PREPRINT ANL/MCS-P664-0597,
MATHEMATICS AND COMPUTER SCIENCE DIVISION, ARGONNE NATIONAL LABORATORY

Application of Interior-Point Methods to Model Predictive Control

Christopher V. Rao[*], Stephen J. Wright[†], and James B. Rawlings[‡]

Abstract

We present a structured interior-point method for the efficient solution of the optimal control problem in model predictive control (MPC). The cost of this approach is linear in the horizon length, compared with cubic growth for a naive approach. We use a discrete-time Riccati recursion to solve the linear equations efficiently at each iteration of the interior-point method, and show that this recursion is numerically stable. We demonstrate the effectiveness of the approach by applying it to three process control problems.

1 Introduction

Model predictive control (MPC) is an optimal control-based strategy that uses a plant model to predict the effect of an input profile on the evolving state of the plant. At each step of MPC, an optimal control problem with Bolza objectives is solved and its optimal input profile is implemented until another plant measurement becomes available. The updated plant information is used to formulate and solve a new optimal control problem—thereby providing feedback from the plant to the model—and the process is repeated. This strategy yields a receding horizon control formulation.

The MPC methodology is appealing to the practitioner because input and state constraints can be explicitly accounted for in the controller. A practical disadvantage is its computational cost, which has tended to limit MPC applications to linear processes with relatively slow dynamics. For such problems, the optimal control problem to be solved at each stage of MPC is a convex quadratic program. While robust and efficient software exists for the solution of unstructured convex quadratic programs, significant improvements often can be made by exploiting the structure of the MPC subproblem.

When input and state constraints are not present, MPC with an infinite horizon is simply the well-known linear-quadratic regulator problem. Even when constraints are present, the

[*]Department of Chemical Engineering, University of Wisconsin-Madison. rao@bevo.che.wisc.edu

[†](Author to whom correspondence should be addressed) Mathematics and Computer Science Division, Argonne National Laboratory. wright@mcs.anl.gov

[‡]Department of Chemical Engineering, University of Wisconsin-Madison. jbraw@che.wisc.edu.

infinite-horizon MPC problem generally reduces to a linear-quadratic regulator after a certain number of stages (see [4, 21, 24]) and therefore can be recast as a finite-dimensional quadratic program. Since this quadratic program can be large, with many stages, it is important that algorithms be efficient for problems with long horizons.

Unconstrained discrete-time linear-quadratic optimal control problems can be solved by using a discrete-time Riccati equation. The computational cost of this algorithm is linear in the horizon length N. A different formulation obtained by eliminating the state variables results in an unconstrained quadratic function whose Hessian is dense, with dimensions that grow linearly in N. The cost of minimizing this quadratic function is cubic in N, making it uncompetitive with the Riccati approach in general. There is a third option, however—an optimization formulation in which the states are retained explicitly as unknowns in the optimization and the model equation is retained as a constraint. The optimality conditions for this formulation reveal that the adjoint variables are simply the Lagrange multipliers for the model equation and that the problem can be solved by factoring a matrix whose dimension again grows linearly with N. In this formulation, however, the matrix is banded, with a bandwidth independent of N, so the cost of the factorization is linear rather than cubic in N. The discrete-time Riccati equation can be interpreted as a block factorization scheme applied to this matrix.

Traditionally, the discrete-time Riccati equation is obtained by using dynamic programming to solve the unconstrained linear optimal control problem. The essential idea in dynamic programming is to work stage-by-stage through the problem in reverse order, starting with the final stage N. The optimization problem reduces to a simpler problem at each stage. (See Berksekas [2] for further details.) Block factorization, like dynamic programming, exploits the multi-staged nature of the optimization problem. The key difference is that the block factorization approach tackles the problem explicitly, whereas dynamic programming tackles the problem semi-implicitly by using Bellman's principle of optimality. The explicit treatment allows greater flexibility, however, since the block factorization approach retains its inherent structure even when inequality constraints are added to the formulation.

When constraints are present, the scheme for unconstrained problems must be embedded in an algorithmic framework that determines which of the inequalities are active and which are inactive at the optimum. At each iteration of the outer algorithm, however, the main computational operation is the solution of a set of linear equations whose structure is very like that encountered in the unconstrained problem. Hence, the cost of performing each iteration of the outer algorithm is linear in the number of stages N. This observation has been made by numerous authors, in the context of outer algorithms based on both active-set and interior-point methods. Glad and Jonson [9] and Arnold et al. [1] demonstrate that the factorization of a structured Lagrangian in an optimal control problem with a Bolza objective for an active set framework yields a Riccati recursion. Wright [25, 27], Steinbach [23], and Lim et al. [12] investigate the Bolza control problem in an interior-point framework.

In this paper we present an MPC algorithm based on an interior-point method, in which a block factorization is used at each iteration to obtain the search direction for the interior-point method. Our work differs from earlier contributions in that the formulation of the optimal control problem is tailored to the MPC application, the interior-point algorithm is based on Mehrotra's algorithm [15] (whose practical efficiency on general linear and quadratic programming problems is well documented), and the linear system at each interior-point iteration is solved efficiently by a Riccati recursion. We compare our approach with the alternative of using the model equation to eliminate the states, yielding a dense quadratic program in the input variables alone, and present results obtained for three large industrial problems.

We use order notation in the following (standard) way: If a matrix, vector, or scalar quantity M is a function of another matrix, vector, or scalar quantity E, we write $M = O(\|E\|)$ if there is a constant β such that $\|M\| \leq \beta\|E\|$ for all $\|E\|$ sufficiently small. We write $M = \Theta(\|E\|)$ if there is a constant β such that $\|E\|/\beta \leq \|M\| \leq \beta\|E\|$.

We say that a matrix is "positive diagonal" if it is diagonal with positive diagonal elements. The term "nonnegative diagonal" is defined correspondingly. We use SPD as an abbreviation for "symmetric positive definite" and SPSD as an abbreviation for "symmetric positive semidefinite."

2 Model Predictive Control

2.1 Infinite-Horizon Problem

The fundamental formulation of the linear model predictive controller is the following infinite-dimensional convex quadratic program:

$$\min_{u,x} \ \Phi(u,x) = \frac{1}{2}\sum_{k=0}^{\infty}(x_k^T Q x_k + u_k^T R u_k + \Delta u_k^T S \Delta u_k), \tag{1}$$

subject to the following constraints:

$$x_0 = \hat{x}_j, \qquad x_{k+1} = A x_k + B u_k, \tag{2a}$$

$$D u_k \leq d, \qquad G \Delta u_k \leq g, \qquad H x_k \leq h, \tag{2b}$$

where $x_k \in \mathbb{R}^n$, $u_k \in \mathbb{R}^m$, and $\Delta u_k = u_k - u_{k-1}$. The vector $\hat{x}_j$ represents the current estimate of the state at discrete time j, whereas x_k represents the state at k sampling steps along the future prediction horizon and u_k represents the input at this same time. We assume that Q and S are SPSD matrices and that R is SPD.

By a suitable adjustment of the origin, the formulation (1), (2) can also account for target tracking and disturbance rejection [16]. If there is a feasible point for the constraints (2), the infinite horizon regulator formulation is stabilizing whenever (A, B) is stabilizable and $(A, Q^{1/2})$ is detectable [22].

For unstable state transition matrices, (1), (2) is ill-conditioned because the infinite horizon formulation can potentially yield unbounded solutions. To improve the conditioning of the optimization, we parameterize the input as $u_k = Lx_k + r_k$, where L is a linear stabilizing feedback gain for (A, B) [11, 20]. The system model becomes

$$x_{k+1} = (A + BL)x_k + Br_k, \tag{3}$$

where r_k is the new manipulated input. By initially specifying a stabilizing (but potentially infeasible) trajectory, we can improve the numerical conditioning of the optimization by excluding unstable solutions.

By expanding Δu_k, we transform (1), (2) into the following more tractable form:

$$\min_{u,x} \; \Phi(u, x) = \frac{1}{2} \sum_{k=0}^{\infty} (x_k^T Q x_k + u_k^T R u_k + 2x_k^T M u_k), \tag{4}$$

subject to

$$x_0 = \hat{x}_j, \qquad x_{k+1} = Ax_k + Bu_k, \tag{5a}$$

$$Du_k - Gx_k \leq d, \qquad Hx_k \leq h. \tag{5b}$$

The original formulation (1), (2) can be recovered from (4), (5) by making the following substitutions into the second formulation:

$$\hat{x}_j \leftarrow \begin{bmatrix} \hat{x}_j \\ u_{j-1} \end{bmatrix}, \quad x_k \leftarrow \begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix}, \quad u_k \leftarrow r_k, \quad A \leftarrow \begin{bmatrix} A + BL & 0 \\ L & 0 \end{bmatrix}, \quad B \leftarrow \begin{bmatrix} B \\ I \end{bmatrix},$$

$$Q \leftarrow \begin{bmatrix} Q + L^T(R+S)L & -L^T S \\ -SL & S \end{bmatrix}, \quad M \leftarrow \begin{bmatrix} L^T(R+S) \\ -S \end{bmatrix}, \quad R \leftarrow R + S,$$

$$D \leftarrow \begin{bmatrix} D \\ G \end{bmatrix}, \quad G \leftarrow \begin{bmatrix} -DL & 0 \\ -L & G \end{bmatrix}, \quad d \leftarrow \begin{bmatrix} d \\ g \end{bmatrix}, \quad H \leftarrow \begin{bmatrix} H & 0 \end{bmatrix}.$$

In the remainder of this section, we address two issues. The first is the replacement of (4), (5) by an equivalent (or similar) finite horizon problem, a step necessary for practical computation of the solution. The second issue is replacement of the constraints $Hx_k \leq h$ by so-called soft constraints. Instead of enforcing these conditions strictly, we add terms to the objective that penalize violations of these conditions. This technique is a more appropriate way of dealing with certain constraints from an engineering point of view.

2.2 Receding Horizon Regulator Formulation

The key step in reducing (4), (5) to a finite-horizon problem is the use of a linear control law to determine u_k after a certain time horizon, that is,

$$u_k = Kx_k, \qquad \text{for all } k \geq N. \tag{6}$$

With this added constraint, the states x_k, $k > N$ and the inputs u_k, $k \geq N$ are completely determined by x_N, the state at the end of the prediction horizon.

Two techniques can be used to determine the law (6). The first, due to Rawlings and Muske [18], sets $K = 0$ uniformly in (6) and produces an approximate solution to (4), (5). With this substitution, we have

$$\frac{1}{2}\sum_{k=N}^{\infty}(x_k^T Q x_k + u_k^T R u_k + 2x_k^T M u_k) = \frac{1}{2}\sum_{k=N}^{\infty} x_k^T \bar{Q} x_k. \tag{7}$$

If A is stable, this sum is equal to $(1/2)x_N^T \bar{Q} x_N$, where $\bar{Q}$ is the solution of the matrix Lyapunov equation

$$\bar{Q} - A^T \bar{Q} A = Q. \tag{8}$$

If A is unstable, the sum (7) may be infinite, so we impose a stabilizing constraint to derive any useful information from the solution of the model problem. The Schur decomposition of A can be used to construct a basis for the stable subspace of A. If this decomposition is

$$A = UTU^T = \left[\begin{array}{cc} U_s & U_u \end{array}\right] \left[\begin{array}{cc} T_{11} & T_{12} \\ 0 & T_{22} \end{array}\right] \left[\begin{array}{c} U_s^T \\ U_u^T \end{array}\right],$$

where the eigenvalues of T_{11} are inside the unit circle whereas those of T_{22} are contained on or outside the unit circle, then the (orthogonal) columns of U_s span the stable subspace of A and the (orthogonal) columns of U_u span orthogonal complement of the stable subspace of A. We add the endpoint constraint

$$F x_N = 0 \qquad (\text{where } F = U_u^T) \tag{9}$$

to ensure that the unstable modes have vanished by stage N [14]. (Since the input u_k is zero for all $k \geq N$, the unstable modes also remain at zero at all subsequent stages.) The evolution of the stable modes on the infinite horizon can be accounted for by solving the following Lyapunov equation for $\bar{Q}$:

$$\bar{Q} - A_s^T \bar{Q} A_s = Q,$$

where $A_s = U_s T_{11} U_s^T$, and replacing the infinite sum with $(1/2)x_N^T \bar{Q} x_N$, as above.

In the second formulation, discussed by Keerthi [11], Sznaier and Damborg [24], Chmielewski and Manousiouthakis [4], and Scokaert and Rawlings [21], the input after stage N is parameterized with the classical linear quadratic gain K obtained from the solution of the steady-state Riccati equation. This matrix, used in conjunction with the control law (6), is the solution to the "unconstrained" version of the problem, in which the inequality constraints (5b) do not appear. By using (6), the infinite tail of the sum in (4) can be written as

$$\frac{1}{2}\sum_{k=N}^{\infty}(x_k^T Q x_k + u_k^T R u_k + 2x_k^T M u_k) = \frac{1}{2}\sum_{k=N}^{\infty}(x_k^T Q x_k + x_k^T K^T R K x_k + 2x_k^T M K x_k).$$

This infinite summation can be replaced by the single term $(1/2)x_N^T \bar{Q} x_N$, where $\bar{Q}$ is the solution of the following discrete-time algebraic Riccati equation:

$$\bar{Q} = Q + A^T \bar{Q} A - (A^T \bar{Q} B + M)(R + B^T \bar{Q} B)^{-1}(B^T \bar{Q} A + M^T). \tag{10}$$

In both formulations, the feedback law (6) is valid only if the constraints (5) are satisfied at *all* stages, including the stages $k \geq N$. Hence, we would like to implement this law only after we reach a state x_N such that the solution generated by the control law (6) and the model equation in (5a) at stages $k \geq N$ satisfies the inequalities (5b) at all such stages. We define a set $\mathcal{X}$ of states for which this property holds, as follows:

$$\mathcal{X} = \{x : H(A - BK)^l x \leq h, \ \ (DK - G)(A - BK)^l x \leq d, \ \text{ for all } l \geq 0\}.$$

where K is the optimal unconstrained linear control law obtained from the following equation:

$$K = -(R + B^T \bar{Q} B)^{-1} (B^T \bar{Q} A + M^T). \tag{11}$$

If N is chosen so that $x_N \in \mathcal{X}$, then the following finite-horizon problem is equivalent[1] to (4), (5):

$$\min_{u,x} \ \Phi(u,x) = \frac{1}{2} \sum_{k=0}^{N-1} (x_k^T Q x_k + u_k^T R u_k + 2 x_k^T M u_k) + \frac{1}{2} x_N^T \bar{Q} x_N, \tag{12}$$

subject to

$$x_0 = \hat{x}_j, \qquad x_{k+1} = Ax_k + Bu_k, \qquad k = 0, 1, 2, \ldots, \tag{13a}$$

$$u_k = Kx_k, \qquad k \geq N, \tag{13b}$$

$$Du_k - Gx_k \leq d, \quad Hx_k \leq h, \qquad k = 0, 1, 2, \ldots, N-1, \tag{13c}$$

where $\bar{Q}$ is defined in (10).

If the components of h and d are strictly positive and the "unconstrained" model is stabilizable, we can show that $\mathcal{X}$ contains 0 in its interior. Under these circumstances, there is an index N_∞ such that

$$x_k \notin \mathcal{X}, \ k < N_\infty, \qquad x_k \in \mathcal{X}, \ k \geq N_\infty. \tag{14}$$

Since N_∞ is difficult to obtain in practise and $\mathcal{X}$ is characterized by a finite number of conditions, we can solve the problem (12), (13) for some fixed value of N and then check that the states and inputs at stages $k \geq N$ continue to satisfy the inequality constraints at subsequent stages. If not, we increase N and repeat the process.

A variety of methods guarantee that the constraints are satisfied on the infinite horizon by checking a finite number of stages $k \geq N$. Scokaert and Rawlings [21] propose constructing an open ball B_x contained within the set $\mathcal{X}$, thereby allowing termination of the search when $x_k \in B_x$ for $k \geq N$. The approximation for $\mathcal{X}$ tends to be conservative, however, since the algorithm is motivated by norm-bounding arguments. A more practical method, given by Gilbert and Tan [7], is to explicitly construct the set $\mathcal{X}$. The constructive representation of $\mathcal{X}$ provides a priori an upper bound l on number of feasible stages $k \in [N, N+l]$ necessary

[1]The finite-horizon problem is also a valid approximation to (4), (5) for $K = 0$ when the endpoint constraint $Fx_N = 0$ is added to (13) and $\bar{Q}$ is defined by (8)

to guarantee that all of the subsequent stages $k \geq l + N$ are feasible. The drawback of this approach is that the algorithm for constructing the maximal sets is not guaranteed to converge for unbounded feasible regions, since $\mathcal{X}$ may be unbounded. For a compact, convex set of states, an alternative approach that circumvents having to check for constraint violations is given by Chmielewski and Manousiouthakis [4]. By examining the extremal points on the feasible region, they calculate a conservative upper bound on the N required to guarantee that the solution is feasible on the infinite horizon.

We have assumed to this point that there exists a feasible solution with respect to the input and endpoint constraints for the optimal control calculation. In the presence of side constraints (2b), it is no longer true that the constrained regulator stabilizes all possible states even when the stabilizability assumption is satisfied. When stabilization is not possible, the problem (4), (5) is an infeasible optimization problem. (In actual operation, an infeasible solution would signal a process exception condition.)

For the Rawlings-Muske formulation, enforcement of the endpoint constraint (9) often results in an infeasible optimization problem. Feasibility can often be recovered by increasing the horizon length N, but when the initial state is not stabilizable, the feasible region will continue to be empty for all N. The existence of a feasible N can easily be checked by solving the following linear program:

$$\min_{u,x,r} \; e^T r, \tag{15}$$

(where e is the vector whose entries are all 1) subject to the constraints

$$\begin{gathered} x_{k+1} = Ax_k + Bu_k, \qquad k = 0, 1, 2, \ldots, N-1, \\ Du_k - Gx_k \leq d, \quad Hx_k \leq h, \qquad k = 0, 1, 2, \ldots, N-1, \\ r - Fx_N \geq 0, \quad r + Fx_N \geq 0. \end{gathered} \tag{16}$$

A positive solution to the linear program indicates that a feasible solution does not exist and the horizon length N must be increased. If the feasibility check fails for some user supplied upper bound on the horizon length, then current state is not constrained stabilizable for the specified regulator.

2.3 Feasibility and Soft Constraints

In the formulation of the MPC problem, some state constraints are imposed by physical limitations such as valve saturation. Other constraints are less important; they may represent desired ranges of operation for the plant, for instance. In some situations, no set of inputs and states for the MPC problem may satisfy all of these constraints. Rather than having the algorithm declare infeasibility and return without a result, we prefer a solution that enforces some constraints strictly ("hard constraints"), while relaxing others and replacing them with penalties on their violation ("soft constraints").

Scokaert and Rawlings [22] replace the soft constraints with penalty terms in the objective that are a combination of ℓ_1 norms and squared ℓ_2 norms of the constraint violations.

Assuming for simplicity that all state constraints $Hx_k \le h$ in (13) are softened in this way, we obtain the following modification to the objective (12):

$$\min_{u,x,\epsilon} \; \Phi(u,x,\epsilon) = \frac{1}{2}\sum_{k=0}^{N-1}(x_k^T Q x_k + u_k^T R u_k + 2x_k^T M u_k + \epsilon_k^T Z \epsilon_k) + z^T \epsilon_k + \frac{1}{2} x_N^T \bar{Q} x_N, \tag{17}$$

where the constraint violations ϵ_k are defined by the following formulae (which replace $Hx_k \le h$):

$$Hx_k - \epsilon_k \le h, \qquad \epsilon_k \ge 0, \tag{18}$$

and the elements of the vector z are nonnegative, while Z is an SPSD matrix. It is known that when the weighting z on the ℓ_1 terms is sufficiently large (see, for example, Section 12.3 in Fletcher [6]), and when the original problem (12), (13) has a nonempty feasible region, then local minimizers of problem (12), (13) modified by (17), (18) defined above correspond to local solutions of the unmodified problem (12), (13). Under these conditions, (17) together with the constraints (18) is referred to as an exact penalty formulation of the original objective (12) with the original constraints $Hx_k \le h$. This formulation has the advantage that it can still yield a solution when the original problem (12), (13) is infeasible.

Prior to actually solving the problem, we cannot know how large the elements of z must be chosen to make the exact penalty property hold. (The threshold value depends on the optimal multipliers for the original problem (12), (13).) A conservative state-dependent upper bound for these multipliers can be obtained by exploiting the Lipschitz continuity of the quadratic program [10]. In practice, however, the exact penalty is not critical, since by definition soft constraints need not be satisfied exactly. Reasonable controller performance can often be achieved by setting $z = 0$ and choosing Z to be a positive diagonal matrix. In fact, the inclusion of the ℓ_2 term $\epsilon_k^T Z \epsilon_k$ is not needed at all for the exact penalty property to hold, but is included here to provide a little more flexibility in the modeling.

In the remainder of the paper, we work with a general form of the MPC problem, which contains all the features discussed in this section: finite horizon, endpoint constraints, and soft constraints. This general form is

$$\min_{u,x,\epsilon} \; \Phi(u,x,\epsilon) = \sum_{k=0}^{N-1} \frac{1}{2}(x_k^T Q x_k + u_k^T R u_k + 2x_k^T M u_k + \epsilon_k^T Z \epsilon_k) + z^T \epsilon_k + x_N^T \bar{Q}_N x_N, \tag{19}$$

subject to

$$\begin{aligned}
x_0 &= \hat{x}_j, \qquad \text{(fixed)} && \text{(20a)}\\
x_{k+1} &= Ax_k + Bu_k, \qquad k = 0,1,\ldots,N-1, && \text{(20b)}\\
Du_k - Gx_k &\le d, \qquad k = 0,1,\ldots,N-1, && \text{(20c)}\\
Hx_k - \epsilon_k &\le h, \qquad k = 1,2,\ldots,N, && \text{(20d)}\\
\epsilon_k &\ge 0, \qquad k = 1,2,\ldots,N, && \text{(20e)}\\
Fx_N &= 0. && \text{(20f)}
\end{aligned}$$

(Note that (20f) is required only when we choose the parameterization $K = 0$.) We assume throughout that the matrices in (19) satisfy the properties

$$R \text{ is PSD}, \qquad Z \text{ is SPSD}, \qquad \begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} \text{ is SPSD.} \tag{21}$$

Note that the last property holds for the matrices considered in Section 2.1, since in making the substitutions to obtain the form (4) we obtain

$$\begin{aligned} \begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} &\leftarrow \begin{bmatrix} Q + L^T(R+S)L & -L^TS & L^T(R+S) \\ -SL & S & -S \\ (R+S)L & -S & (R+S) \end{bmatrix} \\ &= \begin{bmatrix} Q & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} L^T \\ -I \\ I \end{bmatrix} S \begin{bmatrix} L & -I & I \end{bmatrix} + \begin{bmatrix} L^T \\ 0 \\ I \end{bmatrix} R \begin{bmatrix} L & 0 & I \end{bmatrix}, \end{aligned}$$

which is a sum of SPSD matrices and is therefore itself SPSD.

3 The Interior-Point Method

In this section, we describe our interior-point-based approach for solving the MPC problem (19), (20). We start with a general description of the interior-point method of choice for linear and convex quadratic programming: Mehrotra's predictor-corrector algorithm. The remaining sections describe the specialization of this approach to MPC, including the use of the Riccati approach to solve the linear subproblem, handling of endpoint constraints, and hot starting.

3.1 Mehrotra's Predictor-Corrector Algorithm

Active set methods have proved to be efficient for solving quadratic programs with general constraints. The interior-point approach has proved to be an attractive alternative when the problems are large and convex. In addition, this approach has the advantage that the system of linear equations to be solved at each iterate has the same dimension and structure throughout the algorithm, making it possible to exploit any structure inherent in the problem. The most widely used interior-point algorithms do not require a feasible starting point to be specified. In fact, they usually generate infeasible iterates, attaining feasibility only in the limit. From a theoretical viewpoint, interior-point methods exhibit polynomial complexity, in contrast to the exponential complexity of active-set approaches.

In this section, we sketch an interior-point method for general convex quadratic programming problems and discuss its application to the specific problem (19). A more complete description is given by Wright [28].

Consider the following convex quadratic program

$$\min_w \Phi(w) = \frac{1}{2} w^T Q w + c^T w, \quad \text{subject to } Fw = g,\ Cw \le d, \tag{22}$$

where Q is an SPSD matrix. The Karush-Kuhn-Tucker (KKT) conditions for optimality are that there exist vectors π^* and λ^* such that the following conditions are satisfied for $(w, \pi, \lambda) = (w^*, \pi^*, \lambda^*)$:

$$\begin{aligned} Qw + F^T\pi + C^T\lambda + c &= 0, \\ -Fw + g &= 0, \\ -Cw + d &\geq 0, \\ \lambda &\geq 0, \\ \lambda_j(-Cw + d)_j &= 0, \qquad j = 1, 2, \ldots, m, \end{aligned}$$

where m is the number of rows in the matrix C. Because the objective function is convex, the KKT conditions are both necessary and sufficient for optimality. By introducing a vector t of slacks for the constraint $Cw \leq d$, we can rewrite these conditions in a slightly more convenient form:

$$\mathcal{F}(w, \pi, \lambda, t) = \begin{bmatrix} Qw + F^T\pi + C^T\lambda + c \\ -Fw + g \\ -Cw - t + d \\ T\Lambda e \end{bmatrix} = 0, \tag{23a}$$

$$(\lambda, t) \geq 0, \tag{23b}$$

where T and Λ are diagonal matrices defined by

$$T = \operatorname{diag}(t_1, t_2, \ldots, t_m), \qquad \Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_m),$$

and $e = (1, 1, \ldots, 1)^T$.

Primal-dual interior-point methods generate iterates $(w^i, \pi^i, \lambda^i, t^i)$, $i = 1, 2, \ldots$, with $(\lambda^i, t^i) > 0$ that approach feasibility with respect to the conditions (23a) as $i \to \infty$. The search directions are Newton-like directions for the equality conditions in (23a). Dropping the superscript and denoting the current iterate by (w, π, λ, t), we can write the general linear system to be solved for the search direction as

$$\begin{bmatrix} Q & F^T & C^T & \\ -F & & & \\ -C & & & -I \\ & & T & \Lambda \end{bmatrix} \begin{bmatrix} \Delta w \\ \Delta \pi \\ \Delta \lambda \\ \Delta t \end{bmatrix} = \begin{bmatrix} r_Q \\ r_F \\ r_C \\ r_t \end{bmatrix}. \tag{24}$$

(Note that the coefficient matrix is the Jacobian of the nonlinear equations (23a).) Different primal-dual methods are obtained from different choices of the right-hand side vector (r_Q, r_F, r_C, r_t). The duality gap μ defined by

$$\mu = \lambda^T t / m \tag{25}$$

is typically used as a measure of optimality of the current point (w, π, λ, t). In principle, primal-dual interior-point methods ensure that the norm of the function $\mathcal{F}$ defined by (23a) remains bounded by a constant multiple of μ at each iterate, thus ensuring that μ is also a measure of infeasibility of the current point. However, the latter condition is rarely checked in practical algorithms.

We use a variant of Mehrotra's predictor-corrector algorithm [15] to solve (22). This algorithm has proved to be the most effective approach for general linear programs and is similarly effective for convex quadratic programming. The first part of the Mehrotra search direction—the *predictor* or *affine-scaling* step—is simply a pure Newton step for the system (23a), obtained by solving (24) with the following right-hand side:

$$\begin{bmatrix} r_Q \\ r_F \\ r_C \\ r_t \end{bmatrix} = -\mathcal{F}(w, \pi, \lambda, t) = - \begin{bmatrix} Qw + F^T\pi + C^T\lambda + c \\ -Fw + g \\ -Cw - t + d \\ T\Lambda e \end{bmatrix}. \tag{26}$$

We denote the corresponding solution of (24) by $(\Delta w_{\text{aff}}, \Delta\pi_{\text{aff}}, \Delta\lambda_{\text{aff}}, \Delta t_{\text{aff}})$. The second part of the search direction—the *centering-corrector* direction $(\Delta w_{\text{cc}}, \Delta\pi_{\text{cc}}, \Delta\lambda_{\text{cc}}, \Delta t_{\text{cc}})$—is calculated by choosing the centering parameter $\sigma \in [0,1)$ as outlined below and solving the system (24) with the following right-hand side:

$$\begin{bmatrix} r_Q \\ r_F \\ r_C \\ r_t \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -\Delta T_{\text{aff}}\Delta\Lambda_{\text{aff}} e + \sigma\mu e \end{bmatrix}, \tag{27}$$

where ΔT_{aff} and $\Delta\Lambda_{\text{aff}}$ are the diagonal matrices constructed from the elements of Δt_{aff} and $\Delta\lambda_{\text{aff}}$, respectively.

The following heuristic for choosing the value of σ has proved to be highly effective. We first compute the maximum step length α_{aff} that can be taken along the affine-scaling direction, as follows:

$$\alpha_{\text{aff}} = \arg\max\{\alpha \in [0,1] \mid (\lambda, t) + \alpha(\Delta\lambda_{\text{aff}}, \Delta t_{\text{aff}}) \geq 0\}.$$

The duality gap μ_{aff} attained from this full step to the boundary is

$$\mu_{\text{aff}} = (\lambda + \alpha\Delta\lambda_{\text{aff}})^T(t + \alpha\Delta t_{\text{aff}})/m.$$

Finally, we set

$$\sigma = \left(\frac{\mu_{\text{aff}}}{\mu}\right)^3.$$

The search direction is obtained by adding the predictor and centering-corrector directions, as follows:

$$(\Delta w, \Delta\pi, \Delta\lambda, \Delta t) = (\Delta w_{\text{aff}}, \Delta\pi_{\text{aff}}, \Delta\lambda_{\text{aff}}, \Delta t_{\text{aff}}) + (\Delta w_{\text{cc}}, \Delta\pi_{\text{cc}}, \Delta\lambda_{\text{cc}}, \Delta t_{\text{cc}}). \tag{28}$$

Note that the coefficient matrix in (24) is the same for both the predictor and centering-corrector systems, so just one factorization of this matrix is required at each iteration. Apart from this factorization, the main computational operations at each iteration include two back-substitutions for two different right-hand sides, and a number of matrix-vector operations.

The distance we move along the direction (28) is defined in terms of the maximum step $\alpha_{\max}$ that can be taken without violating the condition (23b):

$$\alpha_{\max} = \arg\max \{\alpha \in [0,1] \,|\, (\lambda, t) + \alpha(\Delta\lambda, \Delta t) \geq 0\}.$$

The actual steplength α is chosen to be

$$\alpha \leftarrow \gamma \alpha_{\max}, \tag{29}$$

where γ is a parameter in the range $(0,1)$ chosen to ensure that the pairwise products $\lambda_i t_i$ do not become too unbalanced. The value of γ is typically close to 1; it has proved effective in practice to allow it to approach 1 as the algorithms gets closer and closer to the solution. See Mehrotra [15] for the details of a heuristic for choosing γ.

The algorithm does not require the initial point to be feasible, and checks can be added to detect problems for which no feasible points exist. In our case, feasibility of the MPC problem obtained from the Rawlings and Muske formulation with unstable plants can be determined a priori by solving the linear program (15), (16).

Finally, we note that block elimination can be applied to the system (24) to obtain reduced systems with more convenient structures. By eliminating Δt, we obtain the following system:

$$\begin{bmatrix} Q & F^T & C^T \\ -F & & \\ -C & & \Lambda^{-1}T \end{bmatrix} \begin{bmatrix} \Delta w \\ \Delta \pi \\ \Delta \lambda \end{bmatrix} = \begin{bmatrix} r_Q \\ r_F \\ r_C + \Lambda^{-1} r_t \end{bmatrix} \overset{\text{def}}{=} \begin{bmatrix} \hat{r}_Q \\ \hat{r}_F \\ \hat{r}_C \end{bmatrix}. \tag{30}$$

Since $\Lambda^{-1}T$ is a positive diagonal matrix, we can easily eliminate $\Delta\lambda$ as well to obtain

$$\begin{bmatrix} Q + C^T \Lambda T^{-1} C & F^T \\ -F & 0 \end{bmatrix} \begin{bmatrix} \Delta w \\ \Delta \pi \end{bmatrix} = \begin{bmatrix} r_Q - C^T T^{-1}(\Lambda r_C + r_t) \\ r_F \end{bmatrix}. \tag{31}$$

As we see in the next section, these eliminations can be applied to our particular problem to put the system in a form in which we can apply the Riccati block-elimination technique of Sections 3.3 and 3.4.

We conclude with a note on the sizes of elements in t and λ and their effect on elements of the matrices in (30) and (31). In path-following interior-point methods that adhere rigorously to the theory, iterates are confined to a region in which the pairwise products $t_i\lambda_i$ are not too different from each other in size. A bound of the form

$$t_i \lambda_i \geq \gamma \mu \tag{32}$$

is usually enforced, where μ is the average value of $t_i\lambda_i$ (see (25)) and $\gamma \in (0,1)$ is constant, typically $\gamma = 10^{-4}$. When the primal-dual solution set for (22) is bounded, we have further that

$$t_i \leq \beta, \quad \lambda_i \leq \beta, \quad i = 1, 2, \ldots, m, \tag{33}$$

for some constant bound $\beta > 0$. It follows immediately from (32) and (33) that

$$\frac{\gamma}{\beta^2}\mu \leq \frac{t_i}{\lambda_i} \leq \frac{\beta^2}{\gamma}\mu^{-1}, \quad \frac{\gamma}{\beta^2}\mu \leq \frac{\lambda_i}{t_i} \leq \frac{\beta^2}{\gamma}\mu^{-1}. \tag{34}$$

Hence, the diagonal elements of the matrices $T^{-1}\Lambda$ and $\Lambda^{-1}T$ lie in the range $[\Theta(\mu), \Theta(\mu^{-1})]$.

Although bounds of the form (32) are not enforced explicitly in most implementations of Mehrotra's algorithm, computational experience shows that they are almost always satisfied in practice. Hence, it is reasonable to assume, as we do in the analysis of numerical stability below, that the estimates (34) are satisfied by iterates of our algorithm.

3.2 Efficient MPC Formulation

The optimal control problem (19), (20) traditionally has been viewed as a problem in which just the inputs are variables, while the states are eliminated by direct substitution using the transition equation (20b) (see, for example, Muske and Rawlings [16]). We refer to this formulation hereafter as the standard method. Unfortunately, the constraint and Hessian matrices in the reduced problem resulting from this procedure are generally dense, so the computational cost of solving the problem is proportional to N^3. Efficient commercial solvers for dense quadratic programs (such as QPSOL [8]) can then be applied to the reduced problem.

Unless the number of stages N is small, the $O(N^3)$ cost of the standard method is unacceptable because the "unconstrained" version of (19) is known to be solvable in $O(N)$ time by using a Riccati equation or dynamic programming. We are led to ask whether there is an algorithm for the constrained problem (19), (20) that preserves the $O(N)$ behavior. In fact, the interior-point algorithm of the preceding section almost attains this goal, since it can be applied to the problem (19), (20) at a cost of $O(N)$ operations *per iteration*. The rows and columns of the reduced linear systems (30) and (31) can be rearranged to make these matrices *banded*, with dimension proportional to N and and bandwidth independent of N. Since the number of iterations required by the interior-point algorithm depends only weakly on N in practice, the total computational cost of this approach is only slightly higher than $O(N)$. In both the active set and interior-point approaches, the dependence of solution time on other parameters, such as the number of inputs, the number of states, and the number of side constraints, is cubic.

Wright [25, 27] describes a scheme in which these banded matrices are explicitly formed and factored with a general banded factorization routine. In the next section, we show that the linear system to be solved at each interior-point iteration can be reduced to a form identical to the "unconstrained" version of (19), (20), that is, a form in which the side constraints (20c), (20d) are absent. Hence, a Riccati recursion similar to the technique used for the unconstrained problem can be used to solve this linear system. Even though such a scheme places restrictions on the use of pivoting for numerical stability, we show by a simple argument that numerical stability can be expected.

Suppose that the interior-point algorithm of Section 3.1 is applied to the problem (19), (20). We use λ_k, ζ_k, and η_k to denote the Lagrange multipliers for the constraints (20c), (20d), and (20e), respectively. We rearrange the linear system (30) to be solved at each iteration of the interior-point method by "interleaving" the variables and equations according to stage index. That is, the primal and dual variables for stage 0 are listed before those for stage 1, and so on. For this ordering, the rows of the system (30) that correspond to stage k are as follows:

$$\begin{bmatrix} \cdots & Q & M & -G^T & A^T & & & & & \\ & M^T & R & D^T & B^T & & & & & \\ & -G & D & -\Sigma_k^D & & & & & & \\ & A & B & & & & & & -I & \\ & & & & & -\Sigma_{k+1}^{\epsilon} & & -I & & \\ & & & & & & -\Sigma_{k+1}^H & -I & H & \\ & & & & & -I & -I & Z & & \\ & & & & -I & & H^T & & Q & \cdots \end{bmatrix} \begin{bmatrix} \vdots \\ \Delta x_k \\ \Delta u_k \\ \Delta \lambda_k \\ \Delta p_{k+1} \\ \Delta \xi_{k+1} \\ \Delta \eta_{k+1} \\ \Delta \epsilon_{k+1} \\ \Delta x_{k+1} \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ r_k^x \\ r_k^u \\ r_k^{\lambda} \\ r_{k+1}^p \\ r_{k+1}^{\xi} \\ r_{k+1}^{\eta} \\ r_{k+1}^{\epsilon} \\ r_{k+1}^x \\ \vdots \end{bmatrix}. \tag{35}$$

In this system, the diagonal matrices Σ_k^D, Σ_k^{ϵ}, and Σ_k^H, which correspond to $\Lambda^{-1}T$ in the general system (30), are defined by

$$\Sigma_k^D = (\Lambda_k)^{-1}T_k^{\lambda}, \quad \Sigma_k^{\epsilon} = (\Xi_k)^{-1}T_k^{\xi}, \quad \Sigma_k^H = (\mathcal{H}_k)^{-1}T_k^{\eta}, \tag{36}$$

where Λ_k, Ξ_k, and $\mathcal{H}_k$ are the diagonal matrices whose diagonal elements are the Lagrange multipliers λ_k, ξ_k, and η_k, while T_k^{λ}, T_k^{ξ}, and T_k^{η} are likewise diagonal matrices constructed from the slack variables associated with the constraints (20c), (20d), and (20e), respectively. The final rows in this linear system are

$$\begin{bmatrix} \cdots & Q & M & -G^T & A^T & & & & & \\ & M^T & R & D^T & B^T & & & & & \\ & -G & D & -\Sigma_{N-1}^D & & & & & & \\ & A & B & & & & & & -I & \\ & & & & & -\Sigma_N^{\epsilon} & & -I & & \\ & & & & & & -\Sigma_N^H & -I & H & \\ & & & & & -I & -I & Z & & \\ & & & & -I & & H^T & & \bar{Q}_N & F^T \\ & & & & & & & & F & \end{bmatrix} \begin{bmatrix} \vdots \\ \Delta x_{N-1} \\ \Delta u_{N-1} \\ \Delta \lambda_{N-1} \\ \Delta p_N \\ \Delta \xi_N \\ \Delta \eta_N \\ \Delta \epsilon_N \\ \Delta x_N \\ \Delta \beta \end{bmatrix} = \begin{bmatrix} r_{N-1}^x \\ r_{N-1}^u \\ r_{N-1}^{\lambda} \\ r_{N-1}^p \\ r_{N-1}^{\xi} \\ r_{N-1}^{\eta} \\ r_{N-1}^{\epsilon} \\ r_N^x \\ r^{\beta} \end{bmatrix}, \tag{37}$$

where β denotes the Lagrange multiplier for the endpoint constraint (20f).

By eliminating the Lagrange multiplier steps $\Delta\lambda_k$, $\Delta\xi_k$, $\Delta\eta_k$, and $\Delta\epsilon_k$ from the systems (35) and (37), we derive the following analog of the compact system (31):

$$\begin{bmatrix} R_0 & B^T & & & & & & & & \\ B & & -I & & & & & & & \\ & -I & Q_1 & M_1 & A^T & & & & & \\ & & M_1^T & R_1 & B^T & & & & & \\ & & A & B & & -I & & & & \\ & & & & -I & Q_2 & M_2 & A^T & & \\ & & & & & M_2^T & R_2 & B^T & & \\ & & & & & A & B & \ddots & \ddots & \\ & & & & & & & \ddots & Q_N & F^T \\ & & & & & & & & F & \end{bmatrix} \begin{bmatrix} \Delta u_0 \\ \Delta p_0 \\ \Delta x_1 \\ \Delta u_1 \\ \Delta p_1 \\ \Delta x_2 \\ \Delta u_2 \\ \vdots \\ \Delta x_N \\ \Delta\beta \end{bmatrix} = \begin{bmatrix} \tilde{r}_0^u \\ \tilde{r}_0^p \\ \tilde{r}_1^x \\ \tilde{r}_1^u \\ \tilde{r}_1^p \\ \tilde{r}_2^x \\ \tilde{r}_2^u \\ \vdots \\ \tilde{r}_N^x \\ r^\beta \end{bmatrix}, \tag{38}$$

where

$$\begin{array}{rcll} R_k & = & R + D^T(\Sigma_k^D)^{-1}D, & k = 0,\ldots,N-1, \\ M_k & = & M - G^T(\Sigma_k^D)^{-1}D, & k = 1,\ldots,N-1, \\ Z_k & = & Z + (\Sigma_k^\epsilon)^{-1} + (\Sigma_k^H)^{-1}, & k = 1,\ldots,N, \\ Q_k & = & Q + G^T(\Sigma_k^D)^{-1}G + H^T[(\Sigma_k^H)^{-1} - (\Sigma_k^H Z_k \Sigma_k^H)^{-1}]H, & k = 1,\ldots,N-1, \\ Q_N & = & \bar{Q}_N + H^T[(\Sigma_N^H)^{-1} - (\Sigma_N^H Z_N \Sigma_N^H)^{-1}]H, & \end{array} \tag{39}$$

and

$$\begin{array}{rcll} \tilde{r}_k^u & = & r_k^u + D^T(\Sigma^D)^{-1}r_k^\lambda, & k = 0,\ldots,N-1, \\ \tilde{r}_k^p & = & r_k^p, & k = 0,\ldots,N-1, \\ \tilde{r}_k^\epsilon & = & r^\epsilon - (\Sigma_k^\epsilon)^{-1}r_k^\xi - (\Sigma_k^H)^{-1}r_k^\eta, & k = 1,\ldots,N, \\ \tilde{r}_k^x & = & r_k^x + -G^T(\Sigma_k^D)^{-1}r_k^\lambda + H^T(\Sigma_k^H)^{-1}r_k^\eta + H^T(\Sigma_k^H Z_k)^{-1}\tilde{r}_k^\epsilon, & k = 1,\ldots,N-1, \\ \tilde{r}_N^x & = & r_N^x + H^T(\Sigma_N^H)^{-1}r_N^\eta + H^T(\Sigma_N^H Z_N)^{-1}\tilde{r}_N^\epsilon. & \end{array} \tag{40}$$

This matrix has the same form as the KKT matrix obtained from the following problem in which the only constraint (apart from the model equation and initial state) is a final point condition:

$$\min_{u,x} \; \Phi(u,x) = \frac{1}{2}u_0^T R_0 u_0 + \sum_{k=1}^{N-1} \frac{1}{2}(x_k^T Q_k x_k + u_k^T R_k u_k + 2x_k^T M_k u_k) + \frac{1}{2}x_N^T Q_N x_N, \tag{41}$$

subject to

$$x_0 = \hat{x}_j, \qquad \text{(fixed)}, \tag{42a}$$

$$x_{k+1} = Ax_k + Bu_k, \qquad k = 0,1,\ldots,N-1, \tag{42b}$$

$$Fx_N = 0. \tag{42c}$$

The problem (41), (42) is convex if the matrices R_0, Q_N, and

$$\begin{bmatrix} Q_k & M_k \\ M_k^T & R_k \end{bmatrix}, \qquad k = 1, 2, \ldots, N-1, \tag{43}$$

are all SPSD. The following brief discussion shows that this property holds. First, we show that

$$(\Sigma_k^H)^{-1} - (\Sigma_k^H Z_k \Sigma_k^H)^{-1} \quad \text{is positive diagonal for all } k = 1, 2, \ldots, N. \tag{44}$$

By using the definition of Z_k above, together with the diagonality of Z and Σ_k^ϵ, we have that

$$\begin{aligned} &(\Sigma_k^H)^{-1} - (\Sigma_k^H Z_k \Sigma_k^H)^{-1} \\ &\quad = (\Sigma_k^H)^{-1}[I - (\Sigma_k^H Z_k)^{-1}] = (\Sigma_k^H)^{-1}[I - \left(\Sigma_k^H Z + \Sigma_k^H (\Sigma_k^\epsilon)^{-1} + I\right)^{-1}]. \end{aligned}$$

Since Z, Σ_k^H, and Σ_k^ϵ are all positive diagonal matrices, the final expression above is a product of two positive diagonal matrices, and therefore is itself positive diagonal. Hence, property (44) holds. Note from (39) that Q_N is an SPSD modification of an SPSD matrix, and therefore is itself SPSD. Note too that from (39) again, we have

$$\begin{aligned} &\begin{bmatrix} Q_k & M_k \\ M_k^T & R_k \end{bmatrix} \\ &= \begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} + \begin{bmatrix} D^T \\ -G^T \end{bmatrix} (\Sigma_k^D)^{-1} \begin{bmatrix} D & -G \end{bmatrix} + \begin{bmatrix} H^T[(\Sigma_k^H)^{-1} - (\Sigma_k^H Z_k \Sigma_k^H)^{-1}]H & 0 \\ 0 & 0 \end{bmatrix}, \end{aligned}$$

for all $k = 1, 2, \ldots, N-1$. Because of (44), we have that the left-hand side of this expression is a sum of SPSD terms, and therefore is itself SPSD. Finally, note from (39) that each R_k, $k = 0, 1, \ldots, N-1$ is the sum of a PSD matrix R and an SPSD term $D^T(\Sigma_k^D)^{-1}D$, and is therefore itself SPD. We conclude that the objective function (41) is convex.

If we use n to denote the number of components of each state vector x_k and m to denote the number of components of each input vector u_k, we find that the banded coefficient matrix in (38) has dimension approximately $N(2n+m)$ and half-bandwidth approximately $2n+m$, so that the computational cost of factoring it by Gaussian elimination would be proportional to $N(m+n)^3$. This estimate is linear in N, unlike the naive dense implementation for which the cost grows like $N^3(m+n)^3$.

3.3 Block Elimination: No Endpoint Constraints

We can improve the efficiency of the algorithm by applying a block factorization scheme to (38) in place of the elimination scheme for general banded matrices. In this section, we consider the case in which endpoint constraints are not present in the problem (so that the quantities F, $\Delta\beta$, and r^β do not appear in (38)). We describe a block elimination scheme and show that it yields a Riccati recursion.

For simplicity, we rewrite the system (38) for the case of no endpoint constraints as follows:

$$\begin{bmatrix} R_0 & B^T & & & & & & & & \\ B & & -I & & & & & & & \\ & -I & Q_1 & M_1 & A^T & & & & & \\ & & M_1^T & R_1 & B^T & & & & & \\ & & A & B & & -I & & & & \\ & & & & -I & Q_2 & M_2 & A^T & & \\ & & & & & M_2^T & R_2 & B^T & & \\ & & & & & A & B & \ddots & \ddots & \\ & & & & & & & & \ddots & \bar{Q}_N \end{bmatrix} \begin{bmatrix} \widehat{\Delta u}_0 \\ \widehat{\Delta p}_0 \\ \widehat{\Delta x}_1 \\ \widehat{\Delta u}_1 \\ \widehat{\Delta p}_1 \\ \widehat{\Delta x}_2 \\ \widehat{\Delta u}_2 \\ \vdots \\ \widehat{\Delta x}_N \end{bmatrix} = \begin{bmatrix} \tilde{r}_0^u \\ \tilde{r}_0^p \\ \tilde{r}_1^x \\ \tilde{r}_1^u \\ \tilde{r}_1^p \\ \tilde{r}_2^x \\ \tilde{r}_2^u \\ \vdots \\ \tilde{r}_N^x \end{bmatrix}. \tag{45}$$

Our scheme yields a set of matrices $\Pi_k \in \mathbb{R}^{n\times n}$ and vectors $\pi_k \in \mathbb{R}^n$, $k = N, N-1, \ldots, 1$, such that the following relationship holds between the unknown vectors $\widehat{\Delta p}_{k-1}$ and $\widehat{\Delta x}_k$ in (45):

$$-\widehat{\Delta p}_{k-1} + \Pi_k \widehat{\Delta x}_k = \pi_k, \qquad k = N, N-1, \ldots, 1. \tag{46}$$

We can see immediately from (45) that (46) is satisfied for $k = N$ if we define

$$\Pi_N = \bar{Q}_N, \qquad \pi_N = \tilde{r}_N^x. \tag{47}$$

The remaining quantities Π_k and π_k can be generated recursively. If (46) holds for some k, we can combine this equation with three successive block rows from (45) to obtain the following subsystem:

$$\begin{bmatrix} -I & Q_{k-1} & M_{k-1} & A^T & \\ & M_{k-1}^T & R_{k-1} & B^T & \\ & A & B & 0 & -I \\ & & & -I & \Pi_k \end{bmatrix} \begin{bmatrix} \widehat{\Delta p}_{k-2} \\ \widehat{\Delta x}_{k-1} \\ \widehat{\Delta u}_{k-1} \\ \widehat{\Delta p}_{k-1} \\ \widehat{\Delta x}_k \end{bmatrix} = \begin{bmatrix} \tilde{r}_{k-1}^x \\ \tilde{r}_{k-1}^u \\ \tilde{r}_{k-1}^p \\ \pi_k \end{bmatrix}. \tag{48}$$

Elimination of $\widehat{\Delta p}_{k-1}$ and $\widehat{\Delta x}_k$ yields

$$\begin{bmatrix} -I & Q_{k-1} + A^T\Pi_k A & A^T\Pi_k B + M_{k-1} \\ 0 & B^T\Pi_k A + M_{k-1}^T & R_{k-1} + B^T\Pi_k B \end{bmatrix} \begin{bmatrix} \widehat{\Delta p}_{k-2} \\ \widehat{\Delta x}_{k-1} \\ \widehat{\Delta u}_{k-1} \end{bmatrix} = \begin{bmatrix} \tilde{r}_{k-1}^x + A^T\Pi_k\tilde{r}_{k-1}^p + A^T\pi_k \\ \tilde{r}_{k-1}^u + B^T\Pi_k\tilde{r}_{k-1}^p + B^T\pi_k \end{bmatrix}. \tag{49}$$

Finally, elimination of $\widehat{\Delta u}_{k-1}$ yields the equation

$$-\widehat{\Delta p}_{k-2} + \Pi_{k-1}\widehat{\Delta x}_{k-1} = \pi_{k-1}. \tag{50}$$

where

$$\begin{aligned}
\Pi_{k-1} &= Q_{k-1} + A^T\Pi_k A - \qquad\qquad (51a)\\
&\qquad (A^T\Pi_k B + M_{k-1})(R_{k-1} + B^T\Pi_k B)^{-1}(B^T\Pi_k A + M_{k-1}^T),\\
\pi_{k-1} &= \tilde{r}_{k-1}^x + A^T\Pi_k\tilde{r}_{k-1}^p + A^T\pi_k - \qquad\qquad (51b)\\
&\qquad (A^T\Pi_k B + M_{k-1})(R_{k-1} + B^T\Pi_k B)^{-1}(\tilde{r}_{k-1}^u + B^T\Pi_k\tilde{r}_{k-1}^p + B^T\pi_k).
\end{aligned}$$

The equation (51a) is the famous discrete-time Riccati equation for time-varying weighting matrices.

The solution of (45) can now be obtained as follows. We first set Π_N and π_n using (47), and then apply (51a) to obtain Π_k and π_k for for $k = N-1, N-2, \ldots, 1$. Next, we combine (46) for $k = 1$ with the first two rows of (45), we obtain

$$\begin{bmatrix} R_0 & B^T & \\ B & & -I \\ & -I & \Pi_1 \end{bmatrix} \begin{bmatrix} \widehat{\Delta u}_0 \\ \widehat{\Delta p}_0 \\ \widehat{\Delta x}_1 \end{bmatrix} = \begin{bmatrix} \tilde{r}_0^u \\ \tilde{r}_0^p \\ \pi_1 \end{bmatrix}, \qquad (52)$$

and solve this system for $\widehat{\Delta u}_0$, $\widehat{\Delta x}_1$, and $\widehat{\Delta p}_0$. Next, we obtain from (49) and (48) that

$$\begin{aligned}
\widehat{\Delta u}_k &= (R_k + B^T\Pi_{k+1}B)^{-1}[\tilde{r}_k^u + B^T\Pi_{k+1}\tilde{r}_k^p + B^T\pi_{k+1} - (B^T\Pi_{k+1}A + M_k^T)\widehat{\Delta x}_k],\\
\widehat{\Delta x}_{k+1} &= A\widehat{\Delta x}_k + B\widehat{\Delta u}_k, \qquad k = 1, 2, \ldots, N-1.
\end{aligned}$$

Finally, the steps $\widehat{\Delta p}_k$ for $k = N-1, N-2, \ldots, 1$ can be recovered from (46) The computational cost of the entire process is $O(N(m+n)^3)$.

The question of stability of this approach is an important one. The block elimination/Riccati scheme just described essentially places restrictions on the pivot sequence, that is, the order in which the elements of the matrix in (45) are eliminated. (Note however that pivoting for numerical stability can occur "internally," during the factorization of $(R_{k-1} + B^T\Pi_k B)$ in (51a) and (51b) for $k = N, N-1, \ldots, 2$.) In other circumstances, pivot restrictions are well known to lead to numerical instability, which manifests itself by blowup of the intermediate quantities that arise during the factorization (by which we mean that the intermediate quantities become much larger than the original data of the problem.) However, in the present case, stability can be established by the simple argument of next few paragraphs.

The coefficient matrix in (45) becomes increasingly ill-conditioned near the solution. This feature results from wide variation among the elements of the diagonal matrices Σ_k^D, Σ_k^ϵ, and Σ_k^H defined by (36) which, as we see from (34), can vary between $\Theta(\mu)$ and $\Theta(\mu^{-1})$, where the duality measure μ approaches zero as the iterates approach the solution. It follows from (39) that Q_k, $k = 1, 2, \ldots, N$ has its eigenvalues in the range $[0, \Theta(\mu^{-1})]$, while positive definiteness of R ensures that the eigenvalues of R_k, $k = 0, 1, \ldots, N-1$ lie in an interval $[\Theta(1), \Theta(\mu^{-1})]$. Since we showed earlier that the matrices

$$\begin{bmatrix} Q_k & M_k \\ M_k^T & R_k \end{bmatrix}, \qquad k = 1, 2, \ldots, N-1,$$

are SPSD, we deduce from the comments just made that their eigenvalues too must lie in the range $[0, \Theta(\mu^{-1})]$.

We now show that blowup does not occur during computation of the Riccati matrices Π_k and that, in all cases, their eigenvalues lie in the range $[0, \Theta(\mu^{-1})]$. This is certainly true of the starting matrix Π_N defined by (47). For the remaining matrices defined by (51a), we assume that our assertion is true for Π_k for some k, and prove that it continues to hold for Π_{k-1}. Note that the matrix

$$\left[\begin{array}{cc} Q_{k-1} & M_{k-1} \\ M_{k-1}^T & R_{k-1} \end{array} \right] + \left[\begin{array}{c} A^T \\ B^T \end{array} \right] \Pi_k \left[\begin{array}{cc} A & B \end{array} \right], \tag{53}$$

has both terms SPSD, with eigenvalues in the range $[0, \Theta(\mu^{-1})]$. Since Π_{k-1} is the Schur complement of $(R_{k-1} + B^T \Pi_k B)$ in the matrix (53), it must be positive semidefinite. (Note that Π_{k-1} is well defined by the formula (51a), since $R_{k-1} + B^T \Pi_k B$ is an SPSD modification of the SPD matrix R, and so its inverse is well defined.) Moreover, we can see from (51a) that Π_k is obtained by subtracting an SPSD matrix from he SPSD matrix $Q_{k-1} + A^T \Pi_k A$, and so its eigenvalues are bounded above by the eigenvalues of the latter matrix. By combining these observations, we conclude that the eigenvalues of Π_{k-1} lie in the range $[0, \Theta(\mu^{-1})]$, as claimed.

For the vectors π_k, $k = N, N-1, \ldots, 1$, we have from the invertibility of $R_{k-1} + B^T \Pi_k B$ that they are well defined. Moreover, since the smallest eigenvalue of $R_{k-1} + B^T \Pi_k B$ has size $\Theta(1)$, we have from the formula (51b) and the estimate $\|\Pi_k\| = O(\mu^{-1})$ from the previous paragraph that $\|\pi_k\| = O(\mu^{-2})$, and so this vector does not blow up with k either. (In fact, a more refined analysis can be used to deduce that $\|\pi_k\| = O(\mu^{-1})$, but we omit the details of this argument here.)

We conclude that numerical instability is not a problem in applying the block elimination/Riccati scheme and that, in fact, we can expect this scheme to be as stable as any general scheme based on Gaussian elimination with pivoting.

It might be expected that the inherent ill conditioning of the system (45) may lead to an inaccurate computed solution, even when our numerical scheme is stable. It has long been observed by interior-point practitioners, however, that the computed steps are surprisingly effective steps for the algorithm, even on later iterations on which μ is tiny. This observation has recently found some theoretical support (see Wright [26, 29]) but the issues involved are beyond the scope of this paper.

3.4 Block Elimination: Endpoint Constraints

When endpoint constraints are present in the problem, they can be accounted for by adding extra recursions to the scheme of the previous section. We describe this approach below, but first mention an alternative way to handle the problem. The presence of endpoint constraints in the model is often symptomatic of the transition matrix A having eigenvalues outside the unit circle. In these circumstances, it is known that Riccati-based techniques can encounter stability difficulties. These difficulties are ameliorated by the technique of parameterizing the input as $u_k = Lx_k + r_k$, where L is a linear stabilizing feedback gain for (A, B), as mentioned in Section 2.1. Alternatively, we can simply discard the Riccati strategy and instead apply a standard banded Gaussian-elimination scheme with partial pivoting to the system (38). Though this approach does not exploit the structure of the problem quite as well as the Riccati strategy, its stability is guaranteed. It can be used as a backup approach if stability problems are encountered with the modified Riccati approach that we now describe.

In the language of linear algebra, our modification of the block-elimination approach proceeds by partitioning the coefficient matrix in (38) as

$$\begin{bmatrix} T_{11} & T_{12} \\ T_{12}^T & T_{22} \end{bmatrix},$$

where

$$T_{11} = \begin{bmatrix} R_0 & B^T & & & & & \\ B & & -I & & & & \\ & -I & Q_1 & M_1 & A^T & & \\ & & M_1^T & R_1 & B^T & & \\ & & A & B & \ddots & \ddots & \\ & & & & \ddots & \bar{Q}_N & \end{bmatrix}, \quad T_{12} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ F^T \end{bmatrix}, \quad T_{22} = 0. \tag{54}$$

We partition the right-hand side and solution of (38) correspondingly and rewrite the system as

$$\begin{bmatrix} T_{11} & T_{12} \\ T_{12}^T & T_{22} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \end{bmatrix},$$

where $r_2 = r^\beta$ and $y_2 = \Delta\beta$. By simple manipulation, assuming that T_{11} is nonsingular, we obtain

$$\begin{aligned} [T_{22} - T_{12}^T T_{11}^{-1} T_{12}] y_2 &= r_2 - T_{12}^T T_{11}^{-1} r_1, &\text{(55a)} \\ y_1 &= T_{11}^{-1} r_1 - T_{11}^{-1} T_{12} y_2. &\text{(55b)} \end{aligned}$$

We calculate the vector $T_{11}^{-1} r_1$ by using the approach of Section 3.3. The other major operation is to find $T_{11}^{-1} T_{12}$, which we achieve by solving the following system:

$$\begin{bmatrix} R_0 & B^T & & & & & & & & \\ B & & -I & & & & & & & \\ & -I & Q_1 & M_1 & A^T & & & & & \\ & & M_1^T & R_1 & B^T & & & & & \\ & & A & B & & -I & & & & \\ & & & & -I & Q_2 & M_2 & A^T & & \\ & & & & & M_2^T & R_2 & B^T & & \\ & & & & & A & B & \ddots & \ddots & \\ & & & & & & & \ddots & \bar{Q}_N & \end{bmatrix} \begin{bmatrix} \Phi_0^u \\ \Phi_0^p \\ \Phi_1^x \\ \Phi_1^u \\ \Phi_1^p \\ \Phi_2^x \\ \vdots \\ \Phi_N^x \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ F^T \end{bmatrix}. \tag{56}$$

The structure of this system is identical to (45) except that the right-hand side is now a matrix instead of a vector. As in the preceding section, we seek $n \times n_f$ matrices Ψ_k, $k = N, N-1, \ldots, 1$ (where n_f is the number of rows in F) such that the following relationship holds between Φ^p_{k-1} and Φ^x_k satisfying (56):

$$-\Phi^p_{k-1} + \Pi_k \Phi^x_k = \Psi_k, \qquad k = N, N-1, \ldots, 1. \tag{57}$$

(Note that Π_k in (57) are identical to the matrices generated by the formulae (47), (51a) of the previous section. This is hardly surprising, since these matrices depend only on the coefficient matrix and not on the right-hand side.) An argument like that of the previous section yields the following recursion for Ψ_k:

$$\begin{aligned} \Psi_N &= F^T, \\ \Psi_{k-1} &= A^T \Psi_k - (A^T \Pi_k B + M_{k-1})(R_{k-1} + B^T \Pi_k B)^{-1} B^T \Psi_k, \qquad k = N, N-1, \ldots, 2. \end{aligned}$$

We solve (56) by using a similar technique to the one used for (45).

We now recover the solution of (38) via (55). By substituting from (54) and (56), we find that

$$\begin{aligned} T_{22} - T_{12}^T T_{11}^{-1} T_{12} &= -(\Phi^x_N)^T F^T, \\ r_2 - T_{12}^T T_{11}^{-1} r_1 &= r^\beta - F \widehat{\Delta x}_N, \end{aligned}$$

so that $y_2 = \Delta\beta$ can be found directly by substituting into (55a). We recover the remainder of the solution vector from (55b) by noting that

$$T_{11}^{-1} r_1 - T_{11}^{-1} T_{12} y_2 = \begin{bmatrix} \widehat{\Delta u}_0 \\ \widehat{\Delta p}_0 \\ \widehat{\Delta x}_1 \\ \widehat{\Delta u}_1 \\ \widehat{\Delta p}_1 \\ \widehat{\Delta x}_2 \\ \vdots \\ \widehat{\Delta x}_N \end{bmatrix} - \begin{bmatrix} \Phi^u_0 \\ \Phi^p_0 \\ \Phi^x_1 \\ \Phi^u_1 \\ \Phi^p_1 \\ \Phi^x_2 \\ \vdots \\ \Phi^x_N \end{bmatrix} \Delta\beta.$$

In the implementation, the recurrences for computing Π_k, Ψ_k, and π_k take place simultaneously, as do the recurrences needed for solving the systems (45) and (56). The additional cost associated with the n_f endpoint constraints is $O(N(m+n)^2 n_f)$. When $n_f < n$—which is a necessary condition for (38) to have a unique solution—the cost of solving the full system (38) is less than double the cost of solving the subsystem (45) alone by the method of the preceding section.

3.5 Hot Starting

Model predictive control solves a sequence of similar optimal control problems in succession. If the model is accurate and disturbances are modest, the solution of one optimal control problem can be shifted one time step forward to yield a good approximation to the solution of the next problem in the sequence. Unfortunately, an approximate solution of this type is not a suitable starting guess for the interior-point method, since it usually lies at the boundary of the feasible region, whereas interior-point methods prefer starting point that *strictly* satisfy the inequalities in the constraint set. Starting points close to the so-called central path are more suitable. In the notation of Section 3.1, the characteristics of such points are that their pairwise products $\lambda_i t_i$ are similar in value for $i = 1, 2, \ldots, m$ and that the ratio of the KKT violations in (23a)—measured by $\mathcal{F}(z, \pi, \lambda, t)$—to the duality gap μ is not too large. We can attempt to find near-central points by bumping components of the "shifted" starting point off their bound. (In the notation of Section 3.1, we turn the zero value of either t_i or λ_i into a small positive value.) A second technique is to use a shifted version of one of the earlier interior-point iterates from the previous problem. Since the interior-point algorithm tends to follow the central path, and since the central path is sensitive to data perturbations only near the solution, this strategy generally produces an iterate that is close to the central path for the new optimal control subproblem.

In the presence of new disturbances, the previous solution has little relevance to the new optimal control problem. A starting point can be constructed from the unconstrained solution, or we can perform a cold start from a well-centered point, as is done to good effect in linear programming codes (see Wright [28, Chapter 10]).

4 Computational Results

To gauge the effectiveness of the structured interior-point approach, we tested it against the "standard" quadratic programming approach, in which the states x_k are eliminated from the problem (19), (20) by using the model equation (20b). A reduced problem with unknowns u_k, $k = 0, 1, \ldots, N-1$ and ϵ_k, $k = 1, 2, \ldots, n$ is obtained. The reduction in dimension is accompanied by filling in of the constraint matrices and the Hessian of the objective. The resulting problem is solved with the widely used code QPSOL [8], which implements an active set method using dense linear algebra calculations.

We compared these two approaches on three common applications of the model predictive control methodology.

Example 1: Copolymerization Reactor. Congalidis et al. [5] presented the following normalized model for the copolymerization of methyl methacrylate (MMA) and vinyl acetate

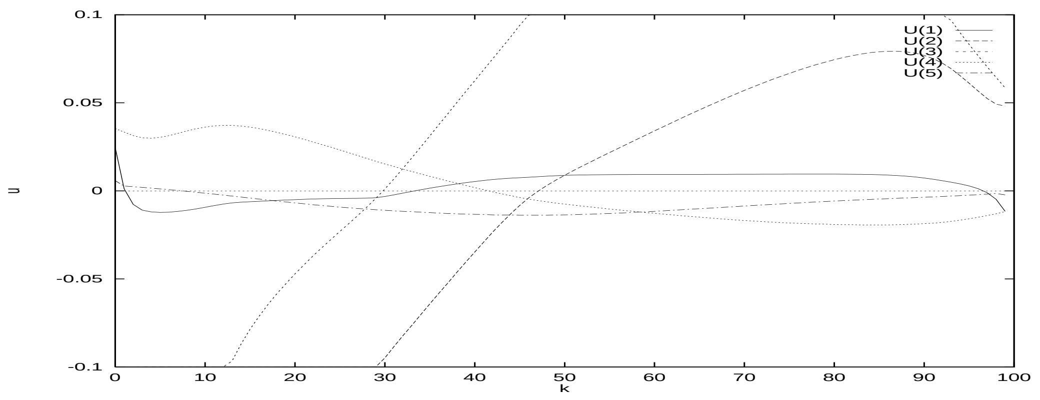

Figure 1: Input Profile for Example 1

(VA) in a continuous stirred tank reactor:

$$G(s) = \begin{bmatrix} \frac{0.34}{0.85s+1} & \frac{0.21}{0.42s+1} & \frac{0.50(0.50s+1)}{12s^2+0.4s+1} & 0 & \frac{6.46(0.9s+1)}{0.07s^2 0.3s+1} \\ \frac{-0.41}{2.41s+1} & \frac{0.66}{1.51s+1} & \frac{-0.3}{1.45s+1} & 0 & \frac{-3.72}{0.8s+1} \\ \frac{0.30}{2.54s+1} & \frac{0.49}{1.54s+1} & \frac{-0.71}{1.35s+1} & \frac{-0.20}{2.71s+1} & \frac{-4.71}{0.008s^2+0.41s+1} \\ 0 & 0 & 0 & 0 & \frac{1.02}{0.07s^2+0.31s+1} \end{bmatrix}.$$

The normalized inputs into the system are the flows of monomer MMA (u_1), monomer VA (u_2), initiator (u_3), and transfer agent (u_4), and the temperature of the reactor jacket (u_5). The normalized outputs of the systems are the polymer production rate (y_1), mole fraction of MMA in the polymer (y_2), average molecular weight of the polymer (y_3), and reactor temperature (y_4). The model was realized in block observer canonical form [3] where the dimension n of state after the realization is 18, and the number m of inputs is 5. The model was discretized with a sample period of 1.

The normalized inputs were constrained to be within 10% of their nominal operating steady–state values The tuning parameters were chosen to be $Q = C^T C$ (where C is the measurement matrix obtained from the state space realization), while $M = 0$, $R = (0.1)I$, and the number of stages N is 100. Due to the very slow dynamics of the reactor, $\bar{Q}$ was obtained from the solution of (8). The parameters z and Z are vacuous, since there are no soft constraints on the state. The controller was simulated with the following state disturbance:

$$[x_0]_j = 0.02 * \sin j.$$

The interior-point method required 14 iterations to solve the optimization problem. Figure 1 shows the optimal control profile normalized with the upper bounds on the input constraints.

Example 2: Gage Control of a Polymer Film Process. We considered the gage (cross-directional control) of a 26-lane polymer film process with 26 actuators. We used the

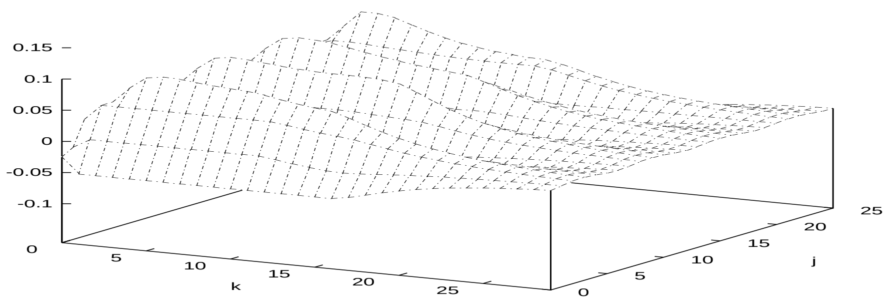

Figure 2: Input Profile for Example 2

following model for our simulation:

$$A = 0.9I, \qquad B = (I - A) * K,$$

where the steady-state gain matrix K was extrapolated from data obtained from a 3M polymer film pilot plant. For this example, the dimension of the state n is 26, and the number m of inputs is 26. The state $[x]_j$ denotes the deviated film thickness in the jth lane, and the input $[u]_j$ denotes the deviated position of the jth actuator.

The actuators were constrained between the values of 0.1 and -0.1, while the velocity of the actuators was constrained between the values of 0.025 and -0.025. Since a large difference between actuator positions can create excess stress on the die, we imposed the following restriction on the change in input from stage to stage:

$$|[u]_j - [u]_{j-1}| < 0.05, \qquad j = 2, 3, \ldots, m.$$

We chose the tuning parameters to be

$$Q = I, \quad R = I, \quad S = I.$$

The matrix $\bar{Q}$ was obtained from the solution of (10). The parameters z and Z are vacuous, since there are no soft constraints on the state. We chose a horizon of $N = 30$ to guarantee that the constraints were satisfied on the infinite horizon. The interior-point method required 11 iterations. Figure 2 shows the calculated optimal input profiles.

Example 3: Evaporator. Ricker et al. [19] presented the following model for an evaporation process in a kraft pulp mill:

$$G(s) = \begin{bmatrix} \frac{1}{30s+1} & 0 \\ \frac{648s}{(30s+1)(20s+1)} & \frac{2.7(-6s+1)}{(20s+1)(5s+1)} \\ \frac{-90s}{(30s+1)(30s+1)} & \frac{-0.1375(-4s+1)}{(30s+1)(2.6s+1)} \end{bmatrix}.$$

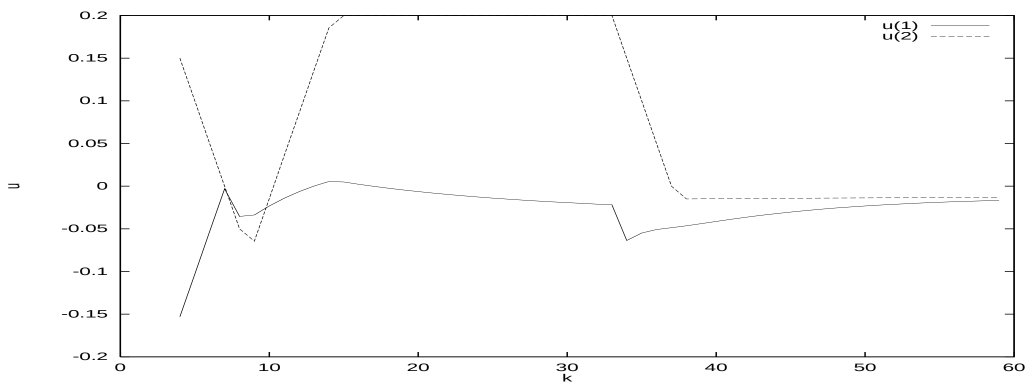

Figure 3: Input Profile at $t = 0$ for Example 3

The normalized outputs of the process are the feed level (y_1), product concentration (y_2), and product level (y_3). The normalized inputs for the the process are the feed level setpoint (u_1) and the steam flow (u_2). The process was realized in block observer canonical form [3] and sampled every 0.5 minutes. The dimension n of the state after the realization is 9, and the number m of input is 3.

Both inputs were constrained to lie in the range $[-0.2, 0.2]$, while the three outputs were constrained to lie in $[-0.05, 0.05]$. A bound of 0.05 was also imposed on the input velocity. The controller was tuned with

$$Q = I, \quad R = I, \quad Z = 0, \quad N = 60.$$

The matrix $\bar{Q}$ was obtained from the solution of (10). A constant ℓ_1 penalty of 1000 was sufficient to force the soft constraints to hold when the solution is feasible. We simulated the controller with the following state disturbance:

$$[x_0]_j = \sin(j) + \cos(j).$$

The interior-point method required 18 iterations to solve the optimization problem. Figure 3 shows the calculated optimal input profile, while Figure 4 shows the predicted output profile. Note that the constraints for y_2 and y_3 are initially violated. The constraint for y_2 is feasible when $k \geq 8$ and the constraint for y_3 is feasible when $k \geq 34$. Increasing the ℓ_1 penalty did not change the resulting solution. Decreasing the ℓ_1 penalty leads to less aggressive control action, but the constraints are violated for a longer duration.

The computational times required by the structured interior-point approach and the naive quadratic programming approach are shown in Table 1. Our platform was a DEC Alphastation 250, and the times were obtained with the Unix `time` command. We used the value $\gamma = 0.995$ in (29) as the proportion of maximum step to the boundary taken by our algorithm.

For the chosen (large) values of the horizon parameter N, the structured interior-point method easily outperforms the naive quadratic programming approach. For the latter approach, we do not include the time required to eliminate the states. These times were often

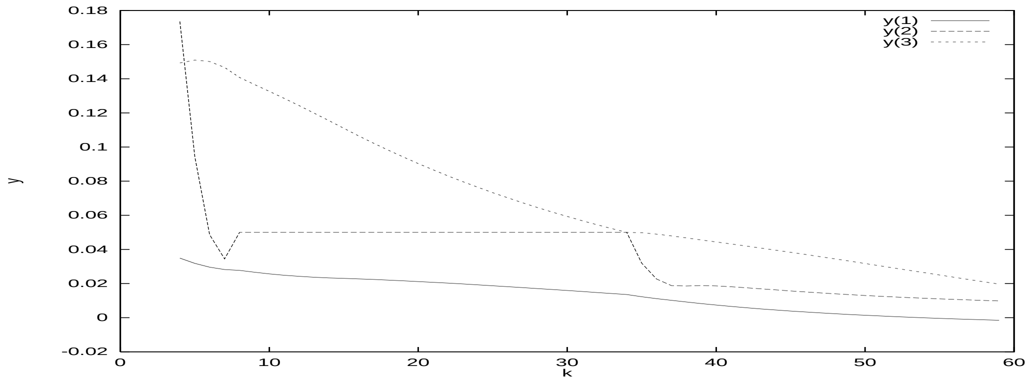

Figure 4: Predicted Output Profile for Example 3

Table 1: Computational Times (sec)

Example	Structured Interior-Point	Naive Quadratic Programming
1	3.80	23.78
2	20.33	276.91
3	2.01	25.32

quite significant, but they are calculated offline. For small values of the horizon parameter N, the naive quadratic programming approach outperforms the structured interior-point method, since the bandwidth is roughly the same relative order of magnitude as the dimensions of (38).

5 Concluding Remarks

We conclude with four brief comments on the structured interior-point method for MPC. The first is that the structured method presented is also directly applicable to the dual problem of MPC, the constrained moving horizon estimation problem. In fact, the estimation problem will provide greater justification for structured approach because long horizons N arise frequently in this context. However, we did not investigate applying the structured optimization approach because the theory for linear constrained receding horizon estimators is still in its infancy.

The second comment is that we can extend the structured method to nonlinear MPC by applying the approach of this paper to the linear-quadratic subproblems generated by sequential quadratic programming. Wright [25], Arnold et al. [1], and Steinbach [23] all apply a similar technique to discrete-time optimal-control problems. While some theory for nonlinear MPC is available, the questions of robust implementation and suitable formulation of nonlinear MPC have not been resolved. See Mayne [13] for a discussion of the some of the issues.

Third, since the computational cost of the proposed algorithm is $O(N(m+n)^3)$, systems with large numbers of states and inputs can still present formidable computational challenges. Since large systems tend to be sparse (that is, A and B tend to be sparse, while Q and R tend to be nearly diagonal), we expect substantial increases in computational performance by exploiting the sparsity in (38) through the use of sparse matrix solvers. Since the sparsity tends to be structured in many applications, different strategies are preferable for different classes of processes. See, for example, the paper of Rao et al. [17], who investigated strategies for further decomposing the problem structure in the gage control of sheet and film forming processes.

The fourth comment concerns time delays, which occur when more than one sampling period elapses before an input u_k affects the state of the system. In the simplest case, we can rewrite the state equation (5a) as

$$x_{k+1} = Ax_k + Bu_{k-d}, \tag{58}$$

for the case in which the delay is d sampling periods. The natural infinite horizon LQR objective function for this case is

$$\Phi = \frac{1}{2}\sum_{k=0}^{\infty}(x_k^T Q x_k + u_k^T R u_k + 2x_{k+d}^T M u_k), \tag{59}$$

where the cross-penalty term relates u_{k-d} and x_k. Since the first $(d+1)$ state vectors $x_0, x_1, \ldots, x_d$ are independent of the inputs, the decision variables in the optimization problem are $x_{d+1}, x_{d+2}, \ldots$ and $u_0, u_1, \ldots$. By defining

$$\tilde{x}_k = x_{k+d}, \qquad k = 0, 1, 2, \ldots,$$

and removing constant terms from (59), the objective function and state equation become

$$\Phi' = \tfrac{1}{2}\textstyle\sum_{k=0}^{\infty}(\tilde{x}_k^T Q \tilde{x}_k + u_k^T R u_k + 2\tilde{x}_k^T M u_k), \tag{60}$$

$$\tilde{x}_0 = x_d, \quad \tilde{x}_{k+1} = A\tilde{x}_k + Bu_k, \quad k = 0, 1, 2, \ldots. \tag{61}$$

These formulae have the same form as (4) and (5).

If no additional constraints of the form (5b) are present, a Riccati equation may be used to solve (60), (61) directly, as in Section 2.2. If state constraints of the form $Hx_k \le h$ or jump constraints of the form $G\Delta u_k \le g$ are present (as in (1)), we can still apply constraint softening (Section 2.3) and use the approaches described in Section 2.2 To obtain finite-horizon versions of (60), (61). The techniques of Section 3.1 can then be used to solve the problem efficiently.

Difficulties may arise, however, when multiple time delays are present, since these may reduce the locality of the relationships between the decision variables and lead to significant broadening of the bandwidths of the matrices in (35) and (38). A process in which two time delays are present (of d_1 and d_2 sampling intervals) can be described by a state equation of the following form:

$$\begin{bmatrix} x^1_{k+1} \\ x^2_{k+1} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x^1_k \\ x^2_k \end{bmatrix} + \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \begin{bmatrix} u^1_{k-d_1} \\ u^2_{k-d_2} \end{bmatrix}.$$

A problem with these dynamics can be solved by augmenting the state vector x_k with the input variables $u_{k-d_1}, u_{k-d_1-1}, \ldots, u_{k-d_2+1}$ (assuming that $d_2 > d_1$) and applying the technique for a single time delay outlined above. Alternatively, the KKT conditions for the original formulation can be used directly as the basis of an interior-point method. The linear system to be solved at each interior-point iteration will contain not only diagonal blocks of the form in (35), but also a number of blocks at some distance from the diagonal. Some rearrangement to reduce the overall bandwidth may be possible, but expansion of the bandwidth by an amount proportional to $(d_2 - d_1)m$ is inevitable.

Of course, we can also revert to the original approach of eliminating the states $x_0, x_1, \ldots$ from the problem to obtain a problem in which the inputs $u_0, u_1, \ldots$ alone are decision variables. The cost of this approach, too, is higher than in the no-delay case, because the horizon length N usually must be increased to incorporate the effects of the delayed dynamics. One could postulate that certain processes would be effectively handled by the standard approach while others would be effectively handled by the structured approach. Perhaps the only solution is to exercise engineering judgment to decompose the full control problem into smaller problems without large delays and treat the neglected delays connecting the decomposed systems as disturbances. This issue remains unresolved and is a topic of current research.

Acknowledgments

This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Computational and Technology Research, U.S. Department of Energy, under Contract W-31-109-Eng-38, and a grant from Aspen Technology Inc. We acknowledge the support of the industrial members of the Texas-Wisconsin Modeling and Control Consortium. We are grateful to three referees of the original version of this paper, whose insightful comments improved the paper considerably.

References

[1] E. Arnold, P. Tatjewski, and P. Wolochowicz. Two methods for large-scale nonlinear optimization and their comparison on a case study of hydropower optimization. *Journal of Optimization Theory and Applications*, 81(2):221–248, 1994.

[2] D. P. Bertsekas. *Dynamic Programming.* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1987.

[3] C.-T. Chen. *Linear System Theory and Design.* Holt, Rhinehart and Winston, New York, 1984.

[4] D. Chmielewski and V. Manousiouthakis. On constrained infinite-time linear quadratic optimal control. *System & Control Letters*, 29:121–129, 1996.

[5] J. B. Congalidis, J. R. Richards, and W. H. Ray. Modeling and control of a copolymerization reactor. In *Proceedings of the American Control Conferance*, pages 1779–1793, Seattle, Washington, June 1986.

[6] R. Fletcher. *Practical Methods of Optimization.* John Wiley & Sons, New York, 1987.

[7] E. G. Gilbert and K. T. Tan. Linear systems with state and control constraints: The theory and application of maximal output admissible sets. *IEEE Transactions on Automatic Control*, 36(9):1008–1020, September 1991.

[8] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. User's guide for SOL/QPSOL: A Fortran package for quadratic programming, technical report SOL 83-12. Technical report, Systems Optimization Laboratory, Department of Operations Research, Stanford University, 1983.

[9] T. Glad and H. Jonson. A method for state and control constrained linear quadratic control problems. In *Proceedings of the 9th IFAC World Congress*, Budapest, Hungary, July 1984.

[10] W. W. Hager. Lipschitz continuity for constrained processes. *SIAM Journal on Control and Optimization*, 17(3):321–338, 1979.

[11] S. S. Keerthi. *Optimal Feedback Control of Discrete-Time Systems with State-Control Constraints and General Cost Functions.* PhD thesis, University of Michigan, 1986.

[12] A. Lim, J. Moore, and L. Faybusovich. Linearly constrained LQ and LQG optimal control. In *Proceedings of the 13th IFAC World Congress*, San Francisco, California, July 1996.

[13] D. Q. Mayne. Nonlinear model predictive control: An assessment. In *Chemical Process Control-V*, volume 93 of *AIChE Symposium Series, Number 316*, pages 217–231. CACHE Publications, 1997.

[14] E. S. Meadows, K. R. Muske, and J. B. Rawlings. Implementable model predictive control in the state space. In *Proceedings of the 1995 American Control Conference*, pages 3699–3703, 1995.

[15] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2:575–601, 1992.

[16] K. R. Muske and J. B. Rawlings. Model predictive control with linear models. *AIChE Journal*, 39(2):262–287, 1993.

[17] C. V. Rao, J. C. Campbell, J. B. Rawlings, and S. J. Wright. Efficient implementation of model predictive control for sheet and film forming processes. In *Proceedings of American Control Conference, Albuquerque, NM*, pages 2940–2944, 1997.

[18] J. B. Rawlings and K. R. Muske. Stability of constrained receding horizon control. *IEEE Transactions on Automatic Control*, 38(10):1512–1516, October 1993.

[19] N. L. Ricker, T. Subrahmanian, and T. Sim. Case studies of model-predictive control in pulp and paper production. In T. J. McAvoy, Y. Arkun, and E. Zafiriou, editors, *Proceedings of the 1988 IFAC Workshop on Model Based Process Control*, pages 13–22, Oxford, 1988. Pergamon Press.

[20] J. A. Rossiter, M. J. Rice, and B. Kouvaritakis. A robust stable state-space approach to stable predictive control stategies. In *Proceedings of the American Control Conferance*, pages 1640–1641, Albuquerque, NM, June 1997.

[21] P. O. Scokaert and J. B. Rawlings. Constrained linear quadratic regulation. Accepted for publication in *IEEE Transactions on Automatic Control*, December 1995.

[22] P. O. Scokaert and J. B. Rawlings. Infinite horizon linear quadratic control with constraints. In *Proceedings of the 13th IFAC World Congress*, San Francisco, California, July 1996.

[23] M. C. Steinbach. A structured interior point SQP method for nonlinear optimal control problems. In R. Burlirsch and D. Kraft, editors, *Computational Optimal Control*, pages 213–222. Birkhäuser, 1994.

[24] M. Sznaier and M. J. Damborg. Suboptimal control of linear systems with state and control inequality constraints. In *Proceedings of the 26th Conference on Decision and Control*, pages 761–762, 1987.

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

[26] S. J. Wright. Modified Cholesky factorizations in interior-point algorithms for linear programming. Preprint ANL/MCS-P600-0596, Mathematics and Computer Science Division, Argonne National Laboratory, May 1996.

[27] S. J. Wright. Applying new optimization algorithms to model predictive control. In *Chemical Process Control-V*, volume 93 of *AIChE Symposium Series, Number 316*, pages 147–155. CACHE Publications, 1997.

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

[29] S. J. Wright. Stability of augmented system factorizations in interior-point methods. *SIAM Journal on Matrix Analysis and its Applications*, 18:191–222, 1997.