandenberghe and S. Boyd. A primal-dual potential reduction method for problems involving matrix inequalities. *Mathematical Programming*, 69(1):205–236, July 1995.

[9] L. A. Zadeh and B. H. Whalen. On optimal control and linear programming. *IRE Transactions on Automatic Control*, pages 45–6, July 1962.

[10] J. Richalet. Industrial applications of model based predictive control. *Automatica*, 29(5):1251–1274, 1993.

[11] R. J. Vanderbei. LOQO user's manual. Technical Report SOL 92–05, Dept. of Civil Engineering and Operations Research, Princeton University, Princeton, NJ 08544, USA, 1992.

[12] Y. Zhang. *User's guide to LIPSOL: a matlab toolkit for linear programming interior-point solvers.* Math. & Stat. Dept., Univ. Md. Baltimore County, October 1994. Beta release.

[13] J. Czyzyk, S. Mehrotra, and S. J. Wright. *PCx User Guide*. Optimization Technology Cneter, March 1997.

[14] F. Alizadeh, J.-P. Haeberly, and M. Overton. A new primal-dual interior-point method for semidefinite programming. In *Proceedings of the Fifth SIAM Conference on Applied Linear Algebra, Snowbird, Utah*, June 1994.

[15] L. Vandenberghe and S. Boyd. SP*: Software for Semidefinite Programming. User's Guide, Beta Version.* Stanford University, October 1994. Available at `http://www-isl.stanford.edu/people/boyd`.

[16] K. Fujisawa and M. Kojima. SDPA (semidefinite programming algorithm) user's manual. Technical Report B-308, Department of Mathematical and Computing Sciences. Tokyo Institute of Technology, 1995.

[17] S.-P. Wu and S. Boyd. SDPSOL: *A Parser/Solver for Semidefinite Programming and Determinant Maximization Problems with Matrix Structure. User's Guide, Version Beta.* Stanford University, June 1996.

[18] P. Gahinet and A. Nemirovskii. *LMI Lab: A Package for Manipulating and Solving LMIs*. INRIA, 1993.

[19] F. Alizadeh, J. P. Haeberly, M. V. Nayakkankuppam, and M. L. Overton. SDPPACK *User's Guide, Version 0.8 Beta.* NYU, June 1997.

[20] B. Borchers. CSDP*, a C library for semidefinite programming*. New Mexico Tech, March 1997.

[21] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*, volume 15 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, June 1994.

[22] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra and Applications*, 284(1-3):193–228, November 1998.

[23] M. S. Lobo, L. Vandenberghe, and S. Boyd. SOCP*: Software for Second-Order Cone Programming*. Information Systems Laboratory, Stanford University, 1997.

[24] F. Alizadeh, J. P. Haeberly, M. V. Nayakkankuppam, M. L. Overton, and S. Schmieta. SDPPACK *User's Guide, Version 0.9 Beta.* NYU, June 1997.

[25] T. Kato. *A Short Introduction to Perturbation Theory for Linear Operators*. Springer-Verlag, 1982.

[26] S. S. Chen, D. Donoho, and M. A. Saunders. Atomic decomposition of basis pursuit. Technical report, Dept. of Statistics, Stanford University, February 1996.

[27] B. D. Rao and I. F. Gorodnitsky. Affine scaling transformation based methods for computing low complexity sparse solutions. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages 1783–6, Atlanta, GA, May 1996.

[28] G. Harikumar and Y. Bresler. A new algorithm for computing sparse solutions to linear inverse problems. In *Proc. ICASSP 96*, pages 1331–1334, May 1996.

[29] M. Mesbahi and G. P. Papavassilopoulos. On the rank minimization problem over a positive semidefinite linear matrix inequality. *IEEE Transactions on Automatic Control*, AC-42(2):239–43, February 1997.

[30] E. H. Anderson and N. W. Hagood. Robust placement of actuators and dampers for structural control. Technical Report of the Space Engineering Research Center SERC#14-93, Massachusetts Institute of Technology, October 1993.

[31] K. C. Goh, M. G. Safonov, and G. P. Papavassilopoulos. A global optimization approach for the BMI problem. In *Proceedings of IEEE Conference on Decision and Control*, volume 4, pages 2009–14, Lake Buena Vista, FL, December 1994.

[32] M. G. Safonov, K. C. Goh, and J. H. Ly. Control system synthesis via bilinear matrix inequalities. In *Proceedings of American Control Conference*, volume 1, pages 45–9, Baltimore, MD, 1994.

[33] K. C. Goh, L. Turan, M. G. Safonov, G. P. Papavassilopoulos, and J. H. Ly. Biaffine matrix inequality properties and computational methods. In *Proc. American Control Conf.*, pages 850–855, 1994.

[34] K.-C. Goh. *Robust Control Synthesis via Bilinear Matrix Inequalities*. PhD thesis, University of Southern California, May 1995.

[35] M. Mesbahi, G. P. Papavassilopoulos, and M. G. Safonov. Matrix cones, complementarity problems and the bilinear matrix inequality. In *Proceedings of IEEE Conference on Decision and Control*, volume 3, pages 3102–7, New Orleans, LA, 1995.