Control Applications of Nonlinear Convex Programming

Stephen Boyd [1], César Crusius [2], Anders Hansson [3]

Information Systems Laboratory, Electrical Engineering Department, Stanford University

Abstract

Since 1984 there has been a concentrated effort to develop efficient interior-point methods for linear programming (LP). In the last few years researchers have begun to appreciate a very important property of these interior-point methods (beyond their efficiency for LP): they extend gracefully to *nonlinear* convex optimization problems. New interior-point algorithms for problem classes such as semidefinite programming (SDP) or second-order cone programming (SOCP) are now approaching the extreme efficiency of modern linear programming codes.

In this paper we discuss three examples of areas of control where our ability to efficiently solve nonlinear convex optimization problems opens up new applications. In the first example we show how SOCP can be used to solve robust open-loop optimal control problems. In the second example, we show how SOCP can be used to simultaneously design the set-point and feedback gains for a controller, and compare this method with the more standard approach. Our final application concerns analysis and synthesis via linear matrix inequalities and SDP.

Submitted to a special issue of *Journal of Process Control*, edited by Y. Arkun & S. Shah, for papers presented at the 1997 *IFAC Conference on Advanced Process Control*, June 1997, Banff. This and related papers available via anonymous FTP at `isl.stanford.edu` or via WWW at URL `http://www-isl.stanford.edu/people/boyd`.

Key words: semidefinite programming, linear programming, convex optimization, interior-point methods, robust optimal control

[1] 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).
[2] Research supported in part by CAPES, Brazil.
[3] Research supported by the Swedish Research Council for Engineering Sciences (under 95–838) and MURI (under F49620-95-1-0525).

1 New advances in nonlinear convex optimization

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

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & a_i^T x \leq b_i, \quad i = 1, \dots, L, \\ & Fx = g, \end{aligned} \tag{1}$$

where the vector x is the optimization variable and a_i, b_i, c, F, and g are problem parameters. Linear programming has been used in a wide variety of fields since Dantzig introduced the simplex method in 1948 [Dan63]. 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 [ZW62]. In the late 70s, Richalet [Ric93] 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 [ML97].

In 1984, Karmarkar introduced a new interior-point algorithm for linear programming which was more efficient than the simplex method in terms of worst-case complexity analysis and also in practice [Kar84]. This event spurred intense research in interior-point methods, which continues even now. In the last year or so several books have been written on interior-point methods for linear programming, *e.g.*, by Wright [Wri97] and Vanderbei [Van97]. Moreover, several high quality, efficient implementations of interior-point LP solvers have become available (see, *e.g.*, [Van92,Zha94,CMW97,GA94,MR97]).

The most obvious advantages of interior-point over simplex-based methods for LP are efficiency (for medium and large-scale problems) and the ability to use approximate search directions, which allows the use of conjugate gradient or related methods that can exploit problem structure. One advantage which is not yet as well appreciated is that interior-point methods extend gracefully to *nonlinear* convex optimization problems, whereas the simplex method, which is based very much on the linear structure of the problem, does not. This observation was made first by Nesterov and Nemirovski [NN94], who developed a general framework for interior-point methods for a very wide variety of nonlinear convex optimization problems. Interior-point methods for nonlinear convex optimization problems have been found to have many of the same characteristics of the methods for LP, *e.g.*, they have polynomial-time worst case complexity, and they are extremely efficient in practice, requiring a number of iterations that hardly varies with problem size, and is typically between 5 and

50. Each iteration requires the solution (or approximate solution) of a linear system of equations.

While the ability of interior-point methods to solve large-scale LPs faster than simplex is certainly important, it is the ability of interior-point methods to solve nonlinear, nondifferentiable convex problems with the same efficiency that opens up many new possible applications.

As an example consider *semidefinite programming* (SDP), a problem family which has the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & x_1 A_1 + \cdots + x_n A_n \preceq B, \\ & Fx = g \end{array} \tag{2}$$

where A_i and B are symmetric $m \times m$ matrices, and the inequality $\preceq$ denotes matrix inequality, *i.e.*, $X \preceq Y$ means $Y - X$ is positive semidefinite. The constraint $x_1 A_1 + \cdots + x_n A_n \preceq B$ 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.*, [VB95,Ali95,OW97,AHO94]) and several SDP codes are now widely available [VB94,FK95,WB96,GN93,AHNO97,Bor97]. The ability to numerically solve SDPs with great efficiency is being applied in several fields, *e.g.*, combinatorial optimization and control [BEFB94]. As a result, SDP is currently a highly active research area.

As another example consider the *second-order cone program* (SOCP) which has the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & \|A_i x + b_i\| \le c_i^T x + d_i, \quad i = 1, \ldots, L, \\ & Fx = g, \end{array} \tag{3}$$

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.*, [LVBL97]. Moreover, efficient interior-point software for SOCP is now available [LVB97,AHN$^+$97].

In this paper we will examine three areas of control where our ability to numerically solve nonlinear, nondifferentiable convex optimization problems, by new interior-point methods, yields new applications. In §2 we show how SOCP can be used to solve robust optimal control problems, which, we predict, will be useful in model predictive control. In §3 we show how SOCP can be used to synthesize optimal linear and affine controllers. We show how this method can be used to jointly synthesize set-points and feedback gains, and compare this method with the usual method in which the set-points and feedback gains are chosen independently. Finally, in §4, we give an example of control analysis and synthesis via linear matrix inequalities. Some general conclusions are given in §5.

2 Robust optimal control

We consider a discrete-time linear dynamic system with finite impulse response, described by the convolution equation

$$y(t) = h_0 u(t) + h_1 u(t-1) + \cdots + h_k u(t-k), \tag{4}$$

where u is the input sequence, y is the output sequence, and h_t is the tth impulse response coefficient. In this section we assume that $u(t)$ and $y(t)$ are scalar, *i.e.*, that our system is single-input single-output, in order to simplify the exposition, but the method readily extends to the case of vector (*i.e.*, multiple) inputs and outputs. (Indeed, the method extends to multi-rate or time-varying systems; linearity is the key property.)

We assume (for now) that the impulse response coefficients $h_0, \ldots, h_k$ are known, and that $u(t) = 0$ for $t \leq 0$. In finite horizon optimal control (or input design) our task is to find an input sequence $u = (u(1), \ldots, u(N))$ up to some input horizon N, that optimizes some objective subject to some constraints on u and y.

To be specific, we might have a desired or target output trajectory $y_{\text{des}} = (y_{\text{des}}(1), \ldots, y_{\text{des}}(M))$, defined up to some output horizon M, and want to minimize the maximum or peak tracking error, given by

$$E = \max_{t=1,\ldots,M} \left| y(t) - y_{\text{des}}(t) \right|.$$

Typical constraints might include an allowable range for the input, *i.e.*,

$$U_{\text{low}} \leq u(t) \leq U_{\text{high}}, \quad t = 1, \ldots, N,$$

and a limit on the slew rate of the input signal, *i.e.*,

$$|u(t+1) - u(t)| \leq S, \quad t = 1, \ldots, N-1.$$

The resulting optimal control problem is to choose u, respecting the input range and slew rate limit, in order to minimize the peak tracking error. This can be expressed as the optimization problem

$$\begin{array}{ll} \text{minimize} & E = \max_{t=1,\dots,M} |y(t) - y_{\text{des}}(t)| \\ \text{subject to} & U_{\text{low}} \le u(t) \le U_{\text{high}}, \quad t = 1, \dots, N, \\ & |u(t+1) - u(t)| \le S, \quad t = 1, \dots, N-1, \end{array} \tag{5}$$

where the optimization variable is $u = (u(1), \dots, u(N))$. We will now show that the optimal control problem (5) can be cast in the form of a linear program. This implies that we can solve the optimal control problem using standard LP software (although greater efficiency can be obtained by developing a specialized code that exploits the special structure of this problem; see, *e.g.*, [Wri93,Wri96,BVG94]).

We first note from (4) that $y(t)$ is a linear function of u. We have $y = Au$ where A is the Toeplitz matrix defined by

$$A_{ij} = \begin{cases} h_{i-j} & \text{if } 0 \le i - j \le k, \\ 0 & \text{otherwise} \end{cases}$$

If a_i^T is the ith row of A, we have

$$y(t) - y_{\text{des}}(t) = a_t^T u - y_{\text{des}}(t)$$

so the problem (5) can be formulated as the LP

$$\begin{array}{ll} \text{minimize} & \gamma \\ \text{subject to} & -\gamma \le a_t^T u - y_{\text{des}}(t) \le \gamma, \quad t = 1, \dots, M, \\ & U_{\text{low}} \le u(t) \le U_{\text{high}}, \quad t = 1, \dots, N, \\ & -S \le u(t+1) - u(t) \le S, \quad t = 1, \dots, N-1, \end{array} \tag{6}$$

where the variables are $u \in \mathbf{R}^N$ and $\gamma \in \mathbf{R}$. This problem is an LP, with $N+1$ variables, $2M + 2N + 2(N-1)$ linear inequality constraints, and no equality constraints.

Let us make several comments before moving on to our next topic. The idea of reformulating an optimal control problem as a linear program extends to far more complex problems, with multiple inputs and outputs and various piecewise linear convex constraints or objectives. Using quadratic programming (QP) instead of LP, for example, we can include traditional sum-of-squares objectives like

$$J = \sum_{t=1}^{M} \left((y(t) - y_{\text{des}}(t)\right)^2 + \rho \sum_{t=1}^{N} u(t)^2.$$

Thus, linear and quadratic programming can be used to (numerically) solve large, complex optimal control problems which, obviously, have no analytical solution. This observation is the basis for model predictive control, in which an LP or QP such as (6) is solved at each time step.

What we have described so far is well known, and certainly follows the spirit of Zadeh and Whalen's observation that some optimal control problems can be solved using LP. We now describe a useful extension that is not yet well known, and relies on our (new) ability to efficiently solve nonlinear convex optimization problems, specifically, SOCPs.

So far we have assumed that the impulse response is known. Now we assume that the impulse response $h = (h_0, \ldots, h_k)$ is not known exactly; instead, all we know is that it lies in an *ellipsoid*:

$$\mathcal{H} = \{ \ h \mid h = \overline{h} + Fp, \ \|p\| \leq 1 \ \}, \qquad (7)$$

where $\|p\| = \sqrt{p^T p}$. We assume that the vector $\overline{h} \in \mathbf{R}^{k+1}$ and matrix $F \in \mathbf{R}^{(k+1)\times q}$, which describe the ellipsoid $\mathcal{H}$, are given.

We can think of $\overline{h}$ (which is the center of the ellipsoid) as the *nominal* impulse response; the matrix F describes the shape and size of the *uncertainty* in the impulse response h. For example, $F = 0$ means the ellipsoid collapses to a single point, *i.e.*, we know h exactly. If the rank of F is less than $k + 1$, then we know the impulse response exactly in some directions (specifically, directions $z \neq 0$ for which $F^T z = 0$). In general, the singular values of F give the lengths of the semi-axes of $\mathcal{H}$, and the left singular vectors give the semi-axes.

Before continuing let us give a specific example of where the ellipsoid uncertainty description (7) might arise in practice. Suppose we perform some system identification tests, in the presence of some measurement or system noise. Assuming or approximating these noises as Gaussian will lead us to an estimate $\hat{h}$ of the impulse response, along with a measurement error covariance Σ_{est}. The associated confidence ellipsoids have the form

$$\mathcal{E}_\alpha = \{ \ h \mid (h - \hat{h})^T \Sigma_{\mathrm{est}}^{-1} (h - \hat{h}) \leq \alpha \ \},$$

where $\alpha > 0$ sets the confidence level. This confidence ellipsoid can be expressed in the form (7) with $\overline{h} = \hat{h}$ and $F = (\alpha\Sigma_{\mathrm{est}})^{1/2}$ (where $X^{1/2}$ denotes the symmetric positive definite square root of the symmetric positive definite matrix X).

Now let us reconsider our original optimal control problem (5). Since the impulse response h is now uncertain (assuming $F \neq 0$), so is the output y resulting from a particular input u. One way to reformulate the optimal control problem, taking into account the uncertainty in h, is to consider the *worst case* peak tracking error, *i.e.*,

$$E_{\rm wc} = \max_{h \in \mathcal{H}} \max_{t=1,\ldots,M} |y(t) - y_{\rm des}(t)|.$$

This leads us to the optimization problem

$$\begin{array}{ll} \text{minimize} & E_{\rm wc} = \max_{h \in \mathcal{H}} \max_{t=1,\ldots,M} |y(t) - y_{\rm des}(t)| \\ \text{subject to} & U_{\rm low} \leq u(t) \leq U_{\rm high}, \quad t = 1, \ldots, N, \\ & |u(t+1) - u(t)| \leq S, \quad t = 1, \ldots, N-1, \end{array} \tag{8}$$

which is the *robust* version of (5).

Before discussing how we can solve (8) via nonlinear convex optimization, let us make some general comments about what the solutions will look like. First we note that when there is no uncertainty, the robust optimal control problem (8) reduces to the original optimal control problem (5). In general, however, the two solutions will differ. The optimal u for the robust problem must work well for *all* impulse responses in the ellipsoid $\mathcal{H}$, whereas the optimal u for the original problem must work well only for the given impulse response. Very roughly speaking, the robust optimal u will generally be more 'cautious' than the optimal u for the non-robust problem; the robust optimal u will not drive or excite the system in directions in which the uncertainty is large. (This will become clearer below.)

We can think of the robust problem as a *regularization* of the original problem, in the following sense. In the original problem it is assumed that the impulse response is known perfectly, so complete confidence is put on the y that results via (4) from a particular u. If adding, say, a large oscillation to u will decrease the objective even a very small amount, then the optimal u will include the oscillation. This will not happen in the robust version, unless the oscillation exploits some common feature of all the impulse responses in $\mathcal{H}$. Thus, the robust problem (8) can be thought of as a regularization of the original problem (5) that is based on our knowledge or confidence in the plant, and not on some generic feature like smoothness (which is the standard basis for regularization). The optimal u for (8) will not take advantage of details of any particular $h \in \mathcal{H}$; it will only take advantage of features that occur in all $h \in \mathcal{H}$.

Now we will show how the robust optimal control problem can be solved via SOCP. It is still true that $y(t)$ is a linear function of u, and can be expressed in the form $y(t) = a_t^T u$. The difference is that now the vector a_t is uncertain. For our analysis here, it will be convenient to write $y(t)$ in the form

$$y(t) = (\overline{h} + Fp)^T D_t u,$$

where $\|p\| \leq 1$ and D_t is a matrix that selects and reorders the components of u appropriately. For example, since $y(1) = h_0 u(1)$ and $y(2) = h_0 u(2) + h_1 u(1)$, we have

$$D_1 = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}, \qquad D_2 = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$

and so on. The worst case peak tracking error can be expressed as

$$\begin{aligned} E_{\mathrm{wc}} &= \max_{h \in \mathcal{H}} \max_{t=1,\ldots,M} |y(t) - y_{\mathrm{des}}(t)| \\ &= \max_{t=1,\ldots,M} \max_{\|p\| \leq 1} |(\overline{h} + Fp)^T D_t u - y_{\mathrm{des}}(t)|. \end{aligned}$$

We can analytically solve for the worst (*i.e.*, maximizing) p, by writing the last expression as

$$|(\overline{h} + Fp)^T D_t u - y_{\mathrm{des}}(t)| = |p^T F^T D_t u + (\overline{h}^T D_t u - y_{\mathrm{des}}(t))|. \tag{9}$$

Now we use the fact that for any vector c, $p^T c$ ranges between the values $\pm\|c\|$ as p ranges over the unit ball $\|p\| \leq 1$ (with the extremes taken on when $p = \pm c/\|c\|$). Thus we have

$$\max_{\|p\| \leq 1} |(\overline{h} + Fp)^T D_t u - y_{\mathrm{des}}(t)| = \|F^T D_t u\| + |(\overline{h}^T D_t u - y_{\mathrm{des}}(t))|$$

and the maximizing p is given by

$$p_{\max} = \frac{\mathrm{sign}(\overline{h}^T D_t u - y_{\mathrm{des}}(t))}{\|F^T D_t u\|} F^T D_t u. \tag{10}$$

Note that the maximizing p depends on t.

We can therefore express the worst case peak tracking error in explicit form as

$$E_{\mathrm{wc}} = \max_{t=1,\ldots,M} \left(\|F^T D_t u\| + |(\overline{h}^T D_t u - y_{\mathrm{des}}(t))| \right).$$

A maximizing (*i.e.*, worst case) p can found as follows: first, find a time s at which the max over t above is achieved, *i.e.*,

$$E_{\mathrm{wc}} = \|F^T D_s u\| + |(\overline{h}^T D_s u - y_{\mathrm{des}}(s))|.$$

Now, take p from (10) with $t = s$. (We note the great practical value of knowing the specific 'worst' impulse response $h \in \mathcal{H}$, *i.e.*, one that yields the worst case peak tracking error $E_{\rm wc}$.)

Finally we can pose the robust optimal control problem (8) as an SOCP. We first express it as

$$\begin{array}{ll} \text{minimize} & \gamma \\ \text{subject to} & \|F^T D_t u\| + \overline{h}^T D_t u - y_{\rm des}(t) \le \gamma \quad t = 1, \dots, M, \\ & \|F^T D_t u\| - (\overline{h}^T D_t u - y_{\rm des}(t)) \le \gamma \quad t = 1, \dots, M, \\ & U_{\rm low} \le u(t) \le U_{\rm high}, \quad t = 1, \dots, N, \\ & -S \le u(t+1) - u(t) \le S, \quad t = 1, \dots, N-1, \end{array} \tag{11}$$

where the variables are $u \in \mathbf{R}^{k+1}$ and $\gamma \in \mathbf{R}$. Simply rearranging the terms in the inequalities yields the standard SOCP form (3). Thus, we can solve the robust optimal control problem by formulating it as a nonlinear convex optimization problem, specifically, an SOCP. (Here too we can exploit the structure of the problem to develop a custom, very efficient SOCP solver specifically for robust optimal control.)

Examining the first two constraints in (11), which are equivalent to

$$\|F^T D_t u\| + |(\overline{h}^T D_t u - y_{\rm des}(t))| \le \gamma,$$

reveals the effect of the uncertainty. Without uncertainty it reduces to

$$|(\overline{h}^T D_t u - y_{\rm des}(t))| \le \gamma.$$

The effect of the uncertainty is to tighten this constraint by adding the (nonnegative) term $\|F^T D_t u\|$ on the left hand side. This term can be thought of as the extra 'margin' that must be maintained to guarantee the constraint is satisfied for all $h \in \mathcal{H}$. We can interpret $\|F^T D_t u\|$ as a measure of how much the input u 'excites' or 'drives' the uncertainty in the system.

The observation that some robust convex optimization problems can be solved using new interior-point methods has been made recently by El Ghaoui and Lebret [EL96] and Ben Tal and Nemirovski [BTN96]. In this section we have considered a specific example, using a peak tracking error criterion, but the method is quite general. It can, for example, be used to find the worst case quadratic cost (see Lobo et al [LVBL97]).

There are several other general ways to describe the uncertainty in the impulse response. The ellipsoid $\mathcal{H}$ can be replaced by a polytope describing the uncertainty; in some cases the resulting robust optimal control problem can be cast as an LP (see, *e.g.*, [CM87,AP92,ZM93]). It is also possible to use a statistical description, *e.g.*, assuming that h has a Gaussian distribution. In this case we require that the constraints hold, or the objective be small, with a minimum probability (*i.e.*, reliability). Some problems of this form can also be cast as SOCPs; see, *e.g.*, [LVBL97].

2.1 Example

We now give a specific numerical example to demonstrate the discussion above. We consider a system with the step response (*i.e.*, $s(t) = h_0 + \cdots + h_t$) shown in figure 1(a). Note that the system exhibits undershoot and overshoot, has a DC or static gain of one, and settles within 30 samples. The desired trajectory y_{des} is shown in figure 1(b). It consists of a slew up to 1, a hold, followed by a slew down to 0.4, another hold, and a final slew down to 0, over a 64 sample period. We take input horizon $N = 80$ and output horizon $M = 96$.

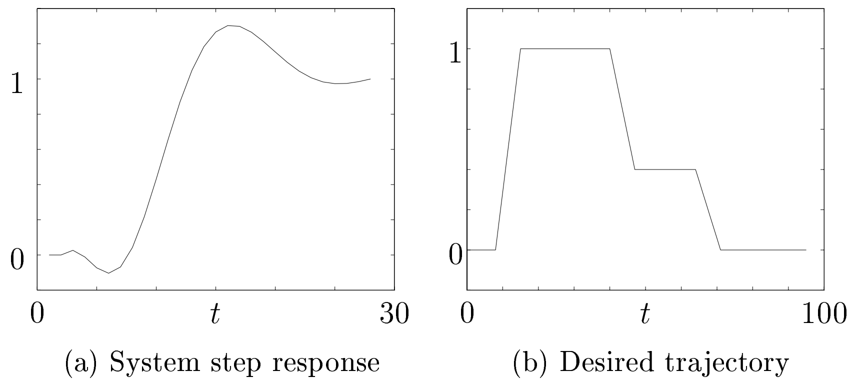


Fig. 1. System step response and desired trajectory.

We impose the input range and slew rate constraints

$$0 = U_{\text{low}} \le u(t) \le U_{\text{high}} = 1.2, \qquad |u(t+1) - u(t)| \le S = 0.5.$$

By solving the LP (6) we obtain the optimal input u_{opt} (and resulting output y_{opt}) shown in figure 2. This input results in peak tracking error 0.019. The input range and slew rate constraints on the input are both active.

Even for this simple small problem the results are impressive, and could not be obtained by any other method, for example some analytical or closed-form solution from control theory. The input u_{opt} evidently has the right form, *i.e.*, it approximately tracks y_{des} with a 15 or so sample delay; but the details (*e.g.*, the oscillations for the first 20 samples) are hardly obvious.

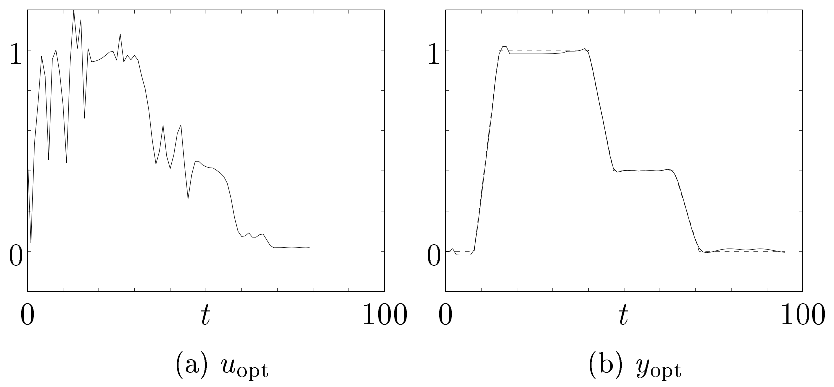


Fig. 2. Optimal control results

We now turn to robust optimal control. We take the impulse response considered above as $\overline{h}$, and the same y_{des}, horizons M and N, and input range and slew limits. The uncertainty is given by an $F \in \mathbf{R}^{28\times 2}$. This means that the impulse response is known perfectly in 26 dimensions, and varies or is uncertain only in a 2-dimensional plane (spanned by the columns of F). To give a feel for the various possible $h \in \mathcal{H}$, we plot the step response associated with various $h \in \mathcal{H}$, in figure 3. The plot shows that the uncertainty is fairly localized in time, concentrated around $t = 20$, and fairly high frequency. (We made no effort to choose F to be realistic; see below.)

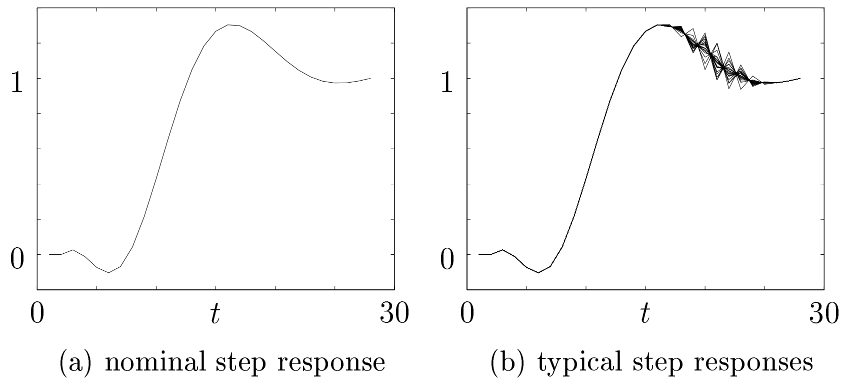
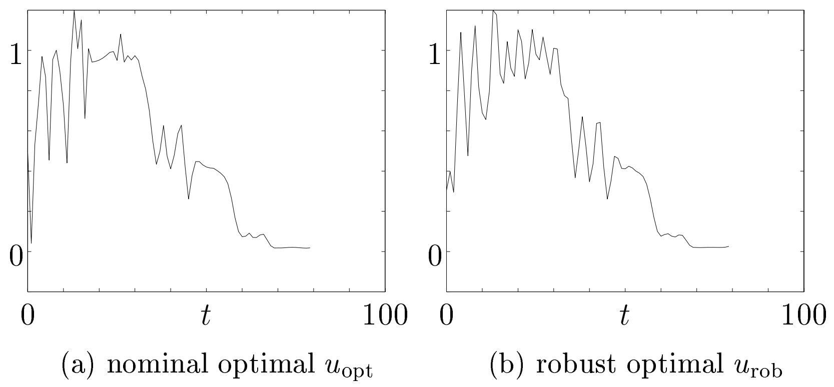


Fig. 3. System step responses

We solve the robust optimal control problem (8), by solving the SOCP (11). The resulting optimal input u_{rob} is shown in figure 4, together with the nominal optimal input u_{opt} for comparison. The inputs look quite similar, despite the fact that the nominal optimal input u_{opt} assumes perfect knowledge of the impulse response, and the robust optimal input u_{rob} works well with a set of impulse responses that lie in a 2-dimensional ellipsoid.

The performance of the two difference inputs, however, is quite different. The



(a) nominal optimal u_{opt} (b) robust optimal u_{rob}

Fig. 4. System inputs

first row of table 1 shows the peak tracking error for each of the inputs u_{opt} and u_{rob}, with the nominal impulse response $\overline{h}$. The robust optimal input u_{rob} achieves worse (larger) peak tracking error: 0.026 versus 0.019 for u_{opt}. Of course this must be the case since u_{opt} achieves the global minimum of peak tracking error; all other inputs do worse (or at least, no better). The second row of the table shows the worst case peak tracking error for the two inputs. (The two maximizing p's are different.) The nominal optimal input u_{opt} does very poorly: an impulse response in $\mathcal{H}$ causes the peak tracking error to increase more than ten-fold to 0.200. The robust optimal input, on the other hand, achieves the same worst case peak tracking error: 0.026. Thus the input u_{rob} is far more robust to the system uncertainty than is the nominal optimal u_{opt}. We note again that the two inputs do not *appear* to be very different.

	u_{opt}	u_{rob}
E (nominal plant)	0.019	0.026
E_{wc} (worst-case plant)	0.200	0.026

Table 1
Nominal and worst case peak tracking errors for u_{opt} and u_{rob}

We can continue this example by scaling the uncertainty, *i.e.*, replacing the matrix F with αF where $\alpha \geq 0$. Thus, $\alpha = 0$ means there is no uncertainty, $\alpha = 1$ means the uncertainty is the same as above, and $\alpha = 1.5$ means the uncertainty is 50% larger than in the example above. For each value of α we can solve the robust optimal control problem, and compare the worst case peak tracking error with that achieved by the nominal optimal controller. The result is plotted in figure 5. The gap between the solid curve (nominal optimal input) and the dashed curve (robust optimal input) grows with increasing uncertainty.

We close this section by admitting that in this example we specifically chose the uncertainty ellipsoid $\mathcal{H}$ to give a good gap in performance between the

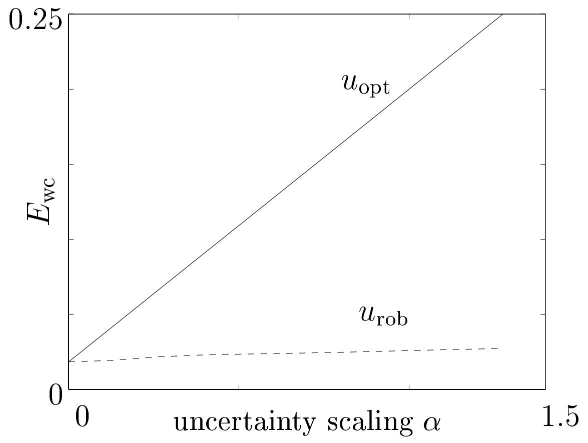


Fig. 5. Performance/robustness tradeoff

nominal and robust optimal inputs. In some cases the gap is far smaller, which means that the nominal optimal input is already relatively robust to impulse response changes. But the point here is that the robust optimal input can be computed with approximately the same effort as the nominal optimal; the difference is solving an SOCP instead of an LP. Thus, at little or no increase in computational cost, we can synthesize inputs that are automatically robust to our particular system uncertainty. This is in contrast with the normal method of regularization, where we simply limit the smoothness or bandwidth of the input signal. Indeed, the example above shows that robustness is not determined entirely by smoothness or bandwidth, since u_{opt} and u_{rob} have approximately the same smoothness and bandwidth, but very different robustness properties.

We conjecture that the use of robust optimal input signals will improve the performance of model predictive controllers, since the inputs generated at each time step would work well with an entire ellipsoid of plants. A careful analysis is quite complicated, however, since a model predictive controller is a very complex, nonlinear feedback system.

3 Synthesis of linear and affine feedback controllers

The second application of nonlinear convex optimization we consider is synthesis of linear and affine feedback controllers. To simplify the exposition we consider the *static* case, *i.e.*, the case in which all signals are vectors that do not change in time. The method discussed here does, however, extend to the dynamic case; see, *e.g.*, [BB91]. For some other references on static feedback control, see, *e.g.*, [SP96,KKB94].

The system we consider is shown in figure 6. The signal $w \in \mathbf{R}^{n_w}$ is the exogenous input, which might include command signals, disturbances, and noises. The signal $z \in \mathbf{R}^{n_z}$ is the critical output that we wish to control or regulate (whether or not we actually sense it). The signal $y \in \mathbf{R}^{n_y}$ is the sensed output, which is the signal we actually measure or sense, *i.e.*, the signal the controller has access to. The signal $u \in \mathbf{R}^{n_u}$ is the actuator input, which is the signal we are allowed to manipulate. We assume that the plant P is known; the problem is to determine or design the controller K.

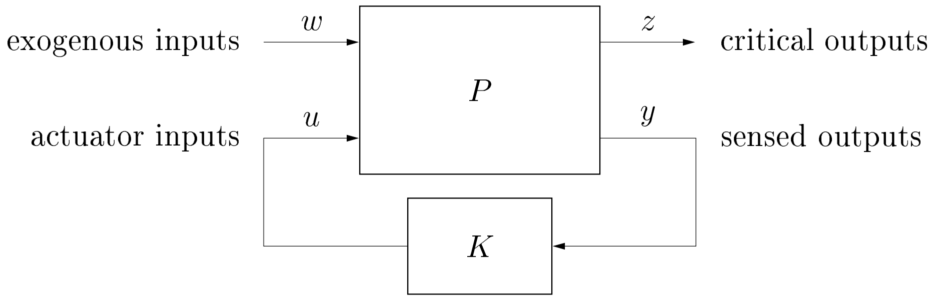


Fig. 6. Closed-loop control system

We start with the case of linear controller and plant, in which P and K are simply matrices of the appropriate dimensions, and the signals are related by

$$\begin{bmatrix} z \\ y \end{bmatrix} = \begin{bmatrix} P_{zu} & P_{zw} \\ P_{yu} & P_{yw} \end{bmatrix} \begin{bmatrix} u \\ w \end{bmatrix}, \qquad u = Ky. \tag{12}$$

To formulate a controller design problem, we need to describe the set of possible disturbance signals w, how we measure the size of z, and what limits there are on the actuator signal u. There are many ways to do this (see, *e.g.*, [BB91]); for concreteness we will take a specific, simple case. We assume that the disturbance vector w is only known to lie in a given ellipsoid

$$\mathcal{W} = \{ \, \overline{w} + Fp \mid \|p\| \le 1 \, \} \, .$$

(Here F might come from an estimated covariance matrix for w, with a given confidence level, as described in §2.)

The regulation performance of the closed-loop system will be judged by the worst case, peak size of z:

$$Z = \max_{w \in \mathcal{W}} \max_{i=1,\dots,n_z} |z_i|. \tag{13}$$

We will constrain the actuator authority, *i.e.*, the maximum peak level of the actuator signal that can be caused by w:

$$U = \max_{w \in \mathcal{W}} \max_{i=1,\dots,n_u} |u_i|. \tag{14}$$

Of course we would like to choose K so that both Z and U are small, *i.e.*, we achieve good regulation performance, using little actuator authority. In general, however, there will be a tradeoff between these two objectives. We can consider the basic controller design problem

$$\begin{array}{ll} \text{minimize} & Z \\ \text{subject to} & U \leq U_{\max}, \end{array} \tag{15}$$

where the variable is the matrix $K \in \mathbf{R}^{n_u \times n_y}$ (which determines U and Z via (13) and (14)).

We will now show how this problem can be cast as an SOCP, using a clever change of variables and a few other transformations. The first step is to express the signals z and u directly in terms of K. Eliminating y from (12) yields $z = Hw$, where

$$H = P_{zw} + P_{zu}K(I - P_{yu}K)^{-1}P_{yw}.$$

The matrix H shows how z depends on w; note that H depends on K is a very complex way. We assume here that $I - P_{yu}K$ is invertible. This assumption is not important; it can be circumvented, for example, by slightly perturbing K (see, *e.g.*, [Vid85,BB91]). In a similar way, we can express u as $u = Gw$, where

$$G = K(I - P_{yu}K)^{-1}P_{yw}.$$

We can express the components of u and z as

$$z_i = h_i^T(\overline{w} + Fp), \qquad u_i = g_i^T(\overline{w} + Fp)$$

where $\|p\| \leq 1$ parameterizes w, and h_i^T (g_i^T) is the ith row of H (G).

Now we can derive explicit expressions for U and Z in terms of G and H:

$$U = \max_{i=1,\ldots,n_u} \max_{\|p\|\leq 1} |g_i^T(\overline{w} + Fp)| = \max_{i=1,\ldots,n_u} \left(|g_i^T\overline{w}| + \|F^T g_i\| \right),$$

and similarly,

$$Z = \max_{i=1,\ldots,n_u} \left(|h_i^T\overline{w}| + \|F^T h_i\| \right).$$

(In deriving these expressions we use the fact that $p^T c$ ranges between $\pm\|c\|$ for $\|p\| \leq 1$; see §2.)

Thus, our basic controller design problem (15) can be expressed as

$$\begin{array}{ll} \text{minimize} & \gamma \\ \text{subject to} & |h_i^T\overline{w}| + \|F^T h_i\| \leq \gamma, \quad i = 1, \ldots, n_z \\ & |g_i^T\overline{w}| + \|F^T g_i\| \leq U_{\max}, \quad i = 1, \ldots, n_u, \end{array} \tag{16}$$

where the variables are the matrix K and the scalar γ.

The problem (16), unfortunately, is a complicated nonlinear optimization problem. Far worse than its nonlinearity is the fact that it is not convex. By a clever change of variables, however, we can convert the problem into an equivalent, but convex problem. We define a new matrix

$$Q = K(I - P_{yu}K)^{-1} \in \mathbf{R}^{n_u \times n_y} \tag{17}$$

so that

$$H = P_{zw} + P_{zu}QP_{yw}, \qquad G = QP_{yw}. \tag{18}$$

It is important to note that G is a linear function of the matrix Q, and H is an affine function, *i.e.*, a linear function plus a constant. This is in contrast to the complicated nonlinear dependence on K of the matrices G and H.

Provided the matrix $I + QP_{yu}$ is invertible, we can invert this transformation to get K from Q as

$$K = (I + QP_{yu})^{-1}Q. \tag{19}$$

Here we will simply ignore the possibility that $I + QP_{yu}$ is singular. In fact, the case of singular $I + QP_{yu}$ causes no problem; it corresponds to integral control. The details (as well as the extension to the dynamic case) can be found in [BB91]. Ignoring these important details, we can view (17) and (19) as inverses, that relate each K with a particular Q and vice versa.

We will now take Q as our design variable; once we have found an optimal Q we can recover the actual controller K via (19). The basic controller design problem, in terms of the variable Q, becomes

$$\begin{array}{ll} \text{minimize} & \gamma \\ \text{subject to} & |h_i(Q)^T\overline{w}| + \|F^T h_i(Q)\| \leq \gamma, \quad i = 1, \ldots, n_z \\ & |g_i(Q)^T\overline{w}| + \|F^T g_i(Q)\| \leq U_{\max}, \quad i = 1, \ldots, n_u, \end{array} \tag{20}$$

where the optimization variables are the matrix Q and the scalar γ. Since h_i and g_i are both affine functions of Q (from (18)), the problem (20) is in fact an SOCP.

The practical ramification is that we can efficiently solve the controller design problem (15) via an interior-point method for SOCP. For a problem with, say, 20 sensors and 10 actuators, we can efficiently optimize 200 gains (*i.e.*, the entries of K). By repeatedly solving the problem for different values of allowed actuator authority $U_{\max}$, we can compute the tradeoff curve relating the (globally optimal) value of performance with actuator authority. We will see examples of this in §3.1.

The idea of using Q as the variable instead of K has been discovered many times, and therefore has several names. The modern version is attributed to Youla (hence called Youla parameter design or Q-parametrization), but simpler versions of the trick were known and used in the 1940s; see [BB91].

We now turn to an interesting and useful extension of the linear feedback controller design problem, *i.e.*, the affine controller design problem. The setup is the same as for linear controller design, except we assume that the plant and controller are affine, *i.e.*, are linear plus a constant. For example, the controller is given by $u = Ky + a$, where $a \in \mathbf{R}^{n_u}$. To specify the controller we must give the feedback gain matrix K as well as the offset or set-point vector a.

In fact, this general scheme with affine plant and controller can be handled directly by the linear controller design method described above: we simply add a new component to w, which is fixed equal to one, and make this same (constant) signal available to the controller as another component of y. When we design a linear controller for this expanded setup, it is the same as an affine controller for the original setup. In terms of classical control, an affine controller corresponds to a linear feedback controller with a (nonzero) set-point. The discussion above shows that we can jointly optimize the set-points and feedback gains by solving an SOCP.

3.1 Example

In this section we demonstrate the ideas above with a plant with 10 critical signals ($n_z = 10$), 5 actuators ($n_u = 5$), 10 sensors ($n_y = 10$), and 10 disturbances ($n_w = 10$). The problem data, *i.e.*, the entries of P, F, $\overline{w}$, and the plant offset vector, were chosen randomly (from a unit normal Gaussian distribution, but it doesn't matter); there was no attempt to make the plant realistic. For this example the controller is characterized by its feedback gain matrix K, which is a 5×10 matrix, and the controller offset a, which is a vector with dimension 5.

We then solved the basic controller design problem (15) for various values of $U_{\max}$, in order to find the optimal tradeoff curve between performance and actuator authority. Three versions of this problem were solved:

- *Set-point only.* We fix the feedback gains to be zero, and simply choose a, the controller offset. Thus the input has the form $u = a \in \mathbf{R}^5$. This case is the classical open-loop controller.
- *Feedback gains only.* Here we fix the controller to have the form $u = K(y - \overline{y})$,

where $\overline{y}$ is the sensor output resulting from $w = \overline{w}$ and $u = 0$. This is the classical linear feedback controller for an affine plant.

- *Joint set-point and feedback.* Here we design a general affine controller; the input has the form $u = Ky + a$. The total number of variables is 55: 50 feedback gains (K) and 5 offsets or set-points (a). As far as we know, joint design of set-point and feedback gains is not done in classical control.

The optimal tradeoff curves for these three cases are shown in figure 7.

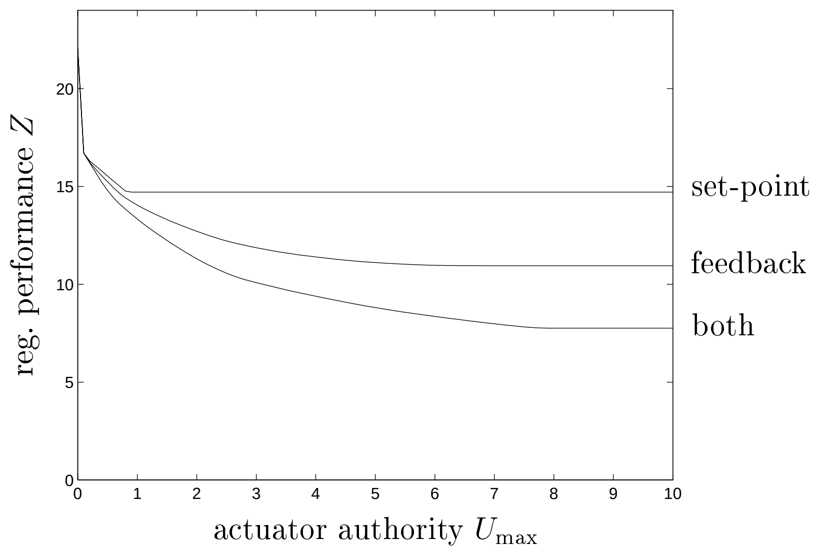


Fig. 7. Optimal tradeoff curves between Z and U for three control strategies: set-point only, feedback only, and joint set-point & feedback.

Since the joint set-point and feedback design is more general than the other two, its tradeoff curve lies below them. In this particular example, the feedback controller always outperforms the set-point only controller, but this need not happen in general. The plots are very useful: we can see that for small allowed actuator authority (say, $U_{\max} \leq 1$) the set-point controller works almost as well as the other two; roughly speaking there simply isn't enough actuator authority to get the benefits of feedback. When the allowed actuator authority reaches a higher level (around 5) the benefits of the feedback only strategy reaches its maximum, but the joint set-point and feedback strategy continues to derive benefit (*i.e.*, reduce Z) with increasing actuator authority.

It is not surprising that the feedback controllers outperform the simple set-point only controller. But this example clearly shows that *joint* design of set-point *and* feedback gains can substantially outperform the classical feedback design.

4 Control system analysis and synthesis via linear matrix inequalities

In this section we describe two examples of control system analysis and synthesis problems that can be cast as semidefinite programs (SDPs), and hence, efficiently solved by new interior-point methods. The importance of LMIs in control has been appreciated since Lur'e and Postinikov's work in the Soviet Union in the 1940s. In the 1960s, Yakubovich, Kalman, and others developed frequency domain and graphical methods to solve some special classes of the resulting LMIs or SDPs. In the 1980s it was observed that the problems could be solved in general, by numerical algorithms such as the ellipsoid method, cutting-plane methods, or subgradient methods. But the real increase in utility (and interest) came when interior-point methods for SDP made it practical to solve the resulting LMI problems with reasonable efficiency. A survey and history of these methods (as well as descriptions of many other problems) can be found in the monograph [BEFB94], which has a bibliography up to 1994. There has been much work done since then, *e.g.*, [BP94,AG95,Gah96].

We first consider a linear time-varying system described by

$$\dot{x}(t) = A(t)x(t), \quad x(0) = x_0 \tag{21}$$

where $A(t)$ can take on the (known) values $A_1, \ldots, A_L$:

$$A(t) \in \{A_1, \ldots, A_L\},$$

but otherwise is arbitrary. Thus, (21) defines a (large) family of trajectories (which depend on how $A(t)$ switches between allowed values).

We are interested in the quadratic performance index

$$J = \int_0^\infty x(t)^T Q x(t)\, dt$$

where Q is symmetric and positive semidefinite. The performance index J, of course, depends on which trajectory of (21) we use. Therefore we will attempt to find or bound the worst case value:

$$J_{\mathrm{wc}} = \max \int_0^\infty x(t)^T Q x(t)\, dt$$

where the maximum (supremum) is taken over all trajectories of (21). Even the question whether $J_{\mathrm{wc}} < \infty$, *i.e.*, whether the system is stable, is interesting (and challenging; it is NP-hard [BT97]).

Before proceeding let us mention one reason why we might be interested in the linear time-varying system described above. Suppose we have a nonlinear, time-varying system

$$\dot{z} = f(z,t), \qquad z(0) = z_0,$$

where $f(0,t) = 0$ and for all v,

$$\nabla f(v) \in \mathbf{Co}\{A_1, \ldots, A_L\},$$

where **Co** denotes convex hull. Then J_{wc} evaluated for the linear time-varying system gives an upper bound on J_{wc} for the nonlinear system, *i.e.*, the results we derive here hold for the nonlinear system. This idea is called global linearization; see, *e.g.*, [BEFB94, §4.3]

We will use a quadratic Lyapunov function x^TPx to establish a bound on J_{wc}. Suppose for some $P = P^T \succ 0$ we have

$$\frac{d}{dt}\left(x(t)^TPx(t)\right) \le -x(t)^TQx(t) \tag{22}$$

for all trajectories and all t (except switching times). Integrate this inequality from $t = 0$ to $t = T$ to get

$$x(T)^TPx(T) - x(0)^TPx(0) \le -\int_0^T x(t)^TQx(t)\, dt.$$

Noting that $x(T)^TPx(T) \ge 0$ and rearranging yields

$$\int_0^T x(t)^TQx(t)\, dt \le x_0^TPx_0.$$

Since this holds for all T we have

$$J = \int_0^\infty x(t)^TQx(t)\, dt \le x_0^TPx_0.$$

This inequality holds for all trajectories, so we have

$$J_{\mathrm{wc}} \le x_0^TPx_0, \tag{23}$$

for any $P \succ 0$ that satisfies (22).

Now we address the question of how to find a $P \succ 0$ that satisfies (22). Since

$$\frac{d}{dt}\left(x(t)^TPx(t)\right) = x(t)^T\left(A(t)^TP + PA(t)\right)x(t),$$

the condition (22) can be shown to be equivalent to

$$A_i^TP + PA_i + Q \preceq 0, \quad i = 1, \ldots, L,$$

which is a set of LMIs in P. (We skip some details here, such as what happens at the switching times; see, *e.g.*, [BEFB94].) Thus, finding a P that provides a bound on J_{wc} can be done by solving an LMI feasibility problem:

$$\begin{array}{ll} \text{find} & P = P^T \\ \text{that satisfies} & P \succ 0, \quad A_i^T P + P A_i + Q \preceq 0, \quad i = 1, \ldots, L. \end{array}$$

Finally, we can optimize over the Lyapunov function P by searching for the best (*i.e.*, smallest) bound of the form (23) over P:

$$\begin{array}{ll} \text{minimize} & x_0^T P x_0 \\ \text{subject to} & P \succ 0, \quad A_i^T P + P A_i + Q \preceq 0, \quad i = 1, \ldots, L. \end{array} \qquad (24)$$

Since $x_0^T P x_0$ is a linear function of the variable P, this problem is an SDP in the variable $P = P^T \in \mathbf{R}^{n \times n}$, and is readily solved using interior-point methods.

This approach seems very simple and straightforward, but is more powerful than many other methods and approaches, *e.g.*, those based on the circle criterion, small-gain methods, etc. The reason is simple: any method that produces, in the end, a quadratic Lyapunov function to establish its bound, is, almost by definition, no more powerful than the simple method of solving (24), which finds the *globally optimal* quadratic Lyapunov function.

The example described above shows how SDP can be used in control system *analysis*. We will now show how a similar method can be used for a controller *synthesis* problems. We consider a time-varying linear system, with input u, given by

$$\dot{x} = A(t)x(t) + B(t)u(t), \quad x(0) = x_0, \qquad (25)$$

where

$$[A(t)\ B(t)] \in \{[A_1\ B_1], \ldots, [A_L\ B_L]\}.$$

We will use a (constant, linear) state feedback gain K, *i.e.*, $u = Kx$, so the closed-loop system is described by

$$\dot{x}(t) = (A(t) + B(t)K)x(t), \quad x(0) = x_0.$$

The cost we consider is

$$J = \int_0^\infty \left(x(t)^T Q x(t) + u(t)^T R u(t) \right) \, dt$$

where Q and R are positive definite. This cost depends on the particular trajectory we take, so as above we consider the worst case cost

$$\begin{aligned} J_{\mathrm{wc}} &= \max \int_0^\infty \left(x(t)^T Q x(t) + u(t)^T R u(t) \right) \, dt \\ &= \max \int_0^\infty x(t)^T (Q + K^T R K) x(t) \, dt \end{aligned}$$

where the maximum is over all possible trajectories.

We can now pose the following problem: find a state feedback gain matrix K and a quadratic Lyapunov function P that minimizes the bound $x_0^T P x_0$ on J_{wc} (for the closed-loop system). Note the interesting twist compared to the standard approach. Here we are *simultaneously* doing the synthesis (*i.e.*, designing K) and the analysis (*i.e.*, finding a Lyapunov function that gives a guaranteed performance bound). In the standard approach, K is first designed, and then P is sought.

We can pose this problem as

$$\begin{array}{ll} \text{minimize} & x_0^T P x_0 \\ \text{subject to} & (A_i + B_i K)^T P + P(A_i + B_i K) + Q + K^T R K \preceq 0, \quad i = 1, \ldots, L, \\ & P \succ 0, \end{array} \tag{26}$$

where the variables are the state feedback gain matrix K and the Lyapunov matrix P. Unfortunately this problem is not an SDP since the constraints involve a quadratic term in K and products between the variables P and K.

We will now show how a change of variables and a few other transformations allow us to reformulate the problem (26) as an SDP (and hence, solve it efficiently). Define new matrices Y and W as

$$Y = P^{-1}, \qquad W = K P^{-1},$$

so that (since $P \succ 0$, and hence $Y \succ 0$) we have

$$P = Y^{-1}, \qquad K = W Y^{-1}.$$

We will use the variables Y and W instead of P and K. The inequality above becomes

$$(A_i + B_i W Y^{-1})^T Y^{-1} + Y^{-1}(A_i + B_i W Y^{-1}) + Q + K^T R K \preceq 0. \tag{27}$$

At first, this looks worse than the original inequality, since it not only involves products of the variables, but also an inverse (of Y). But by applying a congruence by Y, *i.e.*, multiplying the left and the right hand sides by Y, we obtain the following equivalent inequality:

$$YA_i^T + W^TB_i^T + A_iY + B_iW + YQY + W^TRW \preceq 0,$$

which can be written as

$$YA_i^T + W^TB_i^T + A_iY + B_iW + \begin{bmatrix} Y \\ W \end{bmatrix}^T \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix} \begin{bmatrix} Y \\ W \end{bmatrix} \preceq 0.$$

This *quadratic* matrix inequality can, in turn, be expressed as the *linear* matrix inequality

$$L_i(Y,W) = \begin{bmatrix} -YA_i^T - W^TB_i^T - A_iY - B_iW & Y & W^T \\ Y & Q^{-1} & 0 \\ W & 0 & R^{-1} \end{bmatrix} \succeq 0. \qquad (28)$$

(The equivalence is based on the *Schur complement*; see, *e.g.*, [VB96,BEFB94].) To summarize, we have transformed the original (nonconvex) matrix inequality (27), with variables P and K, into the linear matrix inequality (28). (Here we assume that Q is invertible. If Q is singular another more complicated method can be used to derive an equivalent LMI.)

In a similar way we can express $x_0^TPx_0 = x_0^TY^{-1}x_0 \leq \gamma$ as the LMI

$$\begin{bmatrix} \gamma & x_0^T \\ x_0 & Y \end{bmatrix} \succeq 0.$$

Finally, then, we can solve the original (nonconvex) problem (26) by solving

$$\begin{array}{ll} \text{minimize} & \gamma \\ \text{subject to} & L_i(Y,W) \succeq 0, \quad i = 1,\ldots,L, \\ & \begin{bmatrix} \gamma & x_0^T \\ x_0 & Y \end{bmatrix} \succeq 0, \end{array} \qquad (29)$$

which is an SDP.

In this example, we use SDP to synthesize a state feedback gain matrix and, simultaneously, a quadratic Lyapunov function that establishes a guaranteed performance bound for a time-varying system, or, by global linearization, a nonlinear system.

4.1 Example

We illustrate the method described above with a specific numerical example. We consider the differential equation

$$y''' + a_1(t)y'' + a_2(t)y' + a_3(t)y = a_4(t)u$$

with initial condition $y''(0) = y'(0) = y(0) = 1$. We can represent this differential equation as the linear time-varying system

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -a_3(t) & -a_2(t) & -a_1(t) \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 0 \\ a_4(t) \end{bmatrix} u(t) = A(t)x(t) + B(t)u(t),$$

where $x(t) = [\ y(t)\ y'(t)\ y''(t)\ ]^T$. Each coefficient $a_i(t)$ can take on the values 1 or 4, *i.e.*, $a_i(t) \in \{1, 4\}$. The system can be described in the form (25) by taking the 16 extreme values of $a_1, \ldots, a_4$.

The cost we consider is

$$\begin{aligned} J &= \int_0^\infty \left(u(t)^2 + y(t)^2 + y'(t)^2 + y''(t)^2 \right)\, dt \\ &= \int_0^\infty \left(x(t)^T Q x(t) + u(t)^T R u(t) \right)\, dt, \end{aligned}$$

i.e., $Q = I$ and $R = 1$.

We will compare two state feedback gain matrices. The matrix $K_{\rm lqr} \in \mathbf{R}^{1\times 3}$ is chosen to minimize the cost when the coefficients all have the value $a_i = 2.5$, the mean of the high and low values. ($K_{\rm lqr}$ can be found by solving a Riccati equation, for example.) We compute a robust state feedback gain matrix $K_{\rm rob}$ by solving the SDP (29).

The results are summarized in table 2, which shows two costs for each of the state feedback gain matrices $K_{\rm lqr}$ and $K_{\rm rob}$. One cost is the nominal cost $J_{\rm nom}$, which is the cost with the coefficients fixed at the nominal value $a_i = 2.5$. The other cost we show is the worst case cost $J_{\rm wc}$, with the coefficients varying.

The first column shows that when the coefficients are fixed at 2.5, the LQR controller does a bit better than the robust controller (as, of course, it must).

	$\sqrt{J_{\mathrm{nom}}}$	$\sqrt{J_{\mathrm{wc}}}$
K_{lqr}	3.04	∞
K_{rob}	3.93	≤ 9.75

Table 2
Nominal and robust performance of LQR and robust state feedback gains.

The second column needs some explanation. It turns out that the LQR controller yields an unstable closed-loop system for one of the sixteen extreme values of coefficients, so the upper right entry is infinite. On the other hand for the robust controller we have the upper bound 9.75 from solving the SDP (29) (and indeed, we have the Lyapunov function P that proves it). This bound holds for any variation of the coefficients among the 16 vertices. In fact, we do not know what $\sqrt{J_{\mathrm{wc}}}$ is: we only know that it is less than 9.75, and more than 3.93.

So here we have used SDP to synthesize a controller that not only stabilizes a time-varying system (with substantial coefficient variation), but also, simultaneously, a quadratic Lyapunov function that gives a guaranteeed performance bound.

5 Conclusions

In the last five years or so, interior-point methods originally designed for LP have been extended to handle nonlinear convex optimization problems such as SDP and SOCP. Our new ability to efficiently solve such problems has had an impact on several fields, *e.g.*, combinatorial optimization and control. It allows us to more accurately capture real engineering objectives and specifications in an optimization formulation, *e.g.*, by incorporating robustness constraints into our design. Indeed, we can incorporate robustness constraints that accurately reflect our uncertainty in the system.

The new interior-point methods for nonlinear convex optimization also allow us to solve control problems that have been known, but were thought to be difficult or intractable except in special cases with analytic solutions. For example, many control problems involving LMIs were known in the 1960s, but only a small number of such problems could be solved (by graphical, analytical, or Riccati equation methods). The development of efficient interior-point methods for SDP allows us to solve a much wider class of problems.

In this paper we have described three simple examples, which were meant only to be representative samples of the many possible new applications of nonlinear convex programming in control. While the examples we have described are small and simple (involving only a few tens of variables and constraints) standard interior-point methods can easily handle the much larger and more complex problems that would arise in practice (with, say, many hundreds of variables and constraints). Custom interior-point methods, that exploit the special structure of the problems, can handle far larger problems still.

References

[AG95] P. Apkarian and P. Gahinet. A Convex Characteristization of Gain-Scheduled $\mathcal{H}_\infty$ Controllers. *IEEE Trans. Aut. Control*, AC-40(5):853–864, 1995.

[AHN$^+$97] 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.

[AHNO97] F. Alizadeh, J. P. Haeberly, M. V. Nayakkankuppam, and M. L. Overton. SDPPACK *User's Guide, Version 0.8 Beta.* NYU, June 1997.

[AHO94] 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.

[Ali95] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5(1):13–51, February 1995.

[AP92] J. C. Allwright and G. C. Papavasiliou. On linear programming and robust model-predictive control using impulse-responses. *Systems & Control Letters*, pages 159–164, 1992.

[BB91] S. Boyd and C. Barratt. *Linear Controller Design: Limits of Performance.* Prentice-Hall, 1991.

[BEFB94] 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.

[Bor97] B. Borchers. CSDP*, a C library for semidefinite programming.* New Mexico Tech, March 1997.

[BP94] G. Becker and A. Packard. Robust performance of linear parametrically varying systems using parametrically-dependent linear feedback. *Syst. Control Letters*, 1994.

[BT97] V. D. Blondel and J. N. Tsitsiklis. Complexity of elementary hybrid systems. In *Proc. European Control Conf.*, July 1997.

[BTN96] A. Ben-Tal and A. Nemirovski. Robust convex optimization. Technical report, Faculty of Industrial Engineering and Management, Technion, December 1996.

[BVG94] S. Boyd, L. Vandenberghe, and M. Grant. Efficient convex optimization for engineering design. In *Proceedings IFAC Symposium on Robust Control Design*, pages 14–23, September 1994.

[CM87] P. J. Campo and M. Morari. Robust model predictive control. In *Proc. American Control Conf.*, volume 2, pages 1021–1026, San Francisco, 1987.

[CMW97] J. Czyzyk, S. Mehrotra, and S. J. Wright. *PCx User Guide.* Optimization Technology Cneter, March 1997.

[Dan63] G. B. Dantzig. *Linear Programming and Extensions.* Princeton University Press, 1963.

[EL96] L. El Ghaoui and H. Lebret. Robust least squares and applications. In *Proc. IEEE Conf. on Decision and Control*, pages 249–254, December 1996.

[FK95] 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.

[GA94] P. Gahinet and P. Apkarian. A Linear Matrix Inequality Approach to $\mathcal{H}_\infty$ Control. *Int. J. Robust and Nonlinear Control*, 4:421–488, 1994.

[Gah96] P. Gahinet. Explicit Controller Formulas for LMI-based $\mathcal{H}_\infty$ Synthesis. *Automatica*, 32(7):1007–1014, 1996. Also appear in *Proc. IEEE Conf. on Decision and Control*, 1994.

[GN93] P. Gahinet and A. Nemirovskii. *LMI Lab: A Package for Manipulating and Solving LMIs.* INRIA, 1993.

[Kar84] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984.

[KKB94] M. G. Kabuli, R. L. Kosut, and S. Boyd. Improving static performance robustness of thermal processes. In *Proceedings of the 33rd IEEE Conference on Decision and Control*, pages 62–66, Orlando, Florida, 1994.

[LVB97] M. S. Lobo, L. Vandenberghe, and S. Boyd. SOCP*: Software for Second-Order Cone Programming.* Information Systems Laboratory, Stanford University, 1997.

[LVBL97] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Second-order cone programming: interior-point methods and engineering applications. *Linear Algebra and Appl.*, 1997. Submitted.

[ML97] M. Morari and J. H. Lee. Model predictive control: Past, present and future. In *6th International Symposium on Process Systems Engineering*, 1997.

[MR97] A. Megretski and A. Rantzer. System analysis via integral quadratic constraints. *IEEE Trans. Aut. Control*, 42(6):819–830, June 1997.

[NN94] Yu. Nesterov and A. Nemirovsky. *Interior-point polynomial methods in convex programming*, volume 13 of *Studies in Applied Mathematics.* SIAM, Philadelphia, PA, 1994.

[OW97] M. Overton and H. Wolkowicz, editors. *Semidefinite Programming*, Mathematical Programming, Series B, May 1997.

[Ric93] J. Richalet. Industrial applications of model based predictive control. *Automatica*, 29(5):1251–1274, 1993.

[SP96] R. Smith and A. Packard. Optimal control of perturbed linear static systems. *IEEE Transactions on Automatic Control*, 41(4):579–584, 1996.

[Van92] 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.

[Van97] R. J. Vanderbei. *Linear Programming: Foundations and Extensions.* Kluwer Academic Publishers, 1997.

[VB94] 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`.

[VB95] L. Vandenberghe and S. Boyd. A primal-dual potential reduction method for problems involving matrix inequalities. *Mathematical Programming*, 69(1):205–236, July 1995.

[VB96] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, March 1996.

[VBW98] L. Vandenberghe, S. Boyd, and S.-P. Wu. Determinant maximization with linear matrix inequality constraints. *SIAM J. on Matrix Analysis and Applications*, April 1998. To appear.

[Vid85] M. Vidyasagar. *Control System Synthesis: A Factorization Approach.* MIT Press, 1985.

[WB96] 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.

[Wri93] S. J. Wright. Interior-point methods for optimal control of discrete-time system. *J. Optim. Theory Appls 77*, pages 161–187, 1993.

[Wri96] S. J. Wright. Applying new optimization algorithms to model predictive control. 1996.

[Wri97] S. J. Wright. *Primal-Dual Interior-Point Methods.* SIAM, Philadelphia, 1997.

[Zha94] 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.

[ZM93] Z. Q. Zheng and M. Morari. Robust stability of constrained model predictive control. In *Proc. American Control Conf.*, pages 379–383, June 1993.

[ZW62] L. A. Zadeh and B. H. Whalen. On optimal control and linear programming. *IRE Trans. Aut. Control*, pages 45–46, July 1962.