Robust linear programming and optimal control

Lieven Vandenberghe[*] Stephen Boyd[†] Mehrdad Nouralishahi[‡]

Abstract

We describe an efficient method for solving an optimal control problem that arises in robust model-predictive control. The problem is to design the input sequence that minimizes the peak tracking error between the ouput of a linear dynamical system and a desired target output, subject to inequality constraints on the inputs. The system is uncertain, with an impulse response that can take arbitrary values in a given polyhedral set. The method is based on Mehrotra's interior-point method for linear programming, and takes advantage of the problem structure to achieve a complexity that grows linearly with the control horizon, and increases as a cubic polynomial as a function of the system order, the number of inputs, and the number of uncertainty parameters.

[*]Corresponding author. UCLA Department of Electrical Engineering, 68-119 Engineering IV Building, Los Angeles, CA 90095-1594. FAX: (310) 206-4685. E-mail: vandenbe@ee.ucla.edu.

[†]Department of Electrical Engineering, Stanford University (boyd@stanford.edu).

[‡]Department of Electrical Engineering, University of California Los Angeles (mehrdad@ee.ucla.edu).

1 Introduction

We describe an efficient method for solving the optimal control problem

$$\begin{array}{ll} \text{minimize} & \sup_{\|\rho\|_\infty \le 1} \max_{t=1,\ldots,N} |(c_0 + \sum_{i=1}^p \rho_i c_i)^T x(t) - y_{\text{des}}(t)| \\ \text{subject to} & x(1) = Ax_0 + Bu(0) \\ & x(t+1) = Ax(t) + Bu(t), \quad t = 1,\ldots,M-1 \\ & x(t+1) = Ax(t), \quad t = M,\ldots,N-1 \\ & -\mathbf{1} \preceq u(t) \preceq \mathbf{1}, \quad t = 0,\ldots,M-1. \end{array} \tag{1}$$

The problem data are $A \in \mathbf{R}^{n\times n}$, $B \in \mathbf{R}^{n\times m}$, $x_0 \in \mathbf{R}^n$, the vectors $c_k \in \mathbf{R}^n$, $k = 0,\ldots,p$, and the sequence $y_{\text{des}}(t)$, $t = 1,\ldots,N$. The optimization variables are $u(0), \ldots, u(M-1) \in \mathbf{R}^m$, and $x(1), \ldots, x(N) \in \mathbf{R}^n$ $(N > M)$, where $u(t)$ and $x(t)$ are the input and the state of a discrete-time linear dynamical system

$$x(t+1) = Ax(t) + Bu(t), \quad x(0) = x_0.$$

The constraints also include componentwise upper and lower bounds on the inputs. To motivate the objective function, we first consider the special case with $p = 0$, *i.e.*, the problem

$$\begin{array}{ll} \text{minimize} & \max_{t=1,\ldots,N} |c_0^T x(t) - y_{\text{des}}(t)| \\ \text{subject to} & x(1) = Ax_0 + Bu(0) \\ & x(t+1) = Ax(t) + Bu(t), \quad t = 1,\ldots,M-1 \\ & x(t+1) = Ax(t), \quad t = M,\ldots,N-1 \\ & -\mathbf{1} \preceq u(t) \preceq \mathbf{1}, \quad t = 0,\ldots,M-1. \end{array} \tag{2}$$

We interpret $c_0^T x(t)$ as the output of the system at time t, and $y_{\text{des}}(t)$ as a given desired output, that we want to follow as closely as possible. The problem is to find the input sequence $u(0), \ldots, u(M)$ that minimizes the peak tracking error $\max_t |c_0^T x(t) - y_{\text{des}}(t)|$, subject to the constraints $-\mathbf{1} \preceq u(t) \preceq \mathbf{1}$. We will refer to problem (2) as the *output tracking problem.*

Problem (1) is an extension of the output tracking problem in which we include uncertainty in the system parameters. More specifically, we assume that the system output is given by $y(t) = c(\rho)^T x(t)$, where the vector $c(\rho) = c_0 + \sum_{i=1}^p \rho_i c_i$ is an affine function of some parameter $\rho \in \mathbf{R}^p$, which is unknown but bounded, with components between -1 and 1. Alternatively, we can say that the impulse response $h(1), h(2), \ldots$ of the system can take arbitrary values in the polyhedron

$$\{(h(1), h(2), h(3), \ldots) \mid h(t) = h_0(t) + \rho_1 h_1(t) + \cdots + \rho_p h_p(t), \ \|\rho\|_\infty \le 1\}, \tag{3}$$

where $h_k(t)$ is defined as $h_k(t) = c_k^T A^{t-1} B$. In problem (1) we minimize the *worst-case* peak tracking error, considering all possible values of ρ. We therefore refer to the problem as the *robust output tracking problem.*

Example

Figures 1–3 show an example that will illustrate the problem. We consider an FIR system of order $n = 60$, with an impulse reponse that can take arbitrary values in a polyhedral set

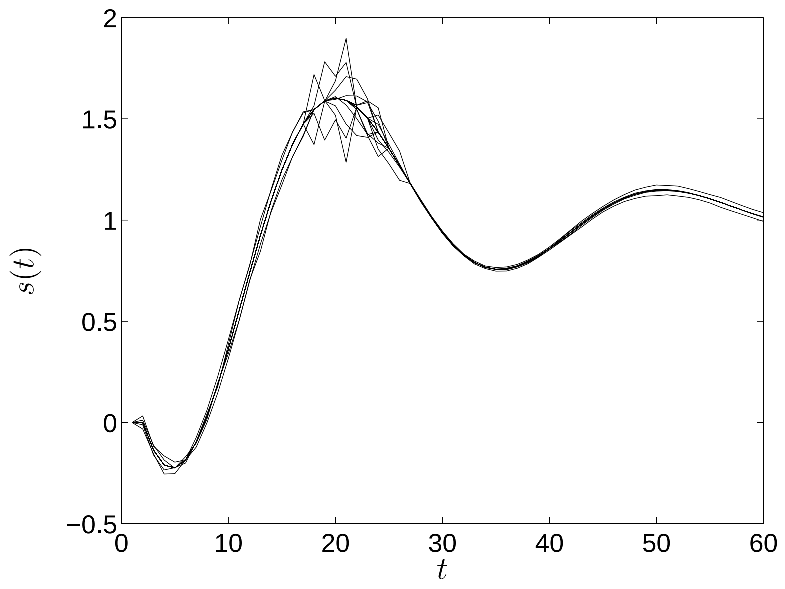

Figure 1: Stepresponses of an uncertain FIR system of order $n = 60$.

of the form (3) with $p = 10$ uncertain parameters. Figure 1 shows the step responses for a few systems in this set.

The dashed lines in figures 2 and 3 show the desired output sequence y_{des}. The solid lines show the worst-case outputs resulting from two input sequences. In the first figure we apply the input sequence u_{nom} that would be optimal if we ignore the uncertainty in the system, *i.e.*, if we solve (2). The peak tracking error between the desired output and the worst-case output is 0.99. In figure 3 we consider the input u_{rob} computed by solving the robust output tracking problem (1). Here the peak tracking error between the worst-case output and the desired output is 0.27.

Algorithms

The robust and nonrobust output tracking problems (1) and (2) are convex optimization problems: the equality and inequality constraints are linear in the variables $x(t)$, $u(t)$; the objective functions are nondifferentiable but convex in the variables $x(t)$. In fact both problems are readily cast as linear programming problems (LPs); see §3.1 and §4.1. We can therefore use general-purpose LP solvers to compute the global optima of both problems. However the computational cost of using a general-purpose solver is quite high. We will see that the LP formulation of the nonrobust problem (2) involves $Nn + Mm + 1$ variables, $2(Nn + Mm)$ inequality constraints, and Nn equality constraints, while the LP formulation of the robust problem (1) involves $N(n+p)+Mm+1$ variables, $2(N(n+p)+Mm)$ inequalities, and Nn equality constraints. We can therefore expect that the computational effort strongly depends on the control horizons N and M, and in particular, that the cost of solving the robust problem is much higher than the cost of solving the nonrobust problem. The main purpose of this paper is to describe an efficient special-purpose algorithm that takes advan-

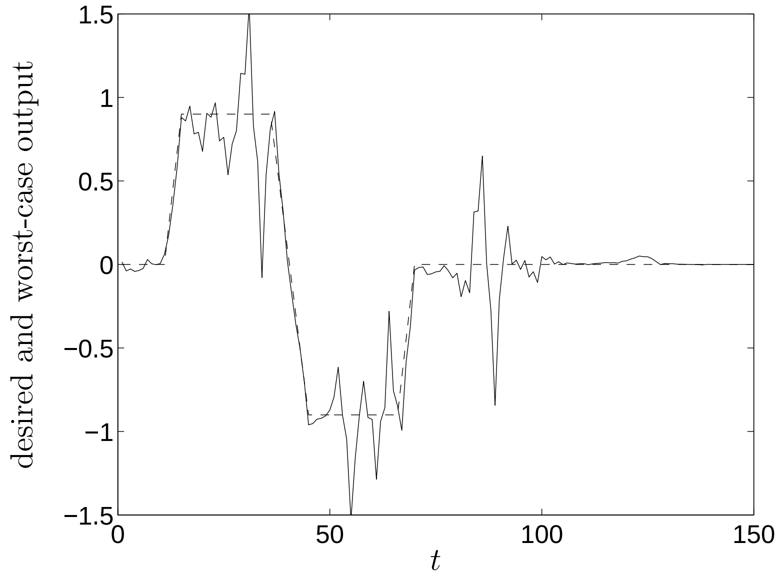

Figure 2: Worst case output for the input that minimizes the peak tracking error of the nominal system. The peak tracking error is 0.99.

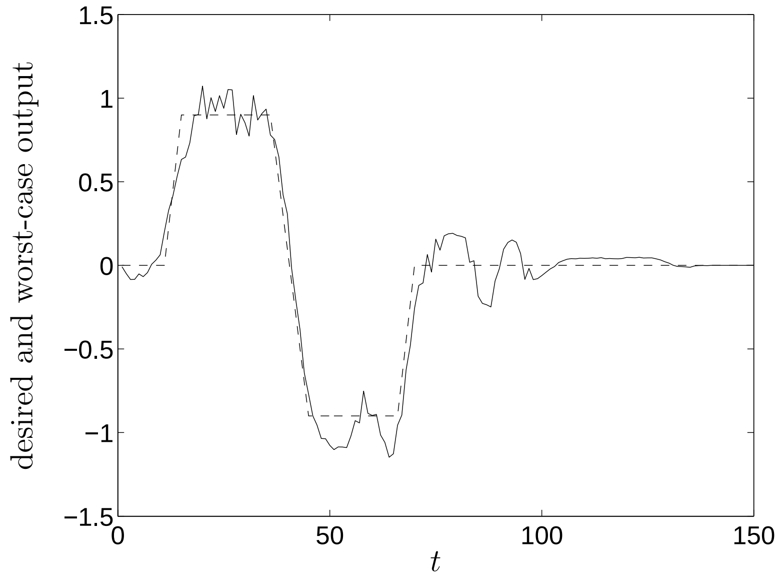

Figure 3: Worst case output for the input that minimizes the worst-case peak tracking error. The peak tracking error is 0.27.

tage of the structure in both problems. The algorithm is iterative, typically requiring 10–50 iterations. The cost of one iteration is $3Nn^3 + M(4mn^2 + 4m^2n + m^3/3)$ floating-point operations (flops) for the nonrobust problem, and $2Npn^2 + 3Nn^3 + M(4mn^2 + 4m^2n + m^3/3)$ flops for the robust problem. In other words, the computational complexity of both problems is comparable, and grows *linearly* with N and M.

Numerical algorithms for linear and quadratic programming have been applied to optimal control since the 60s [ZW62], and are widely used in model-predictive control [ML99, Raw00]. More recently, it was pointed out in [BCH98] that new interior-point methods for nonlinear convex optimization (for example, for second-order cone programming or semidefinite programming; see [NN94]) allow us to efficiently solve a much wider class of optimal control problems, including, for example, problems with uncertain system models, or nonlinear constraints on inputs and states. It was also noted that the resulting convex optimization problems are usually quite large, and may require special-purpose interior-point implementations that take advantage of problem structure.

We find two basic approaches in the literature on numerical implementation of interior-point methods for control. Both approaches focus on speeding up the solution of the large sets of linear equations that need to be solved at each iteration, in order to compute the search directions. A first idea is to use conjugate gradients to solve these linear systems [BVG94, Han00]. Many different types of structure can be exploited this way, often resulting in a speedup by several orders of magnitude. Unfortunately, the performance of the conjugate gradient method is also very sensitive to the problem data, and in general requires good preconditioners. Moreover, the excellent convergence properties of general-purpose interior-point implementations (typically 10–50 iterations) often degrade when conjugate gradients is used to compute search directions. The second approach is less general, but much more reliable, and is based on direct, non-iterative, methods for solving the linear systems fast. Wright, Rao, and Rawlings [Wri93, RWR98] and Hansson [Han00] have studied quadratic programming formulations of optimal control problems with linear constraints. They show that the Riccati recursion of (unconstrained) linear-quadratic optimal control can be used to compute the search directions in an interior-point method fast, *i.e.*, at a cost that is linear in the control horizon, and cubic in the system dimensions. The results of this paper can be viewed as an extension of the quadratic programming method of [RWR98] to the robust and nonrobust output tracking problems (1) and (2).

Robust convex optimization

We should also point out the connection with *robust convex optimization* [BTN98, EL97, EOL98, HB98]. The idea in robust convex optimization is to explicitly incorporate a model of data uncertainty in the formulation of a convex optimization problem, and to optimize for the worst-case scenario under that model. It has been shown that, depending on the uncertainty description, the robust version of a convex optimization problem may very well be intractable (nonconvex). Choosing a good model for the uncertainty involves a trade-off between conservativeness and tractability. Most of the research in the area has therefore focused on formulating robust convex optimization problems that can be solved via convex optimization. However, even if the resulting robust optimization problem is convex (hence solvable in polynomial time), it is often much larger and more difficult to solve than the corresponding nonrobust problem, so the added robustness of the solution comes at a very high price. Our results for the output tracking problem (2) and its robust counterpart (1) show that by exploiting problem structure we can solve certain robust convex optimization problems at a cost that is not much higher than the corresponding nonrobust problem.

Applications

As already mentioned, the main application of our results lies in model-predictive control (MPC). Problem (1) has been applied in robust model-predictive control by Allwright and Papavasiliou [AP92] and Zheng and Morari [ZM00]. Both papers use the LP formulation of §4.1, and solve it using general-purpose linear programming solvers. The main purpose of this paper is to show that these LPs can be solved much more efficiently if we exploit problem structure, and that the cost of solving the robust and nonrobust problems are comparable. Our approach also extends to many variations of problems (1) and (2) that have been studied in model-predictive control, for example, problems with an ℓ_1-objective [RR00], or problems including other input or state constraints (for example, slew rate constraints). Finally, we would like to emphasize that the paper focuses on the numerical aspects of solving (1). We do not address several important questions that arise when applying these results in model-predictive control, such as, for example, the question of stability (discussed in [ZM00]), or the question of updating the uncertainty set in between MPC moves.

Outline and notation

The outline of the paper is as follows. In §2 we discuss the classical Riccati recursion method for solving an unconstrained linear-quadratic optimal control problem. In §3 we formulate the nonrobust output tracking problem (2) as an LP and describe an efficient interior-point method for solving it. In §4 we extend this method to the robust output tracking problem (1). We discuss some extensions and summarize our results in §5. The appendices contain background material on interior-point methods for linear programming, and additional proofs.

Most of our notation is standard. We denote by $\mathbf{S}^n$ the space of symmetric matrices of size $n \times n$. The symbols $\succeq$, $\succ$, $\preceq$, and $\prec$ denote both componentwise inequality between vectors, and matrix inequality, depending on the context. For example, if $x \in \mathbf{R}^n$, then $x \succeq 0$ means $x_k \geq 0$ for $k = 1, \ldots, n$; if $x \in \mathbf{S}^n$, it means x is positive semidefinite. The symbol $\mathbf{1}$ denotes a vector with all its components equal to one. If $x \in \mathbf{R}^n$ and $y \in \mathbf{R}^p$, then $(x, y) \in \mathbf{R}^{n+p}$ denotes the vector $(x, y) = [x^T \ y^T]^T$.

2 Linear-quadratic optimal control

In this section we review the classical method for solving the linear-quadratic (LQ) optimal control problem

$$\begin{array}{ll} \text{minimize} & \sum_{t=1}^{N}(\frac{1}{2}x(t)^TQ(t)x(t) - q(t)^Tx(t)) + \sum_{t=0}^{M-1}(\frac{1}{2}u(t)^TR(t)u(t) - r(t)^Tu(t)) \\ \text{subject to} & x(1) = Ax_0 + Bu(0) \\ & x(t+1) = Ax(t) + Bu(t), \;\; t = 1,\ldots,M-1 \\ & x(t+1) = Ax(t), \;\; t = M,\ldots,N-1. \end{array}$$

The variables are $u(t)$, $t = 0,\ldots,M-1$ and $x(t)$, $t = 1,\ldots,N$. The weights $Q(t) \in \mathbf{S}^n$ and $R(t) \in \mathbf{S}^m$ are given and satisfy $Q(t) \succeq 0$ and $R(t) \succ 0$ for all t. We also assume that $N > M$. To simplify notation, we will express the problem as

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x} - \mathbf{q}^T\mathbf{x} + \frac{1}{2}\mathbf{u}^T\mathbf{R}\mathbf{u} - \mathbf{r}^T\mathbf{u} \\ \text{subject to} & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} = \mathbf{b} \end{array} \tag{4}$$

where

$$\mathbf{x} = \begin{bmatrix} x(1) \\ x(2) \\ \vdots \\ x(N) \end{bmatrix}, \quad \mathbf{q} = \begin{bmatrix} q(1) \\ q(2) \\ \vdots \\ q(N) \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} u(0) \\ u(1) \\ \vdots \\ u(M-1) \end{bmatrix}, \quad \mathbf{r} = \begin{bmatrix} r(0) \\ r(1) \\ \vdots \\ r(M-1) \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} -Ax_0 \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

$$\mathbf{A} = \begin{bmatrix} -I & 0 & 0 & \cdots & 0 & 0 \\ A & -I & 0 & \cdots & 0 & 0 \\ 0 & A & -I & \cdots & 0 & 0 \\ 0 & 0 & A & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & A & -I \end{bmatrix} \in \mathbf{R}^{Nn\times Nn}, \quad \mathbf{B} = \begin{bmatrix} B & 0 & \cdots & 0 \\ 0 & B & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & B \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbf{R}^{Nn\times Mm},$$

$$\mathbf{Q} = \begin{bmatrix} Q(1) & 0 & \cdots & 0 \\ 0 & Q(2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & Q(N) \end{bmatrix} \in \mathbf{S}^{Nn}, \quad \mathbf{R} = \begin{bmatrix} R(0) & 0 & \cdots & 0 \\ 0 & R(1) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R(M-1) \end{bmatrix} \in \mathbf{S}^{Mm}.$$

2.1 Optimality conditions

The quadratic optimization problem (4) can be solved by introducing a Lagrange multiplier $\mathbf{y} = (y(1), y(2), \ldots, y(N)) \in \mathbf{R}^{Nn}$, associated with the equality constraints. The optimality conditions are

$$\begin{bmatrix} 0 & \mathbf{A} & \mathbf{B} \\ \mathbf{A}^T & \mathbf{Q} & 0 \\ \mathbf{B}^T & 0 & \mathbf{R} \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{x} \\ \mathbf{u} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ \mathbf{q} \\ \mathbf{r} \end{bmatrix}, \tag{5}$$

which is a symmetric indefinite set of $2Nn + Mm$ equations in $2Nn + Mm$ variables ($\mathbf{x} \in \mathbf{R}^{Mm}$, $\mathbf{u} \in \mathbf{R}^{Nn}$, $\mathbf{y} \in \mathbf{R}^{Nn}$).

It follows from our assumptions ($\mathbf{Q} \succeq 0$, $\mathbf{R} \succ 0$) that the coefficient matrix is nonsingular: the (3,3)-block $\mathbf{R}$ is nonsingular and its Schur complement

$$\begin{bmatrix} -\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T & \mathbf{A} \\ \mathbf{A}^T & \mathbf{Q} \end{bmatrix} = \begin{bmatrix} -\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T & \mathbf{A} \\ \mathbf{A}^T & 0 \end{bmatrix} \begin{bmatrix} I & \mathbf{A}^{-T}\mathbf{Q} \\ 0 & I + \mathbf{A}^{-1}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{A}^{-T}\mathbf{Q} \end{bmatrix}$$

is also nonsingular, because $\mathbf{A}$ is nonsingular (it is lower triangular with diagonal elements equal to -1), and $\mathbf{Q}$ and $\mathbf{R}$ are positive semidefinite, so the eigenvalues of the matrix

$$\mathbf{A}^{-1}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{A}^{-T}\mathbf{Q}$$

are real and nonnegative.

In the next three paragraphs we describe a very efficient method for solving (5), by taking advantage of the block structure of $\mathbf{A}$, $\mathbf{B}$, $\mathbf{Q}$, and $\mathbf{R}$. The method is based on the Riccati recursion [AM90], with the linear algebra interpretation of [RWR98, Wri93]. We first show in §2.2 that the coefficient matrix can be factored as a product of five easily inverted matrices. The factorization can be obtained in roughly $3Nn^3 + M(4mn^2 + 4m^2n + m^3/3)$ flops, which is much less than the cost of a general-purpose LDL^T-algorithm ($(2Nn+Mm)^3/3$ flops). In particular, the complexity of the Riccati method grows *linearly* with the control horizons N and M. After the factorization is computed, the solution of (5) can be obtained by straightforward backward and forward substitutions at a cost of order $8Nn^2 + M(6mn + 2m^2)$ operations (see §2.3 and §2.4).

The overall complexity of the method is dominated by the factorization. As a consequence that will be important later on, we note that we can solve several linear systems of the form (5) with the same coefficient matrix but different righthand sides, by a single factorization, followed by several forward and backward substitutions. The cost of solving a few ($\ll n, m$) systems with the same coefficient matrix is therefore rougly the same as the cost of solving a single system, *i.e.*, $3Nn^3 + M(4mn^2 + 4m^2n + m^3/3)$ flops.

2.2 Factorization

To simplify the notation we permute the first two block rows and block columns of the matrix (5). We will show that the coefficient matrix can be factored as a product of five matrices as follows:

$$\begin{bmatrix} \mathbf{Q} & \mathbf{A}^T & 0 \\ \mathbf{A} & 0 & \mathbf{B} \\ 0 & \mathbf{B}^T & \mathbf{R} \end{bmatrix} =$$
$$\begin{bmatrix} (\mathbf{A}+\mathbf{B}\mathbf{K})^T & -\mathbf{S} & -\mathbf{K}^T \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ \mathbf{B}^T & 0 & I \end{bmatrix} \begin{bmatrix} -\mathbf{S} & I & 0 \\ I & 0 & 0 \\ 0 & 0 & \bar{\mathbf{R}} \end{bmatrix} \begin{bmatrix} I & 0 & \mathbf{B} \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \mathbf{A}+\mathbf{B}\mathbf{K} & 0 & 0 \\ -\mathbf{S} & I & 0 \\ -\mathbf{K} & 0 & I \end{bmatrix}, \quad (6)$$

where $\mathbf{S}$, $\bar{\mathbf{R}}$ and $\mathbf{K}$ are block matrices with the following structure:

$$\begin{aligned}
\mathbf{S} &= \begin{bmatrix} S(1) & 0 & \cdots & 0 \\ 0 & S(2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & S(N) \end{bmatrix} \in \mathbf{S}^{Nn}, \\
\bar{\mathbf{R}} &= \begin{bmatrix} \bar{R}(0) & 0 & \cdots & 0 \\ 0 & \bar{R}(1) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \bar{R}(M-1) \end{bmatrix} \in \mathbf{S}^{Mm}, \\
\mathbf{K} &= \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ K(1) & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & K(2) & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & K(M-1) & 0 & \cdots & 0 \end{bmatrix} \in \mathbf{R}^{Mm \times Nn},
\end{aligned}$$

with $S(t) \in \mathbf{S}^n$, $\bar{R}(t) \in \mathbf{S}^m$, and $K(t) \in \mathbf{R}^{m\times n}$. Moreover we have $\bar{\mathbf{R}} \succ 0$ and $\mathbf{S} \succeq 0$.

To verify this statement and to find out how we can compute $K(t)$, $S(t)$ and $\bar{R}(t)$, we multiply out the righthand side of (6), which yields

$$\begin{bmatrix} \mathbf{Q} & \mathbf{A}^T & 0 \\ \mathbf{A} & 0 & \mathbf{B} \\ 0 & \mathbf{B}^T & \mathbf{R} \end{bmatrix} = \begin{bmatrix} -\mathbf{A}^T\mathbf{S}\mathbf{A} - \mathbf{S}\mathbf{A} - \mathbf{A}^T\mathbf{S} + \mathbf{K}^T\bar{\mathbf{R}}\mathbf{K} & \mathbf{A}^T & -\mathbf{A}^T\mathbf{S}\mathbf{B} - \mathbf{S}\mathbf{B} - \mathbf{K}^T\bar{\mathbf{R}} \\ \mathbf{A} & 0 & \mathbf{B} \\ -\mathbf{B}^T\mathbf{S}\mathbf{A} - \mathbf{B}^T\mathbf{S} - \bar{\mathbf{R}}\mathbf{K} & \mathbf{B}^T & -\mathbf{B}^T\mathbf{S}\mathbf{B} + \bar{\mathbf{R}} \end{bmatrix}.$$

Therefore, $\mathbf{S}$, $\mathbf{K}$ and $\bar{\mathbf{R}}$ must satisfy

$$\begin{aligned}
\mathbf{S} &= \mathbf{Q} + (I+\mathbf{A})^T\mathbf{S}(I+\mathbf{A}) - \mathbf{K}^T\bar{\mathbf{R}}\mathbf{K} \\
\bar{\mathbf{R}} &= \mathbf{R} + \mathbf{B}^T\mathbf{S}\mathbf{B} \\
\bar{\mathbf{R}}\mathbf{K} &= -\mathbf{B}^T\mathbf{S}(I+\mathbf{A}),
\end{aligned}$$

or, more explicitly, in terms of the individual blocks,

$$\begin{aligned}
S(N) &= Q(N) && (7) \\
S(t) &= Q(t) + A^TS(t+1)A, \quad t = M,\dots,N-1 && (8) \\
S(t) &= Q(t) + A^TS(t+1)A - K(t)^T\bar{R}(t)K(t), \quad t = 1,\dots,M-1 && (9) \\
\bar{R}(t) &= R(t) + B^TS(t+1)B, \quad t = 0,\dots,M-1 && (10) \\
\bar{R}(t)K(t) &= -B^TS(t+1)A, \quad t = 1,\dots,M-1. && (11)
\end{aligned}$$

We have to show that these equations uniquely determine $S(t)$, $\bar{R}(t)$, and $K(t)$, and that $S(t) \succeq 0$ and $\bar{R}(t) \succ 0$. We first note that (7) and (8), combined with our assumption that $Q(t) \succeq 0$, immediately imply that $S(t) \succeq 0$ for $t = M,\dots,N$. Next, we assume that $S(t+1) \succeq 0$, for some $t < M$. We will show that equations (9)–(11) are solvable and yield a solution $S(t)$, $\bar{R}(t)$, $K(t)$ with $\bar{R}(t) \succ 0$ and $S(t) \succeq 0$. By induction this then implies that

$\bar{R}(t)$, $K(t)$, and $S(t)$ are well defined for all t, with $\bar{R}(t) \succ 0$ and $S(t) \succeq 0$. Equation (10) implies that $\bar{R}(t) \succ 0$, because $R(t) \succ 0$ and $S(t+1) \succeq 0$. Equation (11) then uniquely defines $K(t) = -\bar{R}(t)^{-1}B^T S(t+1)A$. Moreover,

$$\begin{bmatrix} Q(t) + A^T S(t+1)A & -K(t)^T \\ -K(t) & \bar{R}(t)^{-1} \end{bmatrix} =$$
$$\begin{bmatrix} Q(t) & 0 \\ 0 & \bar{R}(t)^{-1}R(t)\bar{R}(t)^{-1} \end{bmatrix} + \begin{bmatrix} A^T \\ \bar{R}(t)^{-1}B^T \end{bmatrix} S(t+1) \begin{bmatrix} A & B\bar{R}(t)^{-1} \end{bmatrix} \succeq 0.$$

The $(2,2)$ block of the lefthand side is positive definite, so its Schur complement must be positive semidefinite, *i.e.*,

$$S(t) = Q(t) + A^T S(t+1)A - K(t)^T \bar{R}(t) K(t) \succeq 0,$$

which completes the proof.

To conclude this section, we summarize the factorization algorithm, and give a flop count.

```
Factorization (Riccati recursion)
1.    $S(N) = Q(N)$
2.    for $t = N-1, N-2, \ldots, M$
3.      $S(t) = Q(t) + A^T S(t+1)A$
4.    end
5.    for $t = M-1, M-2, \ldots, 1$
6.      $\bar{R}(t) = R(t) + B^T S(t+1)B$
7.      $K(t) = -\bar{R}(t)^{-1}B^T S(t+1)A$
8.      $S(t) = Q(t) + A^T S(t+1)A - K(t)^T \bar{R}(t)K(t)$
9.    end
10.   $\bar{R}(0) = R(0) + B^T S(1)B$
```

The cost of the two matrix-matrix multiplications in step 3 is $3n^3$ flops ($2n^3$ flops to calculate $S(t+1)A$, and n^3 to calculate the symmetric matrix $A^T S(t+1)A$). The cost of step 6 is $2mn^2 + m^2 n$ ($2mn^2$ flops to calculate $S(t+1)B$ and $m^2 n$ to calculate the symmetric matrix $B^T S(t+1)B$). In step 7 we can first take the Cholesky factorization of $\bar{R}(t)$ ($m^3/3$ flops). This allows us to calculate $K(t)$ at a cost of $2n^3 + 2mn^2 + 2m^2 n$ flops ($2n^3$ to calculate $S(t+1)A$, $2mn^2$ to calculate $B^T S(t+1)A$, and $2m^2$ flops to solve for one column of $K(t)$). Step 8 costs $n^3 + m^2 n$ flops (n^3 to form $A^T S(t+1)A$, since $S(t+1)A$ was calculated in step 7, and $m^2 n$ to form $K(t)\bar{R}(t)K(t)$, since $\bar{R}(t)K(t)$ is known from step 7). Finally, step 10 costs $2mn^2 + m^2 n$ flops. Adding everything we obtain a total complexity of roughly

$$3Nn^3 + M(4mn^2 + 4m^2 n + m^3/3)$$

flops.

The five matrices in the factorization (6) are nonsingular (note that $\mathbf{A} + \mathbf{BK}$ is lower triangular, with diagonal elements -1). Combining the first two and the last two matrices, we can write the factored system as

$$\begin{bmatrix} \mathbf{A}^T & -\mathbf{S} & -\mathbf{K}^T \\ 0 & I & 0 \\ \mathbf{B}^T & 0 & I \end{bmatrix} \begin{bmatrix} -\mathbf{S} & I & 0 \\ I & 0 & 0 \\ 0 & 0 & \bar{\mathbf{R}} \end{bmatrix} \begin{bmatrix} \mathbf{A} & 0 & \mathbf{B} \\ -\mathbf{S} & I & 0 \\ -\mathbf{K} & 0 & I \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{u} \end{bmatrix} = \begin{bmatrix} \mathbf{q} \\ \mathbf{b} \\ \mathbf{r} \end{bmatrix}.$$

This factorization allows us to determine the solution of (5) by a forward and a backward recursion.

2.3 Backward recursion

Define $\tilde{\mathbf{x}} = (\tilde{x}(1), \dots, \tilde{x}(N))$, $\tilde{\mathbf{y}} = (\tilde{y}(1), \dots, \tilde{y}(N))$, $\tilde{\mathbf{u}} = (\tilde{u}(0), \dots, \tilde{u}(M-1))$, as the solution of

$$\begin{bmatrix} \mathbf{A}^T & -\mathbf{S} & -\mathbf{K}^T \\ 0 & I & 0 \\ \mathbf{B}^T & 0 & I \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{x}} \\ \tilde{\mathbf{y}} \\ \tilde{\mathbf{u}} \end{bmatrix} = \begin{bmatrix} \mathbf{q} \\ \mathbf{b} \\ \mathbf{r} \end{bmatrix},$$

and $\hat{\mathbf{x}} = (\hat{x}(1), \dots, \hat{x}(N))$, $\hat{\mathbf{y}} = (\hat{y}(1), \dots, \hat{y}(N))$, $\hat{\mathbf{u}} = (\hat{u}(0), \dots, \hat{u}(M-1))$, as the solution of

$$\begin{bmatrix} -\mathbf{S} & I & 0 \\ I & 0 & 0 \\ 0 & 0 & \bar{\mathbf{R}} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{x}} \\ \hat{\mathbf{y}} \\ \hat{\mathbf{u}} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{x}} \\ \tilde{\mathbf{y}} \\ \tilde{\mathbf{u}} \end{bmatrix}.$$

From the first equation it is clear that $\tilde{\mathbf{y}} = \mathbf{b}$ and $\tilde{\mathbf{u}} = \mathbf{r} - \mathbf{B}^T\tilde{\mathbf{x}}$, and $\tilde{\mathbf{x}}$ is defined by

$$(\mathbf{A} + \mathbf{B}\mathbf{K})^T \tilde{\mathbf{x}} = \mathbf{S}\mathbf{b} + \mathbf{K}^T\mathbf{r} + \mathbf{q}.$$

From $\tilde{\mathbf{x}}$ we can compute the solution of the second system

$$\hat{\mathbf{x}} = \mathbf{b}, \quad \hat{\mathbf{y}} = \tilde{\mathbf{x}} + \mathbf{S}\mathbf{b}, \quad \hat{\mathbf{u}} = \bar{\mathbf{R}}^{-1}(\mathbf{r} - \mathbf{B}^T\tilde{\mathbf{x}})$$

The following backward recursion calculates these variables. (Here, $b(t)$ stands for block t of the vector $\mathbf{b} = (b(0), b(1), \dots, b(N)$. Although in problem (4), $b(0) = -Ax_0$, and $b(t) = 0$ for $t \geq 1$, we give here the general method for solving (5) for arbitrary $\mathbf{b}$.)

```
Backward recursion
1.   $\hat{y}(N) = -q(N)$
2.   $\tilde{x}(N) = \hat{y}(N) - S(N)b(N)$
3.   $\hat{x}(N) = b(N)$
4.   for $t = N-1, N-2, \dots, M$
5.       $\hat{y}(t) = A^T\tilde{x}(t+1) - q(t)$
6.       $\tilde{x}(t) = \hat{y}(t) - S(t)b(t)$
7.       $\hat{x}(t) = b(t)$
8.   end
9.   for $t = M-1, M-2, \dots, 1$
10.      $\hat{y}(t) = A^T\tilde{x}(t+1) - K(t)^T(r(t) - B^T\tilde{x}(t+1)) - q(t)$
11.      $\tilde{x}(t) = \hat{y}(t) - S(t)b(t)$
12.      $\hat{u}(t) = \bar{R}(t)^{-1}(r(t) - B^T\tilde{x}(t+1))$
13.      $\hat{x}(t) = b(t)$
14.  end
15.  $\hat{u}(0) = \bar{R}(0)^{-1}(r(0) - B^T\tilde{x}(1))$
```

The required number of flops is $4Nn^2 + 2M(mn + m^2)$, if we reuse the matrices $A + BK(t)$ and the Cholesky factors of $\bar{R}(t)$ that are computed during the factorization.

2.4 Forward recursion

Finally, we have to solve

$$\begin{bmatrix} \mathbf{A} & 0 & \mathbf{B} \\ -\mathbf{S} & I & 0 \\ -\mathbf{K} & 0 & I \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{u} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}} \\ \hat{\mathbf{y}} \\ \hat{\mathbf{u}} \end{bmatrix},$$

i.e.,

$$(\mathbf{A} + \mathbf{B}\mathbf{K})\mathbf{x} = \hat{\mathbf{x}} - \mathbf{B}\tilde{\mathbf{u}}, \quad \mathbf{y} = \hat{\mathbf{y}} + \mathbf{S}\mathbf{x}, \quad \mathbf{u} = \hat{\mathbf{u}} + \mathbf{K}\mathbf{x}.$$

The solution is computed by the following forward recursion.

```
Forward recursion
1.    $x(1) = -\hat{x}(1) + B\tilde{u}(0)$
2     $y(1) = \hat{y}(1) + S(1)x(1)$
3     $u(0) = \hat{u}(0)$
4.    for $t = 2, 3, \ldots, M$
5.        $x(t) = (A + BK(t-1))x(t-1) - \tilde{x}(t) + B\tilde{u}(t-1)$
6         $y(t) = \hat{y}(t) + S(t)x(t)$
7         $u(t-1) = \hat{u}(t-1) + K(t-1)x(t-1)$
8.    end
9.    for $t = M+1, M+2, \ldots, N$
10.       $x(t) = Ax(t-1) - \tilde{x}(t)$
11.       $y(t) = \hat{y}(t) + S(t)x(t)$
12.   end
```

The required number of operations is $4Nn^2 + 4Mmn$.

3 The nonrobust output tracking problem

We now return to problems (1) and (2). We first describe an efficient method for solving (2), and defer problem (1) to Section §4. Following the matrix notation introduced in §2, we write (2) as

$$\begin{array}{ll} \text{minimize} & \|\mathbf{C}_0\mathbf{x} - \mathbf{y}_{\text{des}}\|_\infty \\ \text{subject to} & -\mathbf{1} \preceq \mathbf{u} \preceq \mathbf{1} \\ & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} = \mathbf{b} \end{array} \tag{12}$$

where $\mathbf{A}$, $\mathbf{x}$, $\mathbf{u}$, $\mathbf{b}$ are defined as in §2, and

$$\mathbf{C}_0 = \begin{bmatrix} c_0^T & 0 & \cdots & 0 \\ 0 & c_0^T & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & c_0^T \end{bmatrix} \in \mathbf{R}^{N \times Nn}, \quad \mathbf{y}_{\text{des}} = \begin{bmatrix} y_{\text{des}}(1) \\ y_{\text{des}}(2) \\ \vdots \\ y_{\text{des}}(N) \end{bmatrix} \in \mathbf{R}^N.$$

3.1 Linear programming formulation

Problem (12) is readily formulated as an LP

$$
\begin{array}{ll}
\text{minimize} & w \\
\text{subject to} & \begin{bmatrix} \mathbf{C}_0 & 0 & -\mathbf{1} \\ -\mathbf{C}_0 & 0 & -\mathbf{1} \\ 0 & I & 0 \\ 0 & -I & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \\ w \end{bmatrix} \preceq \begin{bmatrix} \mathbf{y}_{\mathrm{des}} \\ -\mathbf{y}_{\mathrm{des}} \\ \mathbf{1} \\ \mathbf{1} \end{bmatrix} \\
& \begin{bmatrix} \mathbf{A} & \mathbf{B} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \\ w \end{bmatrix} = \mathbf{b}.
\end{array}
\qquad (13)
$$

The variables are $w \in \mathbf{R}$, $\mathbf{x} \in \mathbf{R}^{Nn}$, and $\mathbf{u} \in \mathbf{R}^{Mm}$. The corresponding dual problem is

$$
\begin{array}{ll}
\text{maximize} & -\mathbf{y}_{\mathrm{des}}^T(\mathbf{z}_0^+ - \mathbf{z}_0^-) - \mathbf{1}^T(\mathbf{z}_u^+ + \mathbf{z}_u^-) - \mathbf{b}^T\mathbf{y} \\
\text{subject to} & \begin{bmatrix} \mathbf{C}_0^T & -\mathbf{C}_0^T & 0 & 0 & \mathbf{A}^T \\ 0 & 0 & I & -I & \mathbf{B}^T \\ -\mathbf{1}^T & -\mathbf{1}^T & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{z}_0^+ \\ \mathbf{z}_0^- \\ \mathbf{z}_u^+ \\ \mathbf{z}_u^- \\ \mathbf{y} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = 0 \\
& \mathbf{z}_0^+ \succeq 0,\ \mathbf{z}_0^- \succeq 0,\ \mathbf{z}_u^+ \succeq 0,\ \mathbf{z}_u^- \succeq 0.
\end{array}
\qquad (14)
$$

Both LPs are strictly feasible. For example, we obtain a strictly primal feasible point by choosing $\mathbf{u} = 0$, $x(1) = Ax_0$, $x(2) = Ax(1)$, $\ldots$, $x(N) = Ax(N-1)$, and any $w > \max_t |c_0^T x(t) - y_{\mathrm{des}}(t)|$. Strictly feasible dual values are given by $\mathbf{z}_0^+ = \mathbf{z}_0^- = \mathbf{1}/(2N)$, $\mathbf{z}_u^+ = \mathbf{z}_u^- = \mathbf{1}$, and $\mathbf{y} = 0$.

We also note that

$$
\mathbf{Rank}\left(\begin{bmatrix} \mathbf{C}_0 & 0 & -\mathbf{1} \\ -\mathbf{C}_0 & 0 & -\mathbf{1} \\ 0 & I & 0 \\ 0 & -I & 0 \\ \mathbf{A} & \mathbf{B} & 0 \end{bmatrix}\right) = Nn + Mm + 1, \quad \mathbf{Rank}\left(\begin{bmatrix} \mathbf{A} & \mathbf{B} & 0 \end{bmatrix}\right) = Nn \qquad (15)
$$

(*i.e.*, both matrices are full rank), because A is nonsingular.

3.2 Solution via interior-point methods

We propose to solve the LPs (13) and (14) using Mehrotra's method, one of the most popular and efficient interior-point methods for linear programming. The details of the method are given in Appendix §A. For our present purposes it is sufficient to note that each iteration of

Mehrotra's method, applied to (13), involves solving two sets of linear equations of the form

$$\begin{bmatrix} -\mathbf{D}_0^+ & 0 & 0 & 0 & 0 & \mathbf{C}_0 & 0 & -\mathbf{1} \\ 0 & -\mathbf{D}_0^- & 0 & 0 & 0 & -\mathbf{C}_0 & 0 & -\mathbf{1} \\ 0 & 0 & -\mathbf{D}_u^+ & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & -\mathbf{D}_u^- & 0 & 0 & -I & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{A} & \mathbf{B} & 0 \\ \mathbf{C}_0^T & -\mathbf{C}_0^T & 0 & 0 & \mathbf{A}^T & 0 & 0 & 0 \\ 0 & 0 & I & -I & \mathbf{B}^T & 0 & 0 & 0 \\ -\mathbf{1}^T & -\mathbf{1}^T & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta\mathbf{z}_0^+ \\ \Delta\mathbf{z}_0^- \\ \Delta\mathbf{z}_u^+ \\ \Delta\mathbf{z}_u^- \\ \Delta\mathbf{y} \\ \Delta\mathbf{x} \\ \Delta\mathbf{u} \\ \Delta w \end{bmatrix} = \begin{bmatrix} \mathbf{r}_0^+ \\ \mathbf{r}_0^- \\ \mathbf{r}_1^+ \\ \mathbf{r}_1^- \\ \mathbf{r}_2 \\ \mathbf{r}_3 \\ \mathbf{r}_4 \\ r_5 \end{bmatrix}. \qquad (16)$$

Both equations have the same values of $\mathbf{D}_0^+$, $\mathbf{D}_0^-$, $\mathbf{D}_u^+$, $\mathbf{D}_u^-$, but different righthand sides. The matrices $\mathbf{D}_0^+, \mathbf{D}_0^- \in \mathbf{S}^N$ and $\mathbf{D}_u^+, \mathbf{D}_u^- \in \mathbf{S}^M$ are positive diagonal, with values that change at each iteration.

To solve (16) we can first eliminate the variables $\Delta\mathbf{z}_0^+$, $\Delta\mathbf{z}_0^-$, $\Delta\mathbf{z}_u^+$, and $\Delta\mathbf{z}_u^-$ from the first four equations, which yields a set of equations of the form

$$\begin{bmatrix} 0 & \mathbf{A} & \mathbf{B} & 0 \\ \mathbf{A}^T & \mathbf{Q} & 0 & \mathbf{d} \\ \mathbf{B}^T & 0 & \mathbf{R} & 0 \\ 0 & \mathbf{d}^T & 0 & \gamma \end{bmatrix} \begin{bmatrix} \Delta\mathbf{y} \\ \Delta\mathbf{x} \\ \Delta\mathbf{u} \\ \Delta w \end{bmatrix} = \begin{bmatrix} \mathbf{r}_6 \\ \mathbf{r}_7 \\ \mathbf{r}_8 \\ r_9 \end{bmatrix}, \qquad (17)$$

where

$$\mathbf{Q} = \mathbf{C}_0^T \mathbf{D}_0 \mathbf{C}_0, \quad \mathbf{R} = (\mathbf{D}_u^-)^{-1} + (\mathbf{D}_u^+)^{-1}, \quad \mathbf{d} = \mathbf{C}_0^T \tilde{\mathbf{D}}_0 \mathbf{1}, \quad \gamma = \mathbf{Tr}\,\mathbf{D}_0, \qquad (18)$$

and

$$\mathbf{D}_0 = (\mathbf{D}_0^+)^{-1} + (\mathbf{D}_0^-)^{-1}, \quad \tilde{\mathbf{D}}_0 = (\mathbf{D}_0^-)^{-1} - (\mathbf{D}_0^+)^{-1}.$$

The righthand side of (17) is given by

$$\begin{aligned} \mathbf{r}_6 &= \mathbf{r}_2 \\ \mathbf{r}_7 &= \mathbf{r}_3 + \mathbf{C}_0^T((\mathbf{D}_0^+)^{-1}\mathbf{r}_0^+ - (\mathbf{D}_0^-)^{-1}\mathbf{r}_0^-) \\ \mathbf{r}_8 &= \mathbf{r}_4 + (\mathbf{D}_u^+)^{-1}\mathbf{r}_1^+ - (\mathbf{D}_u^-)^{-1}\mathbf{r}_1^- \\ r_9 &= r_5 - \mathbf{1}^T((\mathbf{D}_0^+)^{-1}\mathbf{r}_0^+ + (\mathbf{D}_0^-)^{-1}\mathbf{r}_0^-). \end{aligned}$$

It follows from the rank condition (15) that the system (17) is nonsingular (see Appendix §A).

Note that further eliminating Δw from (17) would result in a 3×3 block matrix with a large dense matrix $\mathbf{Q} - (1/\gamma)\mathbf{d}\mathbf{d}^T$ in the $(2,2)$-position. Instead of eliminating Δw, we therefore solve two equations

$$\begin{bmatrix} 0 & \mathbf{A} & \mathbf{B} \\ \mathbf{A}^T & \mathbf{Q} & 0 \\ \mathbf{B}^T & 0 & \mathbf{R} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{y}_1 \\ \Delta\mathbf{x}_1 \\ \Delta\mathbf{u}_1 \end{bmatrix} = \begin{bmatrix} \mathbf{r}_6 \\ \mathbf{r}_7 \\ \mathbf{r}_8 \end{bmatrix}, \quad \begin{bmatrix} 0 & \mathbf{A} & \mathbf{B} \\ \mathbf{A}^T & \mathbf{Q} & 0 \\ \mathbf{B}^T & 0 & \mathbf{R} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{y}_x \\ \Delta\mathbf{x}_2 \\ \Delta\mathbf{u}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{d} \\ 0 \end{bmatrix}, \qquad (19)$$

and then make a linear combination to satisfy the last equation, *i.e.*, calculate the solution of (17) as

$$\begin{bmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{y} \\ \Delta\mathbf{u} \end{bmatrix} = \begin{bmatrix} \Delta\mathbf{x}_1 \\ \Delta\mathbf{y}_1 \\ \Delta\mathbf{u}_1 \end{bmatrix} - \Delta w \begin{bmatrix} \Delta\mathbf{x}_2 \\ \Delta\mathbf{y}_2 \\ \Delta\mathbf{u}_2 \end{bmatrix}, \quad \Delta w = \frac{r_9 - \mathbf{d}^T\Delta\mathbf{x}_1}{\gamma - \mathbf{d}^T\Delta\mathbf{x}_2}.$$

The equations (19) have exactly the same form as (5). Moreover $\mathbf{R}$ is positive diagonal, and $\mathbf{Q}$ is block diagonal with positive semidefinite diagonal blocks

$$Q(t) = D_0(t) c_0 c_0^T, \quad t = 1, \ldots, N,$$

where the diagonal elements of $\mathbf{D}_0$ are denoted by $D_0(t)$, *i.e.*, $\mathbf{D}_0 = \mathbf{diag}(D_0(1), \ldots, D_0(N))$. We can therefore apply the Riccati method of §2, and solve (16) in roughly

$$3Nn^3 + M(4mn^2 + 4m^2 n + m^3/3)$$

operations. (Note that the cost of constructing $\mathbf{Q}$ is $2n^2N$ operations, so it can be neglected compared to the cost of the factorization.)

3.3 Numerical results

The algorithm described above has been implemented in Matlab (Version 6) and tested on a 933 Mhz Pentium III running Linux.

Table 1 summarizes the results for a family of randomly generated problems. The first four columns give the problem dimensions. Columns 5–7 give the number of variables, inequalities, and equality constraints in the corresponding LPs (13). The last three columns give the number of iterations to reach a relative error of 0.1%, the total CPU time, and the CPU time per iteration.

The results confirm that the number of iterations grows slowly with problem size, and typically ranges between 10 and 50. The most important data are in the last column, which gives the CPU time per iteration. It shows that the CPU time per iteration grows linearly with N and M. Within the range of dimensions considered here ($n \leq 40$, $m \leq 20$), the cost per iterations appears to grow more slowly with n and m than predicted by the theory (which predicts a cubic increase).

4 The robust output tracking problem

We now extend the method of the previous paragraph to the robust tracking problem (1), which can be expressed concisely as

$$\begin{array}{ll} \text{minimize} & \sup_{\|\rho\|_\infty \leq 1} \|(\mathbf{C}_0 + \sum_{i=1}^p \rho_i \mathbf{C}_i)\mathbf{x} - \mathbf{y}_{\mathrm{des}}\|_\infty \\ \text{subject to} & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} = \mathbf{b} \\ & -\mathbf{1} \preceq \mathbf{u} \preceq \mathbf{1} \end{array}$$

where

$$\mathbf{C}_i = \begin{bmatrix} c_i^T & 0 & \cdots & 0 \\ 0 & c_i^T & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & c_i^T \end{bmatrix} \in \mathbf{R}^{N \times Nn}.$$

The other matrices and vectors are defined as before.

dimensions				LP dimensions			# iters	CPU time	time/iter
N	M	n	m	#vars.	#ineqs	#eqs.		(seconds)	(seconds)
100	50	5	2	601	1200	1000	8	0.9	0.1
500	450	5	2	3401	6800	5000	9	6.9	0.8
1000	950	5	2	6901	13800	10000	11	17.5	1.6
2000	1950	5	2	13901	27800	20000	13	42.9	3.3
100	50	10	5	1251	2500	2000	10	1.3	0.1
500	450	10	5	7251	14500	10000	14	12.8	0.9
1000	950	10	5	14751	29500	20000	11	20.4	1.9
2000	1950	10	5	29751	59500	40000	14	54.3	3.9
100	50	20	10	2501	5000	4000	13	2.4	0.2
500	450	20	10	14501	29000	20000	15	18.6	1.3
1000	950	20	10	29501	59000	40000	16	41.3	2.6
2000	1950	20	10	59501	119000	80000	21	113.9	5.4
100	50	40	20	5001	10000	8000	17	6.6	0.4
500	450	40	20	29001	58000	40000	15	40.0	2.7
1000	950	40	20	59001	118000	80000	21	121.0	5.8
2000	1950	40	20	119001	238999	160000	27	304.1	11.3

Table 1: Number of iterations and CPU times for a family of output tracking problems with randomly generated data.

4.1 Linear programming formulation

To express (1) as an LP, we first switch the maximum and the supremum in the objective, introduce a scalar variable w, and write the problem as

$$\begin{array}{ll} \text{minimize} & w \\ \text{subject to} & \sup_{\|\rho\|_\infty \le 1} |(c_0 + \sum_{i=1}^p \rho_i c_i)^T x(t) - y_{\text{des}}(t)| \le w, \quad t = 1, \ldots, N \\ & x(1) = Ax_0 + Bu(0) \\ & x(t+1) = Ax(t) + Bu(t), \;\; t = 1, \ldots, M-1 \\ & x(t+1) = Ax(t), \;\; t = M, \ldots, N-1 \\ & -\mathbf{1} \preceq u(t) \preceq \mathbf{1}, \;\; t = 0, \ldots, M-1. \end{array} \tag{20}$$

The first constraint is satisfied at time t if and only if, for all ρ with $\|\rho\|_\infty \le 1$,

$$-w \le c_0^T x(t) + \sum_{i=1}^p \rho_i c_i^T x(t) - y_{\text{des}}(t) \le w,$$

which is true if and only if

$$-w \le c_0^T x(t) - \sum_{i=1}^p |c_i^T x(t)| - y_{\text{des}}(t), \quad c_0^T x(t) + \sum_{i=1}^p |c_i^T x(t)| - y_{\text{des}}(t) \le w.$$

We can cast these two nonlinear convex inequalities as the following set of linear inequalities by introducing p auxiliary variables $v_i(t)$, $i = 1, \ldots, p$:

$$\begin{array}{c} -w \le c_0^T x(t) - \sum_{i=1}^p v_i(t) - y_{\text{des}}(t) \\ c_0^T x(t) + \sum_{i=1}^p v_i(t) - y_{\text{des}}(t) \le w \\ -v_i(t) \le c_i^T x(t) \le v_i(t). \end{array}$$

These four inequalities (which are linear in $x(t)$, $v_i(t)$, w) are equivalent to the first constraint in (20) at time t. We can therefore express (20) as an LP

$$\begin{array}{ll} \text{minimize} & w \\ \text{subject to} & c_0^T x + \sum_{i=1}^p v_i(t) - y_{\text{des}}(t) \le w, \quad t = 1, \ldots, N \\ & c_0^T x - \sum_{i=1}^p v_i(t) - y_{\text{des}}(t) \ge -w, \quad t = 1, \ldots, N \\ & -v_i(t) \le c_i^T x \le v_i(t), \quad t = 1, \ldots, N \\ & x(1) = Ax_0 + Bu(0) \\ & x(t+1) = Ax(t) + Bu(t), \;\; t = 1, \ldots, M-1 \\ & x(t+1) = Ax(t), \;\; t = M, \ldots, N-1 \\ & -\mathbf{1} \preceq u(t) \preceq \mathbf{1}, \;\; t = 0, \ldots, M-1. \end{array}$$

The variables are $w \in \mathbf{R}$, $v_i(t) \in \mathbf{R}$ $(i = 1, \ldots, p,\ t = 1, \ldots, N)$, $x(t) \in \mathbf{R}^n$ $(t = 1, \ldots, N)$, and $u(t) \in \mathbf{R}^m$ $(t = 0, \ldots, M-1)$. We can write the LP more clearly as

$$\begin{array}{ll} \text{minimize} & w \\ \text{subject to} & \begin{bmatrix} \mathbf{C}_0 & 0 & \mathbf{E} & -\mathbf{1} \\ -\mathbf{C}_0 & 0 & \mathbf{E} & -\mathbf{1} \\ \mathbf{C} & 0 & -I & 0 \\ -\mathbf{C} & 0 & -I & 0 \\ 0 & I & 0 & 0 \\ 0 & -I & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \\ \mathbf{v} \\ w \end{bmatrix} \preceq \begin{bmatrix} \mathbf{y}_{\text{des}} \\ -\mathbf{y}_{\text{des}} \\ 0 \\ 0 \\ \mathbf{1} \\ \mathbf{1} \end{bmatrix}, \\ & \begin{bmatrix} \mathbf{A} & \mathbf{B} & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \\ \mathbf{v} \\ w \end{bmatrix} = \mathbf{b}, \end{array} \tag{21}$$

where

$$\mathbf{E} = \begin{bmatrix} I & I & \cdots & I \end{bmatrix} \in \mathbf{R}^{N \times Np}, \quad \mathbf{C} = \begin{bmatrix} \mathbf{C}_1 \\ \mathbf{C}_2 \\ \vdots \\ \mathbf{C}_p \end{bmatrix} \in \mathbf{R}^{Np \times Nn},$$

and $\mathbf{v} = (v_1(1), \ldots, v_1(N), v_2(1), \ldots, v_2(N), \ldots, v_p(1), \ldots, v_p(N)) \in \mathbf{R}^{Np}$. The variables are $\mathbf{x} \in \mathbf{R}^{Nn}$, $\mathbf{u} \in \mathbf{R}^{Nm}$, and the auxiliary variable $\mathbf{v} \in \mathbf{R}^{Np}$. The dual LP is given by

$$\begin{array}{ll} \text{maximize} & -\mathbf{y}_{\text{des}}^T(\mathbf{z}_0^+ - \mathbf{z}_0^-) - \mathbf{1}^T(\mathbf{z}_u^+ + \mathbf{z}_u^-) - \mathbf{b}^T\mathbf{y} \\ \text{subject to} & \begin{bmatrix} \mathbf{C}_0^T & -\mathbf{C}_0^T & \mathbf{C}^T & -\mathbf{C}^T & 0 & 0 & \mathbf{A}^T \\ 0 & 0 & 0 & 0 & I & -I & \mathbf{B}^T \\ \mathbf{E}^T & \mathbf{E}^T & -I & -I & 0 & 0 & 0 \\ -\mathbf{1}^T & -\mathbf{1}^T & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{z}_0^+ \\ \mathbf{z}_0^- \\ \mathbf{z}^+ \\ \mathbf{z}^- \\ \mathbf{z}_u^+ \\ \mathbf{z}_u^- \\ \mathbf{y} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = 0 \\ & \mathbf{z}_0^+ \succeq 0,\ \mathbf{z}_0^- \succeq 0,\ \mathbf{z}^+ \succeq 0,\ \mathbf{z}^- \succeq 0,\ \mathbf{z}_u^+ \succeq 0,\ \mathbf{z}_u^- \succeq 0. \end{array} \tag{22}$$

As in §3, both LPs are strictly feasible. In (21) we can take $\mathbf{u} = 0$, $x(1) = Ax_0$, $x(2) = Ax(1)$, $\ldots$, $x(N) = Ax(N-1)$, any $\mathbf{v}$ satisfying $-\mathbf{v} \prec \mathbf{C}\mathbf{x} \prec \mathbf{v}$, and any w satisfying

$$-w\mathbf{1} + \mathbf{E}\mathbf{v} \prec \mathbf{C}_0\mathbf{x} - \mathbf{y}_{\text{des}} \prec w\mathbf{1} - \mathbf{E}\mathbf{v}.$$

Strictly feasible dual values are given by $\mathbf{z}_0^+ = \mathbf{z}_0^- = \mathbf{1}/(2N)$, $\mathbf{z}^+ = \mathbf{E}^T\mathbf{z}_0^+$, $\mathbf{z}^- = \mathbf{E}^T\mathbf{z}_0^-$, $\mathbf{z}_u^+ = \mathbf{z}_u^- = \mathbf{1}$, and $\mathbf{y} = 0$.

We also note that the constraint matrix in (22) has rank $Nn + Np + Mm + 1$, and $\mathbf{Rank}([\mathbf{A}\ \mathbf{B}\ 0\ 0]) = Nn$, *i.e.*, both matrices have full rank.

4.2 Solution via interior-point methods

We now examine the cost of applying Mehrotra's method to the LPs (21) and (22). At each iteration of Mehrotra's method, we must solve two sets of linear equations of the form

$$\begin{bmatrix} -\mathbf{D}_0^+ & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{C}_0 & 0 & \mathbf{E} & -\mathbf{1} \\ 0 & -\mathbf{D}_0^- & 0 & 0 & 0 & 0 & 0 & -\mathbf{C}_0 & 0 & \mathbf{E} & -\mathbf{1} \\ 0 & 0 & -\mathbf{D}^+ & 0 & 0 & 0 & 0 & \mathbf{C} & 0 & -I & 0 \\ 0 & 0 & 0 & -\mathbf{D}^- & 0 & 0 & 0 & -\mathbf{C} & 0 & -I & 0 \\ 0 & 0 & 0 & 0 & -\mathbf{D}_u^+ & 0 & 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\mathbf{D}_u^- & 0 & 0 & -I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{A} & \mathbf{B} & 0 & 0 \\ \mathbf{C}_0^T & -\mathbf{C}_0^T & \mathbf{C}^T & -\mathbf{C}^T & 0 & 0 & \mathbf{A}^T & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I & -I & \mathbf{B}^T & 0 & 0 & 0 & 0 \\ \mathbf{E}^T & \mathbf{E}^T & -I & -I & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\mathbf{1}^T & -\mathbf{1}^T & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta\mathbf{z}_0^+ \\ \Delta\mathbf{z}_0^- \\ \Delta\mathbf{z}^+ \\ \Delta\mathbf{z}^- \\ \Delta\mathbf{z}_u^+ \\ \Delta\mathbf{z}_u^- \\ \Delta\mathbf{y} \\ \Delta\mathbf{x} \\ \Delta\mathbf{u} \\ \Delta\mathbf{v} \\ \Delta w \end{bmatrix} = \begin{bmatrix} \mathbf{r}_0^+ \\ \mathbf{r}_0^- \\ \mathbf{r}_1^+ \\ \mathbf{r}_1^- \\ \mathbf{r}_2^+ \\ \mathbf{r}_2^- \\ \mathbf{r}_3 \\ \mathbf{r}_4 \\ \mathbf{r}_5 \\ \mathbf{r}_6 \\ r_7 \end{bmatrix}. \tag{23}$$

The matrices $\mathbf{D}_0^+$, $\mathbf{D}_0^-$, $\mathbf{D}^+$, $\mathbf{D}^-$, $\mathbf{D}_u^+$, $\mathbf{D}_u^-$ are positive diagonal, with values that change at each iteration, but are equal for both sets of equations. Eliminating $\Delta\mathbf{z}_0^+$, $\Delta\mathbf{z}_0^-$, $\Delta\mathbf{z}^+$, $\Delta\mathbf{z}^-$, $\Delta\mathbf{z}_u^+$, $\Delta\mathbf{z}_u^-$ from the first six equations yields

$$\begin{bmatrix} 0 & \mathbf{A} & \mathbf{B} & 0 & 0 \\ \mathbf{A}^T & \mathbf{C}_0^T\mathbf{D}_0\mathbf{C}_0 + \mathbf{C}^T\mathbf{D}\mathbf{C} & 0 & -\mathbf{C}_0^T\tilde{\mathbf{D}}_0\mathbf{E} + \mathbf{C}^T\tilde{\mathbf{D}} & \mathbf{C}_0^T\tilde{\mathbf{D}}_0\mathbf{1} \\ \mathbf{B}^T & 0 & \mathbf{R} & 0 & 0 \\ 0 & -\mathbf{E}^T\tilde{\mathbf{D}}_0\mathbf{C}_0 + \tilde{\mathbf{D}}\mathbf{C} & 0 & \mathbf{E}^T\mathbf{D}_0\mathbf{E} + \mathbf{D} & -\mathbf{E}^T\mathbf{D}_0\mathbf{1} \\ 0 & \mathbf{1}^T\tilde{\mathbf{D}}_0\mathbf{C}_0 & 0 & -\mathbf{1}^T\mathbf{D}_0\mathbf{E} & \mathbf{Tr}\,\mathbf{D}_0 \end{bmatrix} \begin{bmatrix} \Delta\mathbf{y} \\ \Delta\mathbf{x} \\ \Delta\mathbf{u} \\ \Delta\mathbf{v} \\ \Delta w \end{bmatrix} = \begin{bmatrix} \mathbf{r}_8 \\ \mathbf{r}_9 \\ \mathbf{r}_{10} \\ \mathbf{r}_{11} \\ r_{12} \end{bmatrix} \tag{24}$$

where the matrices

$$\mathbf{D}_0 = (\mathbf{D}_0^+)^{-1} + (\mathbf{D}_0^-)^{-1}, \quad \mathbf{D} = (\mathbf{D}^+)^{-1} + (\mathbf{D}^-)^{-1}, \quad \mathbf{R} = (\mathbf{D}_u^+)^{-1} + (\mathbf{D}_u^-)^{-1}$$

are positive diagonal, and

$$\tilde{\mathbf{D}}_0 = -(\mathbf{D}_0^+)^{-1} + (\mathbf{D}_0^-)^{-1}, \quad \tilde{\mathbf{D}} = -(\mathbf{D}^+)^{-1} + (\mathbf{D}^-)^{-1}$$

are diagonal. The righthand side is given by

$$\begin{aligned} \mathbf{r}_8 &= \mathbf{r}_3 \\ \mathbf{r}_9 &= \mathbf{r}_4 + \mathbf{C}_0^T((\mathbf{D}_0^+)^{-1}\mathbf{r}_0^+ - (\mathbf{D}_0^-)^{-1}\mathbf{r}_0^-) + \mathbf{C}^T((\mathbf{D}^+)^{-1}\mathbf{r}_1^+ - (\mathbf{D}^-)^{-1}\mathbf{r}_1^-) \\ \mathbf{r}_{10} &= \mathbf{r}_5 + (\mathbf{D}_u^+)^{-1}\mathbf{r}_2^+ - (\mathbf{D}_u^-)^{-1}\mathbf{r}_2^- \\ \mathbf{r}_{11} &= \mathbf{r}_6 + \mathbf{E}^T((\mathbf{D}_0^+)^{-1}\mathbf{r}_0^+ + (\mathbf{D}_0^-)^{-1}\mathbf{r}_0^-) - (\mathbf{D}^+)^{-1}\mathbf{r}_1^+ - (\mathbf{D}^-)^{-1}\mathbf{r}_1^- \\ r_{12} &= r_7 - \mathbf{1}^T((\mathbf{D}_0^+)^{-1}\mathbf{r}_0^+ + (\mathbf{D}_0^-)^{-1}\mathbf{r}_0^-). \end{aligned}$$

Next, we note that the matrix in the $(4,4)$-position of (24) has the form

$$\mathbf{E}^T\mathbf{D}_0\mathbf{E} + \mathbf{D} = \begin{bmatrix} \mathbf{D}_0 & \mathbf{D}_0 & \cdots & \mathbf{D}_0 \\ \mathbf{D}_0 & \mathbf{D}_0 & \cdots & \mathbf{D}_0 \\ \vdots & \vdots & & \vdots \\ \mathbf{D}_0 & \mathbf{D}_0 & \cdots & \mathbf{D}_0 \end{bmatrix} + \begin{bmatrix} \mathbf{D}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{D}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{D}_p \end{bmatrix}$$

(where $\mathbf{D}_i \in \mathbf{S}^n$), so it is easily verified that its inverse is given by

$$
\begin{aligned}
(\mathbf{E}^T\mathbf{D}_0\mathbf{E}+\mathbf{D})^{-1} &= \mathbf{D}^{-1} - \mathbf{D}^{-1}\mathbf{E}^T(\mathbf{D}_0^{-1} + \sum_{i=1}^{p}\mathbf{D}_i^{-1})^{-1}\mathbf{E}\mathbf{D}^{-1} \\
&= \begin{bmatrix} \mathbf{D}_1^{-1} & 0 & \cdots & 0 \\ 0 & \mathbf{D}_2^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{D}_p^{-1} \end{bmatrix} - \begin{bmatrix} \mathbf{D}_1^{-1} \\ \mathbf{D}_2^{-1} \\ \vdots \\ \mathbf{D}_p^{-1} \end{bmatrix} (\mathbf{D}_0^{-1} + \sum_{i=1}^{p}\mathbf{D}_i^{-1})^{-1} \begin{bmatrix} \mathbf{D}_1^{-1} & \mathbf{D}_2^{-1} & \cdots & \mathbf{D}_p^{-1} \end{bmatrix}.
\end{aligned}
$$

We can therefore eliminate $\Delta\mathbf{v}$ from (24). The resulting system of equations is

$$
\begin{bmatrix} 0 & \mathbf{A} & \mathbf{B} & 0 \\ \mathbf{A}^T & \mathbf{Q} & 0 & \mathbf{d} \\ \mathbf{B}^T & 0 & \mathbf{R} & 0 \\ 0 & \mathbf{d}^T & 0 & \gamma \end{bmatrix} \begin{bmatrix} \Delta\mathbf{y} \\ \Delta\mathbf{x} \\ \Delta\mathbf{u} \\ \Delta w \end{bmatrix} = \begin{bmatrix} \mathbf{r}_{13} \\ \mathbf{r}_{14} \\ \mathbf{r}_{15} \\ r_{16} \end{bmatrix} \tag{25}
$$

where

$$
\begin{aligned}
\mathbf{Q} &= \sum_{i=0}^{p}\mathbf{C}_i^T(\mathbf{D}_i - \tilde{\mathbf{D}}_i^2\mathbf{D}_i^{-1})\mathbf{C}_i + (\sum_{i=0}^{p}\mathbf{C}_i^T\tilde{\mathbf{D}}_i\mathbf{D}_i^{-1})(\sum_{i=0}^{p}\mathbf{D}_i^{-1})^{-1}(\sum_{i=0}^{p}\tilde{\mathbf{D}}_i\mathbf{D}_i^{-1}\mathbf{C}_i) && (26)\\
\mathbf{d} &= (\sum_{i=0}^{p}\mathbf{C}_i^T\tilde{\mathbf{D}}_i\mathbf{D}_i^{-1})(\sum_{i=0}^{p}\mathbf{D}_i^{-1})^{-1}\mathbf{1} && (27)\\
\gamma &= \mathbf{Tr}(\sum_{i=0}^{p}\mathbf{D}_i^{-1})^{-1} && (28)\\
\mathbf{r}_{13} &= \mathbf{r}_8 \\
\mathbf{r}_{14} &= \mathbf{r}_9 + (\sum_{i=0}^{p}\mathbf{C}_i^T\tilde{\mathbf{D}}_i\mathbf{D}_i^{-1})(\sum_{i=0}^{p}\mathbf{D}_i^{-1})^{-1}\mathbf{E}\mathbf{D}^{-1}\mathbf{r}_{11} - \mathbf{C}^T\tilde{\mathbf{D}}\mathbf{D}^{-1}\mathbf{r}_{11} \\
\mathbf{r}_{15} &= \mathbf{r}_{10} \\
r_{16} &= r_{12} + \mathbf{1}^T(\sum_{i=0}^{p}\mathbf{D}_i^{-1})^{-1}\mathbf{E}\mathbf{D}^{-1}\mathbf{r}_{11}.
\end{aligned}
$$

(The derivation of these expressions is straightforward but tedious, and is given in Appendix B.)

The matrix $\mathbf{Q}$ defined in (26) is block-diagonal with positive semidefinite diagonal blocks

$$
Q(t) = \sum_{i=0}^{p}\frac{D_i(t)^2 - \tilde{D}_i(t)^2}{D_i(t)}c_ic_i^T + \frac{1}{\sum_{i=0}^{p}D_i(t)^{-1}}\left(\sum_{i=0}^{p}\frac{\tilde{D}_i(t)}{D_i(t)}c_i\right)\left(\sum_{i=0}^{p}\frac{\tilde{D}_i(t)}{D_i(t)}c_i\right)^T,
$$

where $D_i(t)$ and $\tilde{D}_i(t)$ denote the diagonal elements of $\mathbf{D}_i = \mathbf{diag}(D_i(1),\ldots,D_i(N))$ and $\tilde{\mathbf{D}}_i = \mathbf{diag}(\tilde{D}_i(1),\ldots,\tilde{D}_i(N))$. From this expression it is clear that the cost of forming $\mathbf{Q}$ is approximately $2Npn^2$ flops, ignoring lower-order terms. The system (25) has the same structure as (17), and can be solved using the same method. The total complexity of one iteration of Mehrotra's method is therefore about

$$
2Npn^2 + 3Nn^3 + M(4mn^2 + 4m^2n + m^3/3)
$$

operations.

dimensions					LP dimensions			# iters	CPU time	time/iter
N	M	n	m	p	#vars.	#ineqs	#eqs.		(seconds)	(seconds)
100	50	5	2	2	801	1600	1000	10	1.3	0.1
500	450	5	2	2	4401	8800	5000	12	10.4	0.9
1000	950	5	2	2	8901	17800	10000	10	17.6	1.8
2000	1950	5	2	2	17901	35800	20000	12	44.5	3.7
100	50	10	5	5	1751	3500	2000	9	1.4	0.2
500	450	10	5	5	9751	19500	10000	13	13.7	1.1
1000	950	10	5	5	19751	39500	20000	13	28.4	2.2
2000	1950	10	5	5	39751	79500	40000	16	71.5	4.5
100	50	20	10	10	3501	7000	4000	20	4.1	0.2
500	450	20	10	10	19501	39000	20000	16	21.7	1.4
1000	950	20	10	10	39501	79000	40000	21	62.0	3.0
2000	1950	20	10	10	79501	159000	80000	21	126.8	6.0
100	50	40	20	20	7001	14000	8000	18	7.9	0.4
500	450	40	20	20	39001	78000	40000	23	69.8	3.0
1000	950	40	20	20	79001	158000	80000	24	152.7	6.4
2000	1950	40	20	20	159001	318000	160000	22	297.7	13.5

Table 2: Number of iterations and CPU times for a family of robust output tracking prolems with randomly generated data.

4.3 Numerical results

Table 2 summarizes numerical results for a family of robust output tracking problems with randomly generated data. The algorithm was implemented in Matlab and tested on a 933 MHz Pentium III.

We again note that the CPU time per iteration grows linearly with the tracking and control horizons N and M, and appears to grow more slowly than cubicly with n, m, and p, at least in the range $n \leq 40$, $m, p \leq 20$.

Comparing with table 1 we also note that the cost of solving the robust output tracking problem is only slightly higher than the cost of solving the nonrobust problem, in spite of the fact that the corresponding LPs are much larger.

5 Conclusions

We have described efficient methods for solving a constrained linear optimal control problem and its robust counterpart. The methods are based on a primal-dual interior-point method for linear programming, and take a number of iterations that typically ranges between 10 and 50 and appears to grow very slowly with problem size. The cost per iteration is dominated by the solution of a large, structured set of linear equations. By exploiting problem structure, we are able to reduce these linear equations to the solution of an unconstrained quadratic linear optimal control problem, which can be solved very efficiently by the well-known Riccati recursion.

We compare in detail the cost of solving the output tracking problem and its robust counterpart. The robust problem is readily formulated as an LP, similar to the LP formulation of the nonrobust problem, but with an additional Np variables and $2Np$ inequality constraints. The main contribution of the paper is to show that, despite the size differences of the corresponding LPs, the robust output tracking problem can be solved at a cost that is not much higher than the nonrobust problem.

The techniques discussed here extend to a variety of related problems, for example, problems with an ℓ_1-objective or a quadratic objective, problems with additional convex constraints such as slew rate constraints, and problems with ellipsoidal uncertainty.

Acknowledgment

We thank Anders Hansson for interesting discussions on the topic of this paper.

References

[AM90] B. Anderson and J. B. Moore. *Optimal Control: Linear Quadratic Methods.* Prentice-Hall, 1990.

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

[BCH98] S. Boyd, C. Crusius, and A. Hansson. Control applications of nonlinear convex programming. *Journal of Process Control*, 8(5-6):313–324, 1998. Special issue for papers presented at the 1997 IFAC Conference on Advanced Process Control, June 1997, Banff.

[BTN98] A. Ben-Tal and A. Nemirovski. Robust convex optimization. *Mathematics of Operations Research*, 23:769–805, 1998.

[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.

[EL97] L. El Ghaoui and H. Lebret. Robust solutions to least-squares problems with uncertain data. *SIAM J. on Matrix Analysis and Applications*, 18(4):1035–1064, October 1997.

[EOL98] L. El Ghaoui, F. Oustry, and H. Lebret. Robust solutions to uncertain semidefinite programs. *SIAM J. on Optimization*, 9(1):33–52, 1998.

[Han00] A. Hansson. A primal-dual interior-point method for robust optimal control of linear discrete-time systems. *IEEE Trans. Aut. Control*, 45:1639–1655, 2000.

[HB98] H. Hindi and S. Boyd. Robust solutions to ℓ_1, ℓ_2 and ℓ_∞ uncertain linear approximation problems using convex optimization. In *Proc. American Control Conf.*, 1998.

[ML99] M. Morari and J. H. Lee. Model predictive control: past, present and future. *Computers and Chemical Engineering*, 23:667–682, 1999.

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

[Raw00] J. B. Rawlings. Tutorial overview of model predictive control. *IEEE Control Systems Magazine*, 20:38–52, 2000.

[RR00] C. V. Rao and J. B. Rawlings. Linear programming and model predictive control. *Journal of Process Control*, 10:283–289, 2000.

[RWR98] C. V. Rao, S. J. Wright, and J. B. Rawlings. Application of interior-point methods in model predictive control. *Journal of Optimization Theory and Applications*, 99:723–757, 1998.

[Wri93] S. J. Wright. Interior point methods for optimal control of discrete time systems. *Journal of Optimization Theory and Applications*, 77:161–187, 1993.

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

[ZM00] A. Zheng and M. Morari. Robust control of linear time-varying systems with constraints. *International Journal of Robust and Nonlinear Control*, 10:1063–1078, 2000.

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

A Mehrotra's predictor-corrector method

In this appendix we describe Mehrotra's method, one of the most popular algorithms for linear programming. We consider LPs of the form

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Gx \preceq g \\ & Hx = h. \end{array}$$

The variables is $x \in \mathbf{R}^n$. The problem data are $c \in \mathbf{R}^n$, $G \in \mathbf{R}^{m\times n}$, $g \in \mathbf{R}^m$, $H \in \mathbf{R}^{p\times n}$, $h \in \mathbf{R}^p$. We assume that

$$\mathbf{Rank}\, H = p, \quad \mathbf{Rank}\left(\left[\begin{array}{c} G \\ H \end{array}\right]\right) = n. \tag{29}$$

We can also define a slack variable $s \in \mathbf{R}^m$ for the inequality constraint, and write the LP as

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Gx + s = g \\ & Hx = h \\ & s \succeq 0 \end{array} \tag{30}$$

with variables x and s. The dual LP is given by

$$\begin{array}{ll} \text{maximize} & -g^T z - h^T y \\ \text{subject to} & G^T z + H^T y + c = 0 \\ & z \succeq 0, \end{array} \tag{31}$$

with variables $z \in \mathbf{R}^m$ and $y \in \mathbf{R}^p$.

A.1 Algorithm

We follow Wright [Wri97, Chapter 10], with minor modifications that reflect our notation and problem format. The algorithm starts with initial estimates s, z, x, that have to satisfy $s \succ 0$ and $z \succ 0$ (for example, $x = 0$, $s = \mathbf{1}$, $z = \mathbf{1}$), and repeats the following steps.

1. *Evaluate stopping criteria.* Terminate if the following four conditions are satisfied

$$\begin{array}{rcl} \|Gx + s - g\| & \leq & \epsilon_{\text{feas}}(1 + \|g\|) \\ \|Hx - h\| & \leq & \epsilon_{\text{feas}}(1 + \|h\|) \\ \|G^T z + H^T y + c\| & \leq & \epsilon_{\text{feas}}(1 + \|c\|) \\ c^T x + g^T z + h^T y & \leq & \max\{\epsilon_{\text{abs}}, -\epsilon_{\text{rel}}(c^T x), -\epsilon_{\text{rel}}(g^T z + h^T y)\}, \end{array}$$

   where ϵ_{feas}, ϵ_{abs}, ϵ_{rel} are given positive tolerances, or if a specified maximum allowable number of iterations is reached. Otherwise go to step 2.

2. *Compute the affine scaling directions* Δx^{aff}, Δy^{aff}, Δz^{aff}, Δs^{aff}, by solving the linear set of equations

$$\begin{bmatrix} \mathbf{diag}(z) & \mathbf{diag}(s) & 0 & 0 \\ I & 0 & 0 & G \\ 0 & 0 & 0 & H \\ 0 & G^T & H^T & 0 \end{bmatrix} \begin{bmatrix} \Delta s^{\text{aff}} \\ \Delta z^{\text{aff}} \\ \Delta y^{\text{aff}} \\ \Delta x^{\text{aff}} \end{bmatrix} = \begin{bmatrix} -\mathbf{diag}(s)z \\ -(Gx + s - g) \\ -(Hx - h) \\ -(H^T y + G^T z + c) \end{bmatrix}. \tag{32}$$

3. *Compute the centering-plus-corrector directions* Δx^{cc}, Δy^{cc}, Δz^{cc}, Δs^{cc}, by solving the set of linear equations

$$\begin{bmatrix} \mathbf{diag}(z) & \mathbf{diag}(s) & 0 & 0 \\ I & 0 & 0 & G \\ 0 & 0 & 0 & H \\ 0 & G^T & H^T & 0 \end{bmatrix} \begin{bmatrix} \Delta s^{\text{cc}} \\ \Delta z^{\text{cc}} \\ \Delta y^{\text{cc}} \\ \Delta x^{\text{cc}} \end{bmatrix} = \begin{bmatrix} \sigma\mathbf{1} - \mathbf{diag}(\Delta s^{\text{aff}})\Delta z^{\text{aff}} \\ 0 \\ 0 \\ 0 \end{bmatrix}, \tag{33}$$

where

$$\sigma = \frac{((s + \alpha_x \Delta s^{\mathrm{aff}})^T (z + \alpha_z \Delta z^{\mathrm{aff}}))^3}{m(s^T z)^2}$$

and

$$\begin{aligned} \alpha_x &= \max\{\alpha \in [0,1] \mid s + \alpha \Delta s^{\mathrm{aff}} \geq 0\} \\ \alpha_z &= \max\{\alpha \in [0,1] \mid z + \alpha \Delta z^{\mathrm{aff}} \geq 0\}. \end{aligned}$$

4. *Update the primal and dual variables:*

$$x := x + \alpha_x \Delta x, \quad s := s + \alpha_x \Delta s, \quad y := y + \alpha_z \Delta y, \quad z := z + \alpha_z \Delta z$$

where

$$\Delta x = \Delta x^{\mathrm{aff}} + \Delta x^{\mathrm{cc}}, \quad \Delta s = \Delta s^{\mathrm{aff}} + \Delta s^{\mathrm{cc}}, \quad \Delta y = \Delta y^{\mathrm{aff}} + \Delta y^{\mathrm{cc}}, \quad \Delta z = \Delta z^{\mathrm{aff}} + \Delta z^{\mathrm{cc}},$$

and

$$\begin{aligned} \alpha_x &= \min\{1, 0.99 \max\{\alpha \geq 0 \mid s + \alpha \Delta s \succeq 0\}\} \\ \alpha_z &= \min\{1, 0.99 \max\{\alpha \geq 0 \mid z + \alpha \Delta z \succeq 0\}\}. \end{aligned}$$

Go to step 1.

A.2 Discussion

Starting point

The algorithm can start at any x, y, z, s, as long as $s \succ 0$ and $z \succ 0$. We say the starting point is *strictly feasible* if it satisfies $s \succ 0$, $z \succ 0$, and, in addition,

$$Gx + s = h, \quad Hx = h, \quad G^T z + H^T y + c = 0.$$

If we start at a strictly feasible initial point, then all iterates remain strictly feasible. Indeed, if x, s, y and z are feasible, then the last three entries of the righthand side of (32) are zero. Therefore, the affine search directions satisfy

$$G\Delta x^{\mathrm{aff}} + \Delta s^{\mathrm{aff}} = 0, \quad H\Delta x^{\mathrm{aff}} = 0, \quad H^T \Delta y^{\mathrm{aff}} + G^T \Delta z^{\mathrm{aff}} = 0.$$

From (33), it is clear that the centering-plus-corrector directions satisfy similar expressions. Therefore the sum of both directions, used in Step 4, satisfies

$$G\Delta x + \Delta s = 0, \quad H\Delta x = 0, \quad H^T \Delta y + G^T \Delta z = 0.$$

The step size selection in Step 4 then guarantees that the new iterates $x + \alpha \Delta x$, $s + \alpha_x \Delta s$, $y + \alpha_z \Delta y$, $z + \alpha_z \Delta z$ are also strictly feasible.

Stopping criteria

If we start with feasible initial values, the first three stopping conditions are always satisfied. The fourth inequality is satisfied if at least one of the following conditions holds

- $c^T x + g^T z + h^T y \le \epsilon_{\text{abs}}$. If x, z, y are feasible, this means that the duality gap is less than ϵ_{abs}, which guarantees
$$c^T x - p^\star \le \epsilon_{\text{abs}}, \quad p^\star + g^T z + h^T y \le \epsilon_{\text{abs}},$$
where $p^\star$ is the optimal value of (30). *i.e.*, the absolute accuracy is better than ϵ_{abs}.
- $c^T x < 0$ and $(c^T x + g^T z + h^T y)/|c^T x| \le \epsilon_{\text{rel}}$. This guarantes that the relative accuracy is better than ϵ_{rel}, *i.e.*,
$$\frac{c^T x - p^\star}{|p^\star|} \le \epsilon_{\text{rel}}, \quad \frac{p^\star + g^T z + h^T y}{|p^\star|} \le \epsilon_{\text{rel}}.$$
- $-g^T z - h^T y > 0$ and $(c^T x + g^T z + h^T y)/(-g^T z - h^T y) \le \epsilon_{\text{rel}})$. In this case the relative accuracy is also certainly less than ϵ_{rel}.

If we start at infeasible initial points, the algorithm will attempt to simultaneously achieve primal and dual feasibility and optimality. If the problem is primal or dual infeasible, then the four inequalities in Step 1 will never be satisfied, so the algorithm simply terminates when the maximum number of iterations is exceeded. General-purpose implementations use more sophisticated stopping criteria that include more graceful tests for identifying infeasible problems. The LPs considered in this paper, however, are always strictly primal and dual feasible (and, indeed, feasible starting points are readily obtained), so the simple criteria in Step 1 are adequate for our purposes.

Computing search directions

The main computation at each iteration is the solution of the two linear equations (32) and (33), which both have the form

$$\begin{bmatrix} \mathbf{diag}(z) & \mathbf{diag}(s) & 0 & 0 \\ I & 0 & 0 & G \\ 0 & 0 & 0 & H \\ 0 & G^T & H^T & 0 \end{bmatrix} \begin{bmatrix} \Delta s \\ \Delta z \\ \Delta y \\ \Delta x \end{bmatrix} = \begin{bmatrix} r_s \\ r_z \\ r_y \\ r_x \end{bmatrix}.$$

We can simplify the system by eliminating Δs from the first equation, *i.e.*, solve

$$\begin{bmatrix} -\mathbf{diag}(z)^{-1}\mathbf{diag}(s) & 0 & G \\ 0 & 0 & H \\ G^T & H^T & 0 \end{bmatrix} \begin{bmatrix} \Delta z \\ \Delta y \\ \Delta x \end{bmatrix} = \begin{bmatrix} r_z - \mathbf{diag}(z)^{-1} r_s \\ r_y \\ r_x \end{bmatrix}, \tag{34}$$

and then set $\Delta s = \mathbf{diag}(z)^{-1}(r_s - \mathbf{diag}(s)\Delta z)$.

If the rank conditions (29) are satisfied, then the coefficient matrix is nonsingular, as can be seen as follows. We have to show that

$$\begin{bmatrix} -\mathbf{diag}(z)^{-1}\,\mathbf{diag}(s) & 0 & G \\ 0 & 0 & H \\ G^T & H^T & 0 \end{bmatrix} \begin{bmatrix} \Delta z \\ \Delta y \\ \Delta x \end{bmatrix} = 0 \qquad (35)$$

is only possible if $\Delta z = 0$, $\Delta y = 0$, $\Delta x = 0$. Suppose $V \in \mathbf{R}^{n\times(n-p)}$ spans the nullspace of H, *i.e.*, $HV = 0$ and $\mathbf{Rank}\, V = n-p$. The second equation of (35) means $\Delta x = V\Delta w$ for some Δw. The first equation implies $\Delta z = \mathbf{diag}(z)\,\mathbf{diag}(s)^{-1}GV\Delta w$. Substituting in the third equation gives

$$G^T\,\mathbf{diag}(z)\,\mathbf{diag}(s)^{-1}GV\Delta w + H^T\Delta y = 0.$$

Premultiplying with V^T yields

$$V^TG^T\,\mathbf{diag}(z)\,\mathbf{diag}(s)^{-1}GV\Delta w = 0.$$

This is only possible if $GV\Delta w = 0$, *i.e.*,

$$\begin{bmatrix} G \\ H \end{bmatrix} \Delta x = 0.$$

Since the matrix on the left has full rank, we must have $\Delta x = 0$. The first equation then implies $\Delta z = 0$, and the third equation implies $H^T\Delta y = 0$. Since $\mathbf{Rank}\, H = p$, $\Delta y = 0$.

Overall complexity

When applied to an LP that is primal and dual feasible, Mehrotra's method typically converges in about 10–50 iterations, almost independently of the problem dimensions and data. The main computation in each iteration is the solution of the two linear systems (32) and (33). Both systems have the same coefficient matrix, and it is well known from linear algebra that the cost of solving several linear systems with the same coefficient matrix and different right-hand sides, is roughly equal to the cost of solving a single linear system. In other words, for large problems, the cost of solving (32) and (33) is roughly equal to the cost of solving one of the two equations. As a practical rule of thumb, we can therefore say that the cost of solving the LPs (30) and (31) equals the cost of solving about 10–50 sets of linear equations of the form

$$\begin{bmatrix} -D & 0 & G \\ 0 & 0 & H \\ G^T & H^T & 0 \end{bmatrix} \begin{bmatrix} \Delta z \\ \Delta y \\ \Delta x \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix},$$

where the matrix D is positive diagonal with values that change at each iteration.

B Elimination of v in (24)

We first prove (26). $\mathbf{Q}$ is defined as

$$\begin{aligned} \mathbf{Q} &= \mathbf{C}_0^T\mathbf{D}_0\mathbf{C}_0 + \mathbf{C}^T\mathbf{D}\mathbf{C} - (\mathbf{C}_0^T\tilde{\mathbf{D}}_0\mathbf{E} - \mathbf{C}^T\tilde{\mathbf{D}})(\mathbf{E}^T\mathbf{D}_0\mathbf{E} + \mathbf{D})^{-1}(\mathbf{E}^T\tilde{\mathbf{D}}_0\mathbf{C}_0 - \tilde{\mathbf{D}}\mathbf{C}) \\ &= \mathbf{C}_0^T\mathbf{D}_0\mathbf{C}_0 + \mathbf{C}^T\mathbf{D}\mathbf{C} - (\mathbf{C}_0^T\tilde{\mathbf{D}}_0\mathbf{E} - \mathbf{C}^T\tilde{\mathbf{D}})(\mathbf{D}^{-1} - \mathbf{D}^{-1}\mathbf{E}^T(\sum_{i=0}^{p}\mathbf{D}_i^{-1})^{-1}\mathbf{E}\mathbf{D}^{-1})(\mathbf{E}^T\tilde{\mathbf{D}}_0\mathbf{C}_0 - \tilde{\mathbf{D}}\mathbf{C}). \end{aligned}$$

We expand the righthand side and make use of the fact that diagonal matrices commute, to write this expression as

$$\mathbf{Q} = \mathbf{C}_0^T\mathbf{D}_0\mathbf{C}_0 + \mathbf{C}^T(\mathbf{D} - \tilde{\mathbf{D}}^2\mathbf{D}^{-1})\mathbf{C} + \mathbf{C}_0^T\left(-\tilde{\mathbf{D}}_0^2(\sum_{i=1}^p \mathbf{D}_i^{-1}) + \tilde{\mathbf{D}}_0^2(\sum_{i=1}^p \mathbf{D}_i^{-1})^2(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\right)\mathbf{C}_0$$
$$+ (\sum_{i=1}^p \mathbf{C}_i^T\tilde{\mathbf{D}}_i\mathbf{D}_i^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}(\sum_{i=1}^p \mathbf{D}_i^{-1}\tilde{\mathbf{D}}_i\mathbf{C}_i) + \mathbf{C}_0^T\tilde{\mathbf{D}}_0\left(I - (\sum_{i=1}^p \mathbf{D}_i^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\right)(\sum_{i=1}^p \tilde{\mathbf{D}}_i\mathbf{D}_i^{-1}\mathbf{C}_i)$$
$$+ (\sum_{i=1}^p \mathbf{C}_i^T\tilde{\mathbf{D}}_i\mathbf{D}_i^{-1})\left(I - (\sum_{i=1}^p \mathbf{D}_i^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\right)\tilde{\mathbf{D}}_0\mathbf{C}_0.$$

(Note that $\mathbf{E}\mathbf{D}^{-1}\mathbf{E}^T = \sum_{i=1}^p \mathbf{D}_i^{-1}$ and $\mathbf{C}^T\tilde{\mathbf{D}}\mathbf{D}^{-1}\mathbf{E}^T = \sum_{i=1}^p \mathbf{C}_i^T\tilde{\mathbf{D}}_i\mathbf{D}_i^{-1}$.) Next we move the factors $(\sum_{i=0}^p \mathbf{D}_i^{-1})^T$ outside of the middle parentheses in the third, fifth, and sixth terms of the sum, and simplify:

$$\begin{aligned}\mathbf{Q} \;=\;& \mathbf{C}_0^T\mathbf{D}_0\mathbf{C}_0 + \mathbf{C}^T(\mathbf{D} - \tilde{\mathbf{D}}^2\mathbf{D}^{-1})\mathbf{C} - \mathbf{C}_0^T\tilde{\mathbf{D}}_0^2\mathbf{D}_0^{-1}(\sum_{i=1}^p \mathbf{D}_i^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\mathbf{C}_0\\ &+ (\sum_{i=1}^p \mathbf{C}_i^T\tilde{\mathbf{D}}_i\mathbf{D}_i^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}(\sum_{i=1}^p \tilde{\mathbf{D}}_i\mathbf{D}_i^{-1}\mathbf{C}_i) + (\mathbf{C}_0^T\tilde{\mathbf{D}}_0\mathbf{D}_0^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}(\sum_{i=1}^p \tilde{\mathbf{D}}_i\mathbf{D}_i^{-1}\mathbf{C}_i)\\ &+ (\sum_{i=1}^p \mathbf{C}_i^T\tilde{\mathbf{D}}_i\mathbf{D}_i^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}(\tilde{\mathbf{D}}_0\mathbf{D}_0^{-1}\mathbf{C}_0)\end{aligned}$$

By adding and subtracting $\mathbf{C}_0^T\tilde{\mathbf{D}}_0^2\mathbf{D}_0^{-2}(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\mathbf{C}_0$ we can complete the square given by last three terms. This yields

$$\begin{aligned}\mathbf{Q} \;=\;& \mathbf{C}_0^T\mathbf{D}_0\mathbf{C}_0 + \mathbf{C}^T(\mathbf{D} - \tilde{\mathbf{D}}^2\mathbf{D}^{-1})\mathbf{C} - \mathbf{C}_0^T\tilde{\mathbf{D}}_0^2\mathbf{D}_0^{-1}(\sum_{i=1}^p \mathbf{D}_i^{-1} + \mathbf{D}_0^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\mathbf{C}_0\\ &+ (\sum_{i=0}^p \mathbf{C}_i^T\tilde{\mathbf{D}}_i\mathbf{D}_i^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}(\sum_{i=0}^p \tilde{\mathbf{D}}_i\mathbf{D}_i^{-1}\mathbf{C}_i)\\ =\;& \sum_{i=0}^p \mathbf{C}_i^T(\mathbf{D}_i - \tilde{\mathbf{D}}_i^2\mathbf{D}_i^{-1})\mathbf{C}_i + (\sum_{i=0}^p \mathbf{C}_i^T\tilde{\mathbf{D}}_i\mathbf{D}_i^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}(\sum_{i=0}^p \tilde{\mathbf{D}}_i\mathbf{D}_i^{-1}\mathbf{C}_i).\end{aligned}$$

This proves (26). The proof of equation (27) is similar: $\mathbf{d}$ is given by

$$\begin{aligned}\mathbf{d} \;=\;& \mathbf{C}_0^T\tilde{\mathbf{D}}_0\mathbf{1} - (\mathbf{C}_0^T\tilde{\mathbf{D}}_0\mathbf{E} - \mathbf{C}^T\tilde{\mathbf{D}})(\mathbf{E}^T\mathbf{D}_0\mathbf{E} + \mathbf{D})^{-1}(\mathbf{E}^T\mathbf{D}_0\mathbf{1})\\ =\;& \mathbf{C}_0^T\tilde{\mathbf{D}}_0\mathbf{1} - (\mathbf{C}_0^T\tilde{\mathbf{D}}_0\mathbf{E} - \mathbf{C}^T\tilde{\mathbf{D}})(\mathbf{D}^{-1} - \mathbf{D}^{-1}\mathbf{E}^T(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\mathbf{E}\mathbf{D}^{-1})\mathbf{E}^T\mathbf{D}_0\mathbf{1}.\end{aligned}$$

We have $\mathbf{E}\mathbf{D}^{-1}\mathbf{E}^T = \sum_{i=1}^p \mathbf{D}_i^{-1}$, $\mathbf{C}^T\tilde{\mathbf{D}}\mathbf{D}^{-1}\mathbf{E}^T = \sum_{i=1}^p \mathbf{C}_i^T\tilde{\mathbf{D}}_i\mathbf{D}_i^{-1}$, so after expanding the righthand side we obtain

$$\mathbf{d} \;=\; \mathbf{C}_0^T\tilde{\mathbf{D}}_0\mathbf{1} - \mathbf{C}_0^T\tilde{\mathbf{D}}_0(\mathbf{D}_0\sum_{i=1}^p \mathbf{D}_i^{-1} + \mathbf{D}_0(\sum_{i=1}^p \mathbf{D}_i^{-1})^2(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1})\mathbf{1}$$

$$
\begin{aligned}
&\quad - (\sum_{i=1}^p \mathbf{C}_i^T \tilde{\mathbf{D}}_i \mathbf{D}_i^{-1})(\mathbf{D}_0 - \mathbf{D}_0(\sum_{i=1}^p \mathbf{D}_i^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1})\mathbf{1} \\
&= \mathbf{C}_0^T \tilde{\mathbf{D}}_0 \mathbf{1} - \mathbf{C}_0^T \tilde{\mathbf{D}}_0 (\sum_{i=1}^p \mathbf{D}_i^{-1} + \mathbf{D}_0(\sum_{i=1}^p \mathbf{D}_i^{-1})^2 - \mathbf{D}_0(\sum_{i=1}^p \mathbf{D}_i^{-1})^2)(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\mathbf{1} \\
&\quad + (\sum_{i=1}^p \mathbf{C}_i^T \tilde{\mathbf{D}}_i \mathbf{D}_i^{-1})(I + \mathbf{D}_0 \sum_{i=1}^p \mathbf{D}_i^{-1} - \mathbf{D}_0 \sum_{i=1}^p \mathbf{D}_i^{-1}\Big) (\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\mathbf{1} \\
&= (\sum_{i=0}^p \mathbf{C}_i^T \tilde{\mathbf{D}}_i \mathbf{D}_i^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\mathbf{1}.
\end{aligned}
$$

Finally, γ is given by

$$
\begin{aligned}
\gamma &= \mathbf{Tr}\,\mathbf{D}_0 - (\mathbf{1}^T\mathbf{D}_0\mathbf{E})(\mathbf{E}^T\mathbf{D}_0\mathbf{E} + \mathbf{D})^{-1}(\mathbf{E}^T\mathbf{D}_0\mathbf{1}) \\
&= \mathbf{Tr}\,\mathbf{D}_0 - \mathbf{1}^T\mathbf{D}_0\mathbf{E}(\mathbf{D}^{-1} - \mathbf{D}^{-1}\mathbf{E}^T(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\mathbf{E}\mathbf{D}^{-1})\mathbf{E}^T\mathbf{D}_0\mathbf{1} \\
&= \mathbf{Tr}\left(\mathbf{D}_0 - \mathbf{D}_0^2 \sum_{i=1}^p \mathbf{D}_i^{-1} + \mathbf{D}_0^2(\sum_{i=1}^p \mathbf{D}_i^{-1})^2(\mathbf{D}_0^{-1} + \sum_{i=1}^p \mathbf{D}_i^{-1})^{-1}\right) \\
&= \mathbf{Tr}\left((I + \mathbf{D}_0(\sum_{i=1}^p \mathbf{D}_i^{-1}) - \mathbf{D}_0(\sum_{i=1}^p \mathbf{D}_i^{-1}) - \mathbf{D}_0^2(\sum_{i=1}^p \mathbf{D}_i^{-1})^2 + \mathbf{D}_0^2(\sum_{i=1}^p \mathbf{D}_i^{-1})^2)(\sum_{i=0}^p \mathbf{D}_i^{-1})\right) \\
&= \mathbf{Tr}\left((\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\right),
\end{aligned}
$$

which proves (29). The expressions for $\mathbf{r}_{14}$ and $\mathbf{r}_{16}$ can be verified in a similar way:

$$
\begin{aligned}
\mathbf{r}_{14} &= \mathbf{r}_9 + (\mathbf{C}_0^T\tilde{\mathbf{D}}_0\mathbf{E} - \mathbf{C}^T\tilde{\mathbf{D}})(\mathbf{E}^T\mathbf{D}_0\mathbf{E} + \mathbf{D})^{-1}\mathbf{r}_{11} \\
&= \mathbf{r}_9 + (\mathbf{C}_0^T\tilde{\mathbf{D}}_0\mathbf{E} + \mathbf{C}^T\tilde{\mathbf{D}})(\mathbf{D}^{-1} - \mathbf{D}^{-1}\mathbf{E}^T(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\mathbf{E}\mathbf{D}^{-1})\mathbf{r}_{11} \\
&= r_9 + \mathbf{C}_0^T\tilde{\mathbf{D}}_0(I - (\sum_{i=1}^p \mathbf{D}_i^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1})\mathbf{E}\mathbf{D}^{-1}\mathbf{r}_{11} + (\mathbf{C}^T\tilde{\mathbf{D}}\mathbf{D}^{-1}\mathbf{E}^T)(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\mathbf{E}\mathbf{D}^{-1}\mathbf{r}_{11} \\
&\quad - \mathbf{C}^T\tilde{\mathbf{D}}\mathbf{D}^{-1}\mathbf{r}_{11} \\
&= \mathbf{r}_9 + (\mathbf{C}_0^T\mathbf{D}_0^{-1} + \mathbf{C}^T\tilde{\mathbf{D}}\mathbf{D}^{-1}\mathbf{E}^T)(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}(\mathbf{E}\mathbf{D}^{-1}\mathbf{r}_{11}) - \mathbf{C}^T\tilde{\mathbf{D}}\mathbf{D}^{-1}\mathbf{r}_{11} \\
r_{16} &= r_{12} + \mathbf{1}^T\mathbf{D}_0\mathbf{E}(\mathbf{E}^T\mathbf{D}_0\mathbf{E} + \mathbf{D})^{-1}\mathbf{r}_{11} \\
&= r_{12} + \mathbf{1}^T\mathbf{D}_0\mathbf{E}(\mathbf{D}^{-1} - \mathbf{D}^{-1}\mathbf{E}^T(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}\mathbf{E}\mathbf{D}^{-1})\mathbf{r}_{11} \\
&= r_{12} + \mathbf{1}^T\mathbf{D}_0(I - (\sum_{i=1}^p \mathbf{D}_i^{-1})(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1})\mathbf{E}\mathbf{D}^{-1}\mathbf{r}_{11} \\
&= r_{12} + \mathbf{1}^T(\sum_{i=0}^p \mathbf{D}_i^{-1})^{-1}(\mathbf{E}\mathbf{D}^{-1}\mathbf{r}_{11}).
\end{aligned}
$$