Technical Report - EE03016
Interior-Point Methods for Linear Model Predictive Control.

A. G. Wills
Centre for Integrated Dynamics And Control (CIDAC),
University of Newcastle, Australia.
email: onyx@ecemail.newcastle.edu.au
Phone:+61 2 49215963
Fax:+61 2 49216993

and

W.P. Heath
Centre for Integrated Dynamics And Control (CIDAC),
University of Newcastle, Australia.

9 April, 2003.

Abstract

In this report we discuss linear Model Predictive Control (MPC) from a computational and algorithmic perspective. We describe two commonly encountered MPC formulations; the first includes the system model as an explicit equality constraint while the second includes the system model implicitly by projecting onto the sub-space described by this linear relation. Both formulations are expressed as Quadratic Programming (QP) problems. We discuss a popular primal-dual predictor-corrector interior-point algorithm used in solving QP problems. We go on to discuss how the interior-point strategy can exploit the structure of MPC when solving a linear sub-problem found within such algorithms. This leads to the well known Riccati recursion approach which has associated linear complexity in the prediction horizon. A simulation example is used to illustrate that the explicit formulation of MPC may indeed exhibit this property even without using a Riccati recursion.

1 Introduction

In this report, we are concerned with optimisation problems associated with linear Model Predictive Control (MPC). Linear MPC (see e.g. Muske and Rawlings 1993, Qin and Badgwell 1997) assumes a linear system model, that the constraints sets are representable via linear inequalities and that the objective function is convex quadratic. Linear MPC is appealing because the associated optimisation problem - which is typically solved at each time interval - may be expressed as a convex quadratic program. Comprehensive reviews of linear and general MPC can be found in e.g. Clarke *et al.* (1987), Garcia *et al.* (1989), Muske and Rawlings (1993), Qin and Badgwell (1997), Chen and Allgöwer (1998), Mayne *et al.* (2000) and Maciejowski (2002).

Polak *et al.* (1990) presented an algorithm based on barrier functions (in particular, they use a method of centres - see e.g. Lieu and Huard (1966)) for a general optimal control problem with control and state inequality constraints. This paper and references therein present applications of classical interior-point methods to constrained optimal control problems. However, Wright (1993) appears to be the first to have explicitly discussed new polynomial-time interior-point methods for the Quadratic Programs (QP's) and Sequential Quadratic Programs (SQP's) associated with MPC. These ideas are developed further in Wright (1997*a*), where infeasible-start interior-point methods are used. Lim *et al.* (1996) also study constrained control from an interior-point framework.

Under certain formulations, the MPC optimisation problem has exploitable structure (see e.g. Section 2.1). Exploiting this structure can, in certain cases, lead to efficient algorithms. An example of this is shown by Dunn and Bertsekas (1989) who consider unconstrained MPC and exploit the coupling between successive intervals in terms of the states and co-states. They develop a recursion algorithm with complexity that grows linearly in the horizon. This is a common theme in the literature on MPC and interior-point methods.

Wright (1993) expresses the optimality conditions on an interval-by-interval basis which leads to a block-banded system of equations; such a system must be factored and solved (at least once) at each iteration of an interior-point method. Factorisation of this matrix grows linearly in the prediction horizon. Rao *et al.* (1998) exploit a Riccati recursion approach within the interior-point framework for solving this block-banded matrix. This also results in an algorithm whose complexity grows linearly in the prediction horizon.

Gopal and Biegler (1998) offer an authoritative account of large-scale optimisation and control. Their analysis and observations are primarily from the perspective of SQP, but are pertinent to QP in general – see also Albuquerque *et al.* (1997). Gopal and Biegler (1998) also point out that certain formulations of MPC result in structure that can be exploited and consider this structure in the context of SQP.

From a practical perspective, active-set methods offer an alternative to interior-point methods. The topic of active-set methods is beyond the scope of this report, but Bartlett *et al.* (2000) offer one comparison between interior-point and active-set methods. Active-set methods can also exploit the MPC structure; Bartlett *et al.* (2000) refer to a Schur-Complement approach which allows for efficient updating of the linear sub-system solved at each iteration. Glad and Jonson (1984) also exploit the recursive nature of the MPC structure in an active-set framework.

Other approaches for solving the quadratic program that occurs in linear MPC include the use of Mixed Weight Least Squares by Rossiter and Kouvaritakis (1993). Soroush and Valluri (1999) explore a subclass of linear MPC and consider input saturation non-linearities from the perspective of optimal directionality compensation; they use a result from Barnard (1976) regarding the convergence of a simple feedback loop used for solving simply bound QP's. Syaichu-Rohman *et al.* (2003) have recently studied this type of approach. These methods seem to fall into general

category of subgradient evolution equation methods – see e.g. Brézis (1973). A number of authors have considered sub-optimal solutions to MPC (see e.g. Kouvaritakis *et al.* 2002), which is beyond the scope of this report.

The optimal solution for linear MPC is piecewise linear in the states (Zafiriou 1990). Some authors have exploited this structure to generate (off-line) the local optimal control law for each combination of active constraints (see e.g. Bemporad *et al.* 2002, Seron *et al.* 2002, Tøndel *et al.* 2003). Such methods are beyond the scope of this report. We stress, however, that this report is focused on solving the MPC optimisation problem on-line. Our intention here is to reconfirm that the interior-point approach is both highly practical and flexible when used for MPC on-line.

We should point out that interior-point methods have been used in other areas of MPC, most notably in the areas of steady-state reference calculation and robustness. Kassmann *et al.* (2000) study steady-state reference calculations where the model is subject to uncertainty. Their analysis utilises the second-order cone programming framework – see e.g. Nesterov and Todd (1997) and Nesterov and Todd (1998). Hansson (2000) studies robust MPC also in a conic programming framework. Vandenberghe *et al.* (2002) look at robust linear programming for optimal control and incorporate the Riccati recursion approach of Rao *et al.* (1998) for solving the linear system of equations associated with their algorithm. From a different perspective, interior-point geometry is exploited to generalise MPC in Wills and Heath (2002*a*), Wills and Heath (2002*b*), Heath and Wills (2002) and Wills (2003) where barrier functions are used in the cost function to change controller dynamics near constraint boundaries.

The report is organised as follows. We present one form of linear MPC in Section 2. The explicit and implicit optimisation formulations are given in Sections 2.1 and 2.2 respectively. In Section 3 we discuss Quadratic Programming in a general interior-point setting and comment on equality constraints in Section 3.2. We present a primal-dual interior-point algorithm based on Mehrotra's (1992) predictor corrector method in Section 4. In Section 5 we show how to exploit the MPC structure for interior-point methods and in Section 5.1 we reproduce Rao *et al.*'s (1998) Riccati-recursion method for solving a similar system. A brief discussion and some concluding remarks are given in section 7.

Our treatment of linear MPC is by no means complete. Throughout this report we try to point-out some very important and typical extensions to our presentation, e.g. including integral action and handling soft constraints.

2 Assumptions and Notation

The system dynamics are assumed to be given by the following linear difference equations

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t), && (1)\\ y(t) &= Cx(t). && (2) \end{aligned}$$

In the above, $x \in \mathbb{R}^{n_x}$ is the state variable, $u \in \mathbb{R}^{n_u}$ is the input variable and $y \in \mathbb{R}^{n_y}$ is the output variable.

In what follows, we will make frequent use of state and input sequences. To this end we introduce a positive integer N (later referred to as the prediction horizon), which is assumed to be constant throughout this report. Let $\mathscr{X}$ denote a sequence with $N+1$ elements where each element is a

vector of dimension n_x, i.e.

$$\mathscr{X} \in \prod_{i=0}^{N} \mathbb{R}^{n_x}, \qquad \mathscr{X} = \{x_0, \ldots, x_N\}.$$

Here the product symbol denotes a Cartesian product. Similarly let $\mathscr{U}$ denote a sequence with N elements where each element is a vector of dimension n_u given by

$$\mathscr{U} \in \prod_{i=0}^{N-1} \mathbb{R}^{n_u}, \qquad \mathscr{U} = \{u_0, \ldots, u_{N-1}\}.$$

Where convenient, we will treat $\mathscr{X}$ and $\mathscr{U}$ as $(N+1)n_x$ and Nn_u dimensional vectors respectively.

The MPC objective function is denoted by J_θ, where θ is a vector containing the co-efficients for the analytical structure of J_θ. Included in θ is the prediction horizon parameter N which also determines the sequence lengths in $\mathscr{X}$ and $\mathscr{U}$. The analytical structure of J_θ is given by

$$J_\theta(\mathscr{X}, \mathscr{U}) \quad = \quad \tfrac{1}{2}\|x_N - x_N^r\|_P^2 + \tfrac{1}{2}\sum_{i=0}^{N-1}\left(\|x_i - x_i^r\|_Q^2 + \|u_i - u_i^r\|_R^2\right). \tag{3}$$

In the above, the vectors x_i^r and u_i^r denote the i'th element of the state and input reference sequences $\mathscr{X}^r$ and $\mathscr{U}^r$ respectively (defined in a similar manner to $\mathscr{X}$ and $\mathscr{U}$). Further, the matrices P and Q are assumed to be positive semi-definite and symmetric and the matrix R is assumed to be positive definite and symmetric. The reference sequences $\mathscr{X}^r$ and $\mathscr{U}^r$ together with the co-efficients in P, Q and R are subsumed into θ. For brevity, we will often write J_θ as J.

Remark 2.1 *It is sometimes desirable to consider a separate control horizon - which is usually shorter than the prediction horizon - where the control action is allowed to vary. After such an horizon has elapsed, the control is assumed to be constant. Additionally, the states and inputs are sometimes penalised only after a certain delay time has elapsed. The corresponding objective function is given by*

$$J(\mathscr{X}, \mathscr{U}) \quad = \quad \tfrac{1}{2}\|x_N - x_N^r\|_P^2 + \tfrac{1}{2}\sum_{i=d_x}^{N-1}\|x_i - x_i^r\|_Q^2 + \tfrac{1}{2}\sum_{i=d_u}^{N_u-1}\|u_i - u_i^r\|_R^2,$$

where d_x, d_u, N and N_u are non-negative integers with obvious meaning. For ease of exposition, the following development considers only one prediction horizon - as represented by the positive scalar N - but can be formulated (or synthesised) to handle the case of separate horizons and cost delays. □

Remark 2.2 *A popular framework used extensively in industry considers an input-output model rather than a state-space model. The input-output framework is beyond the scope of this report - see Clarke* et al. *(1987) for further details. An important practical aspect made explicit in the input-output framework is integral action. Typically, the noise model assumes a "random walk" component and the cost function is expressed in terms of changes in control action rather than absolute control moves. Combined, this provides a form of integral action. Integral action is not discussed in this report - see e.g. Bitmead* et al. *(1990) for more details in the case of state-space models.* □

Remark 2.3 *Aside from the case presented in the previous remark, some formulations of MPC include a penalty on changes in the control action in addition to penalising absolute control moves.*

Changes in the control action are typically denoted by $\Delta u(t) = u(t) - u(t-1)$. *This may be adequately included into the current framework, for example, by augmenting the system equations and including the penalty in* Q *as follows. Let the original system be given by*

$$\begin{aligned} \tilde{x}(t+1) &= \tilde{A}\tilde{x}(t) + \tilde{B}u(t), \\ y(t) &= \tilde{C}\tilde{x}(t). \end{aligned}$$

The augmented system is given by

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t), \end{aligned}$$

with

$$A = \begin{bmatrix} \tilde{A} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} \tilde{B} \\ 1 \\ 1 \end{bmatrix}, \qquad C = \begin{bmatrix} \tilde{C} & 0 & 0 \end{bmatrix}.$$

Hence a penalty on $\Delta u(t)$ *may be straightforwardly included into* Q. *This is not necessarily the most efficient manner to include a penalty on* Δu. □

At each time interval t, the current state of the system $x(t)$ is used as the initial state x_0 for controller predictions and the optimal input trajectory is calculated using this initial state. Typically, $x(t)$ is not available and an estimate of the state $\hat{x}(t)$ is used instead. We assume that $\hat{x}(t)$ is a reasonable estimate in that it satisfies all physical limitations of the system. We make no further comment on constrained receding horizon estimation which can be used to ensure the latter (see e.g. Rao *et al.* (2003)). Since the system is assumed to be time-invariant, no further reference to the time instance t is made. Instead, the initial state is denoted by $\bar{x}$ and the first state in $\mathscr{X}$ is required to equal $\bar{x}$. The optimal input trajectory is optimal in the sense that it minimises J subject to a number of constraints, namely:

- The first element in $\mathscr{X}$ should equal the initial state vector $\bar{x}$, i.e.

$$x_0 = \bar{x}.$$

- Each successive element in $\mathscr{X}$ should satisfy the system equations (1), i.e.

$$x_{i+1} = Ax_i + Bu_i, \quad \text{for } i = 1, \ldots, N.$$

- Each element in $\mathscr{U}$ and the last N elements in $\mathscr{X}$ should respect input, state and terminal state constraints respectively, i.e.

$$\begin{aligned} u_i &\in \mathbb{U}, \quad \text{for } i = 0, \ldots, N-1, \\ x_i &\in \mathbb{X}, \quad \text{for } i = 1, \ldots, N-1, \\ x_N &\in \mathbb{X}_F. \end{aligned}$$

Remark 2.4 *It should be noted that many results in the literature assume (whether explicitly or implicitly) the existence of feasible solutions to the MPC optimisation problem. Hard constraints often render this optimisation problem infeasible(Qin and Badgwell 1997). However, in many practical situations certain constraint violations are tolerable, but should be kept to a minimum. It is often convenient to categorise the types of constraints encountered in MPC as either hard or soft. Typically, hard constraints must be strictly satisfied while soft constraints should be satisfied*

if possible. Inclusion of constraints and their respective categorisation as either hard or soft is a control engineering decision – see e.g. Qin and Badgwell (1997). For example, actuator limits are often included as hard constraints on the inputs. Meanwhile, state constraints are often deemed to be soft since strict satisfaction may not always be possible.

Scokaert and Rawlings (1999) summarise two different strategies found in the literature for handling the infeasible case (see also Maciejowski (2002)). One such strategy includes a penalty term in the cost function which penalises soft constraint violations. This is called the "soft-constraint" strategy and is often formulated using an exact penalty. The desirable property of an exact penalty method is that all constraints are strictly satisfied if possible. If not, then violations of the soft constraints are penalised in the cost. Exact penalty functions are summarised in e.g. Fletcher (1987). The exact penalty property is usually guaranteed by including a suitably chosen linear term in the cost function. If the linear term is not sufficiently large, then the exact penalty property may be lost. Unfortunately it is difficult to determine a suitable linear term apriori.

If the MPC optimisation problem is feasible then – provided the linear term is sufficiently large – the exact penalty approach does not interact with the controller dynamics. If infeasible, however, a tradeoff exists between satisfying constraints and returning to the origin of the system. By choosing the linear term to be large, emphasis is inherently placed on satisfying constraints. Therefore, system dynamics may be considerably awry.

In Wills and Heath (2003), we propose a two step strategy which subsumes the exact penalty approach as a special case. The first step determines feasibility of the MPC optimisation problem and the second step solves a suitably chosen MPC problem depending on the result of the first step. Separating the problem into two stages allows for a more intuitive tuning procedure.

For brevity, we do not consider soft constraints further in this report. Note however, that the following discussion may be straightforwardly extended for this important case. Moreover, we point the interested reader to Rao et al. *(1998) who consider the exact penalty approach in their discussion of interior-point methods for linear MPC; they show that the MPC structure can be exploited even with the additional terms incurred using a soft constraint formulation.* □

State, terminal state and input constraint sets are assumed to have the following structure respectively,

$$\begin{aligned} \mathbb{X} &= \{x \in \mathbb{R}^{n_x} : L_x x \preceq k_x\}, \\ \mathbb{X}_F &= \{x \in \mathbb{R}^{n_x} : L_F x \preceq k_F\}, \\ \mathbb{U} &= \{u \in \mathbb{R}^{n_u} : L_u u \preceq k_u\}. \end{aligned}$$

Use of the "$\preceq$" symbol denotes element-wise inequality.

Using the above definitions, we define the MPC optimisation problem as follows.

Definition 2.1 *Given some initial state $\bar{x}$, solve (if possible) the following minimisation problem,*

$$\begin{aligned} (\mathcal{MPC}): \qquad \min_{\mathscr{X},\mathscr{U}} \quad & J(\mathscr{X},\mathscr{U}) \\ \text{s.t.} \quad & x_0 = \bar{x}, \\ & x_{i+1} = Ax_i + Bu_i, \\ & u_i \in \mathbb{U}, \ \textit{for } i = 0,\ldots,N-1, \\ & x_i \in \mathbb{X}, \ \textit{for } i = 1,\ldots,N-1, \\ & x_N \in \mathbb{X}_F. \end{aligned}$$

□

In Section 2.1 below, we outline one formulation of $(\mathcal{MPC})$ where the linear model is included explicitly as an equality constraint. In Section 2.2 we discuss an alternative formulation where the linear model is subsumed into the cost function.

2.1 Explicit Formulation

Treating $\mathscr{X}$ and $\mathscr{U}$ as vectors, we can express J as

$$J(\mathscr{X},\mathscr{U}) \;=\; \tfrac{1}{2}\|\mathscr{X}-\mathscr{X}^r\|^2_{\bar{Q}} + \tfrac{1}{2}\|\mathscr{U}-\mathscr{U}^r\|^2_{\bar{R}}.$$

In the above, $\bar{Q}$ is given by the block diagonal matrix $\bar{Q} = \mathrm{diag}\{Q,\ldots,Q,P\}$ and $\bar{R}$ is given similarly by $\bar{R} = \mathrm{diag}\{R,\ldots,R\}$.

The system dynamics can be expressed as

$$\bar{A}\mathscr{X} + \bar{B}\mathscr{U} \;=\; b, \tag{4}$$

where $\bar{A}$ is an $(N+1)n_x \times (N+1)n_x$ matrix, $\bar{B}$ is an $(N+1)n_x \times Nn_u$ matrix and b is an $(N+1)n_x$ vector given respectively by

$$\bar{A} \;=\; \begin{bmatrix} I & 0 & 0 & \cdots & 0 \\ -A & I & 0 & \cdots & 0 \\ 0 & -A & I & \cdots & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & \cdots & & -A & I \end{bmatrix}, \qquad \bar{B} \;=\; \begin{bmatrix} 0 & 0 & \cdots & 0 \\ -B & 0 & \cdots & 0 \\ 0 & -B & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & & -B \end{bmatrix}, \qquad b \;=\; \begin{bmatrix} \bar{x} \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

The inequality constraints may be expressed as

$$\begin{aligned} L_{\mathscr{X}}\mathscr{X} &\preceq k_{\mathscr{X}}, \\ L_{\mathscr{U}}\mathscr{U} &\preceq k_{\mathscr{U}}, \end{aligned}$$

where $L_{\mathscr{X}}$, $k_{\mathscr{X}}$, $L_{\mathscr{U}}$ and $k_{\mathscr{U}}$ are given by

$$L_{\mathscr{X}} \;=\; \begin{bmatrix} 0 & L_x & 0 & \cdots & 0 \\ 0 & 0 & L_x & \cdots & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & \cdots & & 0 & L_F \end{bmatrix}, \qquad L_{\mathscr{U}} \;=\; \begin{bmatrix} L_u & 0 & \cdots & 0 \\ 0 & L_u & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & & L_u \end{bmatrix},$$

$$k_{\mathscr{X}} \;=\; \begin{bmatrix} k_x \\ k_x \\ \vdots \\ k_F \end{bmatrix}, \qquad k_{\mathscr{U}} \;=\; \begin{bmatrix} k_u \\ k_u \\ \vdots \\ k_u \end{bmatrix}.$$

Problem $(\mathcal{MPC})$ can be restated as follows.

Definition 2.2 *Given some initial state $\bar{x}$, solve (if possible) the following minimisation problem.*

$$\begin{aligned} (\mathcal{MPC}_E): \qquad \min_{\mathscr{X},\mathscr{U}} \quad & \tfrac{1}{2}\|\mathscr{X}-\mathscr{X}^r\|^2_{\bar{Q}} + \tfrac{1}{2}\|\mathscr{U}-\mathscr{U}^r\|^2_{\bar{R}} \\ \text{s.t.} \quad & \bar{A}\mathscr{X} + \bar{B}\mathscr{U} \;=\; b, \\ & L_{\mathscr{X}}\mathscr{X} \;\preceq\; k_{\mathscr{X}}, \\ & L_{\mathscr{U}}\mathscr{U} \;\preceq\; k_{\mathscr{U}}. \end{aligned}$$

□

2.2 Implicit Formulation

Problem $(\mathcal{MPC}_E)$ can be interpreted as minimising a convex quadratic objective subject to inequality constraints with the further requirement that the solution lie in a subspace described by equation (4). By projecting $(\mathcal{MPC}_E)$ onto this subspace we can reduce the overall dimension. In Section 3.2 we discuss such a projection in more detail. One particularly common and convenient projection is described as follows.

Since $\bar{A}$ has full rank, then equation (4) may be equivalently stated as

$$\mathscr{X} \quad = \quad \bar{A}^{-1}(b - \bar{B}\mathscr{U}). \tag{5}$$

Furthermore, the inverse of $\bar{A}$ has structure given by

$$\bar{A}^{-1} \quad = \quad \begin{bmatrix} I & 0 & \cdots & \cdots & 0 \\ A & I & 0 & \cdots & 0 \\ A^2 & A & I & \cdots & 0 \\ \vdots & & & \ddots & \\ A^N & A^{N-1} & A^{N-2} & \cdots & I \end{bmatrix} \tag{6}$$

Expanding (5) in terms of (6) gives the following expression for $\mathscr{X}$,

$$\mathscr{X} \quad = \quad \Lambda\bar{x} + \Phi\mathscr{U}, \tag{7}$$

where Λ and Φ are given by

$$\Lambda \quad = \quad \begin{bmatrix} I \\ A \\ \vdots \\ A^N \end{bmatrix}, \qquad \Phi \quad = \quad \begin{bmatrix} 0 & \cdots & \cdots & 0 \\ B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & & \ddots & \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix}$$

Using relation (7), we may express $(\mathcal{MPC})$ as follows.

Definition 2.3 *Given some initial state $\bar{x}$, solve (if possible) the following minimisation problem.*

$$\begin{aligned} (\mathcal{MPC}_I): \qquad \min_{\mathscr{U}} \quad & \tfrac{1}{2}\|\Lambda\bar{x} + \Phi\mathscr{U} - \mathscr{X}^r\|^2_{\bar{Q}} + \tfrac{1}{2}\|\mathscr{U} - \mathscr{U}^r\|^2_{\bar{R}} \\ \text{s.t.} \quad & L_{\mathscr{X}}(\Lambda\bar{x} + \Phi\mathscr{U}) \preceq k_{\mathscr{X}}, \\ & L_{\mathscr{U}}\mathscr{U} \preceq k_{\mathscr{U}}. \end{aligned}$$

□

In the case of unstable system models, $(\mathcal{MPC}_I)$ can become ill-conditioned (see e.g. Rao *et al.* 1998, Tenny *et al.* 2002). Indeed, since A has eigenvalues outside the unit circle in this case, then $\max\{\text{eig}(A^i)\}$ can become large, even for small values of i. One possibility (Keerthi 1986) is to introduce a dummy variable $v(k)$ and exploit a stabilising state feedback gain K. Suppose we write

$$u(k) = -Kx(k) + v(k)$$

with $A_{cl} = A - BK$ stable. Let $\mathscr{V}$ denote a sequence with N elements where each element is a vector of dimension n_u given by

$$\mathscr{V} \in \prod_{i=0}^{N-1} \mathbb{R}^{n_u}, \qquad \mathscr{V} = \{v_0, \ldots, v_{N-1}\}.$$

so that

$$\mathscr{U} = \bar{K}\mathscr{X} + \mathscr{V}$$

with

$$\bar{K} = \begin{bmatrix} -K & & 0 \\ & \ddots & \vdots \\ & -K & 0 \end{bmatrix}$$

We can thus express $(\mathcal{MPC})$ as

Definition 2.4 *Given some initial state $\bar{x}$, solve (if possible) the following minimisation problem.*

$$\begin{array}{lll} (\mathcal{MPC}_{Ecl}): & \min\limits_{\mathscr{X},\mathscr{V}} & \frac{1}{2}\|\mathscr{X} - \mathscr{X}^r\|^2_{\bar{Q}} + \frac{1}{2}\|\bar{K}\mathscr{X} + \mathscr{V} - \bar{K}\mathscr{X}^r - \mathscr{U}^r\|^2_{\bar{R}} \\ & \text{s.t.} & (\bar{A} + \bar{B}\bar{K})\mathscr{X} + \bar{B}\mathscr{V} = b, \\ & & L_{\mathscr{X}}\mathscr{X} \preceq k_{\mathscr{X}}, \\ & & L_{\mathscr{U}}\mathscr{V} \preceq k_{\mathscr{U}} - L_{\mathscr{U}}\bar{K}\mathscr{X}. \end{array}$$

□

We can thus find a numerically stable implicit formulation by substituting

$$\begin{aligned} \mathscr{X} &= (\bar{A} - \bar{B}\bar{K})^{-1}(b - \bar{B}\mathscr{V}) \\ &= \Lambda_{cl}\bar{x} + \Phi_{cl}\mathscr{V} \end{aligned}$$

with

$$\Lambda_{cl} = \begin{bmatrix} I \\ A_{cl} \\ \vdots \\ A_{cl}^N \end{bmatrix}, \qquad \Phi_{cl} = \begin{bmatrix} 0 & \cdots & \cdots & 0 \\ B & 0 & \cdots & 0 \\ A_{cl}B & B & \cdots & 0 \\ \vdots & & \ddots & \\ A_{cl}^{N-1}B & A_{cl}^{N-2}B & \cdots & B \end{bmatrix}$$

Specifically, we may express $(\mathcal{MPC})$ as follows.

Definition 2.5 *Given some initial state $\bar{x}$, solve (if possible) the following minimisation problem.*

$$\begin{array}{lll} (\mathcal{MPC}_{Icl}): & \min\limits_{\mathscr{U}} & \frac{1}{2}\|\Lambda_{cl}\bar{x} + \Phi_{cl}\mathscr{V} - \mathscr{X}^r\|^2_{\bar{Q}} + \frac{1}{2}\|\bar{K}\Lambda_{cl}\bar{x} + (\bar{K}\Phi_{cl} + I)\mathscr{V} - \bar{K}\mathscr{X}^r - \mathscr{U}^r\|^2_{\bar{R}} \\ & \text{s.t.} & L_{\mathscr{X}}(\Lambda_{cl}\bar{x} + \Phi_{cl}\mathscr{V}) \preceq k_{\mathscr{X}}, \\ & & L_{\mathscr{U}}(\bar{K}\Lambda_{cl}\bar{x} + (\bar{K}\Phi_{cl} + I)\mathscr{V}) \preceq k_{\mathscr{U}}. \end{array}$$

□

A natural choice of K is the equivalent state gain when there are no constraints. In this case the unconstrained minimum of the cost function occurs where $\mathscr{V} = 0$. In particular, if the original MPC problem is structured as (3) with P, Q and R satisfying the discrete Riccati equation

$$P = A^T PA - A^T PB(R + B^T PB)^{-1}B^T PA + Q$$

then the natural choice for K is

$$K = (R + B^T PB)^{-1}B^T PA$$

An alternative approach to reducing numerical ill-conditioning it to use a numerically stable projection. In the above we have restricted the projection to the type given in Equation (5). Alternatively, we could introduce a dummy variable $\mathscr{E}$ and represent $\mathscr{X}$ and $\mathscr{U}$ by

$$\begin{aligned} \mathscr{X} &= S^T\mathscr{E} + b, \\ \mathscr{U} &= T^T\mathscr{E}, \end{aligned}$$

where S and T have the following properties,

$$\begin{bmatrix} \bar{A} & \bar{B} \end{bmatrix} \begin{bmatrix} S \\ T \end{bmatrix} = 0, \qquad \begin{bmatrix} S^T & T^T \end{bmatrix} \begin{bmatrix} S \\ T \end{bmatrix} = I.$$

Hence $\mathscr{X}$ and $\mathscr{U}$ can be replaced by there respective affine functions in $\mathscr{E}$ and the resulting problem solved for $\mathscr{E}$. Specifically, we may express $(\mathcal{MPC})$ as follows.

Definition 2.6 *Given some initial state $\bar{x}$, solve (if possible) the following minimisation problem.*

$$\begin{aligned} (\mathcal{MPC}_{IP}): \qquad \min_{\mathscr{E}} \quad & \tfrac{1}{2}\|S^T\mathscr{E} + b - \mathscr{X}^r\|^2_{\bar{Q}} + \tfrac{1}{2}\|T^T\mathscr{E} - \mathscr{U}^r\|^2_{\bar{R}} \\ \text{s.t.} \quad & L_{\mathscr{X}}(S^T\mathscr{E} + b) \preceq k_{\mathscr{X}}, \\ & L_{\mathscr{U}}(T^T\mathscr{E}) \preceq k_{\mathscr{U}}. \end{aligned}$$

□

Of course $\mathscr{X}$ and $\mathscr{U}$ can then be recovered using the same equations given above. A caveat of this approach is that constraints will most likely become dense, thus effecting overall computational efficiency (see also Section 3.2).

Any potential numerical errors in the formation of Φ (or equivalent) are exacerbated when the Hessian matrix of the cost function is formed. In $(\mathcal{MPC}_I)$, $(\mathcal{MPC}_{Icl})$ and $(\mathcal{MPC}_{IP})$ the Hessians are given respectively as

$$\begin{aligned} H_I &= \Phi^T\bar{Q}\Phi + \bar{R} \\ H_{Icl} &= \Phi_{cl}^T\bar{Q}\Phi_{cl} + (I + \bar{K}\Phi_{cl})^T\bar{R}(I + \bar{K}\Phi_{cl}) \\ H_{IP} &= S^T\bar{Q}S + T^T\bar{R}T. \end{aligned}$$

One approach for reducing numerical ill-conditioning is to avoid forming the Hessian matrix explicitly, but instead represent it as, respectively,

$$\begin{aligned} H_I &= G_I^T G_I, \quad & G_I^T &= \begin{bmatrix} \Phi^T\bar{Q}^{1/2} & \bar{R}^{1/2} \end{bmatrix} \\ H_{Icl} &= G_{Icl}^T G_{Icl}, \quad & G_{Icl}^T &= \begin{bmatrix} \Phi_{cl}^T\bar{Q}^{1/2} & (I + \bar{K}\Phi_{cl})^T\bar{R}^{1/2} \end{bmatrix}, \\ H_{IP} &= G_{IP}^T G_{IP}, \quad & G_{IP}^T &= \begin{bmatrix} S^T\bar{Q}^{1/2} & T^T\bar{R}^{1/2} \end{bmatrix}. \end{aligned}$$

When solving the linear sub-system found in interior-point methods (see Equation (21) below), this structure can be exploited. For example, Equation (21) can be expressed as (note that the S matrix below is not that defined in this section),

$$H + L^T S^{-1} Z L = \begin{bmatrix} G^T & \left(S^{-1}Z\right)^{1/2} L^T \end{bmatrix} \begin{bmatrix} G \\ \left(S^{-1}Z\right)^{1/2} L \end{bmatrix}.$$

Hence, the Cholesky factor of $H + L^T S^{-1} Z L$ may be obtained from the QR factorisation as follows,

$$Q_F R_F = \begin{bmatrix} G \\ \left(S^{-1}Z\right)^{1/2} L \end{bmatrix},$$

where R_F is also the Cholesky factor in this case (see e.g. Golub and Van Loan 1996). Therefore, the Hessian matrix H is not explicitly formed, thus reducing numerical ill-conditioning.

3 Quadratic Programming and Interior Point Methods

The following discussion is based (primarily) on Gopal and Biegler (1998), Wright (1997*a*), Wright (1999), Wright (1997*b*), Wright (1995), Vanderbei (1999) and Mehrotra (1992). The interior-point algorithm presented in Section 4 is based on Mehrotra's (1992) predictor-corrector algorithm , which is widely accepted as a highly practical and efficient algorithm.

In Section 3.1, we describe a quadratic programming problem and give the KKT (Karush-Kuhn-Tucker - see e.g. (Mangasarian 1969)) optimality conditions for this problem. These conditions may be restated as a system of non-linear equalities with some variables constrained to be non-negative. Loosely speaking, the interior-point approach attempts to solve a related system of equalities whilst strictly honouring the non-negativity constraints. In particular, the system of non-linear equalities is linearised about the current point and the resulting *linear* system of equations is solved. Computational efficiency of interior-point methods depends heavily upon how efficiently we can solve this linear subproblem.

3.1 Quadratic Programming Optimality Conditions

Consider the following quadratic program,

$$
\begin{aligned}
(\mathcal{QP}): \qquad \min_{x} \quad & \tfrac{1}{2}x^T H x + f^T x + c_0, \\
\text{s.t.} \quad & Lx \preceq k, \\
& Ax = b.
\end{aligned}
$$

Let $x \in \mathbb{R}^n$. It is assumed that L is an $m \times n$ matrix and A is an $m_{eq} \times n$ matrix with $m_{eq} \leq n$. It is further assumed that A has independent rows and $b = Ax_0$ for some x_0. The matrix H is assumed to be positive semi-definite and symmetric.

Optimality conditions for $(\mathcal{QP})$ are given by (see e.g. Boyd and Vandenberghe 2002),

$$
\begin{aligned}
Hx + f + L^T z + A^T y &= 0, && (8)\\
Lx + s &= k, && (9)\\
Ax &= b, && (10)\\
z^T s &= 0, && (11)\\
z &\succeq 0, && (12)\\
s &\succeq 0.. && (13)
\end{aligned}
$$

Here y represents the Lagrange multipliers for the equality constraints, z represents the Lagrange multipliers for the inequality constraints and the slack variable s has been introduced for convenience.

We will be concerned with a related problem where the complementarity condition $z^T s = 0$ is replaced by the following relation,

$$
ZSe = \mu e,
$$

where $Z = \text{diag(z)}$, $S = \text{diag(s)}$, e is a vector of all ones and $\mu > 0$ is the "target" (the parameter μ goes by various names but it is instructive to think of μ as the target). Clearly, as $\mu \to 0$, the two problems coincide.

For $z, s \succeq 0$, the related problem is to find x, y, z, s such that

$$F_\mu(x,y,z,s) \quad = \quad \left\{ \begin{array}{c} Hx + f + L^T z + A^T y \\ Lx + s - k \\ Ax - b \\ ZSe - \mu e \end{array} \right\} \quad = \quad 0.$$

The above system of non-linear equations can be solved iteratively via, say, Newton's method. Indeed, interior-point methods take this approach. However, exact solutions for each target value of μ are not required. Rather, μ is adaptively reduced at each iteration aiming in the limit for $\mu = 0$. In this case the optimal conditions for $(\mathcal{QP})$ are recovered.

Applying Newton's method to $F_\mu(\cdot)$ can be interpreted as follows: find a search direction $\Delta p = [\Delta x^T \ \Delta y^T \ \Delta z^T \ \Delta s^T]^T$ that satisfies

$$J(x,y,z,s)\Delta p \quad = \quad -F\mu(x,y,z,s), \tag{14}$$

where $J(.)$ represents the Jacobian matrix for $F\mu(.)$. Expanding (14) gives that

$$\begin{bmatrix} H & A^T & L^T & 0 \\ A & 0 & 0 & 0 \\ L & 0 & 0 & I \\ 0 & 0 & S & Z \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta s \end{bmatrix} \quad = \quad \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{bmatrix}, \tag{15}$$

where

$$\begin{array}{rcl} r_1 & = & -Hx - f - L^T z - A^T y, \\ r_2 & = & b - Ax, \\ r_3 & = & k - Lx - s, \\ r_4 & = & \sigma\mu e - ZSe. \end{array} \tag{16}$$

It is possible to reduce equation (15) to a more convenient form (known as the Augmented form) as follows. From the last equation it follows that

$$\Delta s \quad = \quad Z^{-1}(r_4 - S\Delta z). \tag{17}$$

Using the above expression for Δs, then (15) reduces to

$$\begin{bmatrix} H & A^T & L^T \\ A & 0 & 0 \\ L & 0 & -Z^{-1}S \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} \quad = \quad \begin{bmatrix} r_1 \\ r_2 \\ r_3 - Z^{-1}r_4 \end{bmatrix}. \tag{18}$$

After solving the above system for $(\Delta x, \Delta y, \Delta z)$, Δs is obtained from (17). If there are no explicit equality constraints then (18) becomes

$$\begin{bmatrix} H & L^T \\ L & -Z^{-1}S \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta z \end{bmatrix} \quad = \quad \begin{bmatrix} r_1 \\ r_3 - Z^{-1}r_4 \end{bmatrix}. \tag{19}$$

In certain cases it may be desirable to eliminate Δz as

$$\Delta z = S^{-1}Z(L\Delta x - r_3 + Z^{-1}r_4), \tag{20}$$

resulting in the following system for Δx,

$$\left[H + L^T S^{-1} Z L\right] \Delta x = r_1 + L^T S^{-1} Z (r_3 - Z^{-1} r_4). \tag{21}$$

Note that Δz can be obtained from (20) and Δs from (17). System (21) is positive definite and may be factored using a Cholesky factorisation. Some care must be taken when μ is small since

then $S^{-1}Z$ may have very large/small components (this phenomenon is common to all interior-point methods). Wright (1999) describes a modified Cholesky factorisation which is stable under these circumstances (see also Wright (1997*b*) for stability of the augmented system in (18)). It has been observed (see e.g. Vanderbei 1999) that reducing system (19) to (21) may introduce unnecessary computational error.

The choice of factorisation method often depends on the problem structure (sparsity, presence of equality constraints etc.). For systems where (15) is sparse then a direct Gaussian elimination approach is often used (see e.g. Gertz and Wright 2001). The factorisation step is usually the most computationally expensive operation at each iteration of an interior point algorithm. Most practical algorithms perform one factorisation and possibly several solve operations per cycle.

3.2 A Note on Linear Equality Constraints

The equality constraints in $(\mathcal{QP})$ may be handled in a number of ways. One common approach is to represent equality constraints as a two sided inequality with the same bound on both sides, i.e. $b \preceq Ax \preceq b$. A variant of this approach is used in the LOQO package by Vanderbei (1999). Another approach is to include these constraints explicitly and derive optimality conditions for both inequality and equality constraints as done in Section 3.1 and in the OOQP package by Gertz and Wright (2001). Both approaches have the benefit of preserving original problem structure, and in particular, sparsity is preserved.

A possible disadvantage of the above approach is observed when A is dense with almost full rank. When there is a significant number of equality constraints m_{eq} (with respect to n), then equation (18) may be large. In addition, if A is dense, then this system may be expensive to solve. Further, when m_{eq} is large then the optimisation problem has few degrees of freedom left to satisfy inequality constraints (the case of $m = n$ is excluded since $Ax = b$ has a unique solution in this case - hence the optimisation problem has only one *possible* solution). However, this is not reflected (computationally) in (18) which grows with m_{eq}.

An alternative approach (under these circumstances) is to project $(\mathcal{QP})$ onto the space described by $Ax = b$. In general this will effect sparsity, however with some care this effect may be reduced. The benefit is that each iteration involves the solution of a reduced order system of the form (c.f. (19)),

$$\begin{bmatrix} \tilde{H} & \tilde{L}^T \\ \tilde{L} & -D \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \end{bmatrix},$$

where D is a positive definite diagonal matrix and $\tilde{H}$ and $\tilde{L}$ are described as follows. Let $A^{\perp}$ be an $n \times (n - m_{eq})$ matrix denoting the orthogonal complement of A. Since $b = Ax_0$ for some x_0, then $A(x - x_0) = 0$ and hence $x = A^{\perp}\tilde{x} + x_0$ for some $\tilde{x}$. Making this substitution for x in $(\mathcal{QP})$ produces the following optimisation problem,

$$(\mathcal{QP}_{\tilde{P}}): \quad \min_{\tilde{x}} \tfrac{1}{2}\tilde{x}^T\tilde{H}\tilde{x} + \tilde{f}^T\tilde{x} + \tilde{c}_0, \quad \text{subject to} \quad \tilde{L}\tilde{x} \preceq \tilde{k},$$

where,

$$\begin{aligned} \tilde{H} &= (A^{\perp})^T H A^{\perp}, \\ \tilde{f} &= (A^{\perp})^T [Hx_0 + f], \\ \tilde{c}_0 &= c_0 + f^T x_0 + \tfrac{1}{2} x_0^T H x_0, \\ \tilde{L} &= LA^{\perp}, \\ \tilde{k} &= k - Lx_0. \end{aligned}$$

Hence, in this case, it suffices to consider a problem with inequality constraints only.

4 Primal-Dual Path-Following Algorithm

The following algorithm is based (primarily) on Mehrotra's (1992) predictor-corrector method. This algorithm requires one factorisation operation and two solve operations per cycle. Suitable starting points and termination conditions for this algorithm are discussed in Sections 4.1 and 4.2 respectively.

Algorithm 1 *Given an initial point* $(x_0, y_0, z_0, s_0) \in \mathbb{R}^n \times \mathbb{R}^{m_{eq}} \times \mathbb{R}^m_{++} \times \mathbb{R}^m_{++}$ *(see Section 4.2), iterate the following (starting with* $i = 0$*),*

START:

1. *Check termination conditions - see Section 4.2. If satisfied then stop.*
2. *Obtain the "predictor" direction by solving,*

$$\begin{bmatrix} H & A^T & L^T & 0 \\ A & 0 & 0 & 0 \\ L & 0 & 0 & I \\ 0 & 0 & S_i & Z_i \end{bmatrix} \begin{bmatrix} \Delta x_i^p \\ \Delta y_i^p \\ \Delta z_i^p \\ \Delta s_i^p \end{bmatrix} = \begin{bmatrix} r_{1,i} \\ r_{2,i} \\ r_{3,i} \\ r_{4,i}^p \end{bmatrix},$$

with

$$\begin{aligned} r_{1,i} &= -Hx_i - f - L^T z_i - A^T y_i, \\ r_{2,i} &= b - Ax_i, \\ r_{3,i} &= k - Lx_i - s_i, \\ r_{4,i}^p &= -Z_i S_i e. \end{aligned}$$

3. *Define the distance to the boundary* α_i^p *along the predictor direction as,*

$$\alpha_i^p = \left(\max \left\{ \max_j \left\{ \frac{-\Delta z_i^p(j)}{z_i(j)} \right\}, \max_j \left\{ \frac{-\Delta s_i^p(j)}{s_i(j)} \right\}, 1 \right\} \right)^{-1}.$$

4. *Define the adaptive "centring" parameter* σ_i *as,*

$$\sigma_i = \left(\frac{(z_i + \alpha_i^p \Delta z_i^p)^T (s_i + \alpha_i^p \Delta s_i^p)}{m \mu_i} \right)^3.$$

where

$$\mu_i = z_i^T s_i / m.$$

5. *Obtain the combined "predictor-corrector" direction by solving,*

$$\begin{bmatrix} H & A^T & L^T & 0 \\ A & 0 & 0 & 0 \\ L & 0 & 0 & I \\ 0 & 0 & S_i & Z_i \end{bmatrix} \begin{bmatrix} \Delta x_i^{pc} \\ \Delta y_i^{pc} \\ \Delta z_i^{pc} \\ \Delta s_i^{pc} \end{bmatrix} = \begin{bmatrix} r_{1,i} \\ r_{2,i} \\ r_{3,i} \\ r_{4,i}^{pc} \end{bmatrix},$$

with

$$r_{4,i}^{pc} = \sigma_i \mu_i e - Z_i S_i e - \Delta Z_i^p \Delta S_i^p e.$$

where $\Delta Z_i^p = \mathrm{diag}(\Delta z_i^p)$ *and* $\Delta S_i^p = \mathrm{diag}(\Delta s_i^p)$.

6. *Obtain the final step length α_i as follows (see e.g. Gertz and Wright 2001). Choose a positive number γ_f such that $0 < \gamma_f \ll 1$ and compute the following,*

$$\begin{aligned}
\alpha_i^z &= \left(\max_j \left\{\frac{-\Delta z_i^p(j)}{z_i(j)}\right\}\right)^{-1}, \\
\alpha_i^s &= \left(\max_j \left\{\frac{-\Delta s_i^p(j)}{s_i(j)}\right\}\right)^{-1}, \\
j_i^z &= \text{argmax}_j \left\{\frac{-\Delta z_i^p(j)}{z_i(j)}\right\}, \\
j_i^s &= \text{argmax}_j \left\{\frac{-\Delta s_i^p(j)}{s_i(j)}\right\}.
\end{aligned}$$

Let α_i^{pc} be given by,

$$\alpha_i^{pc} = \min\{\alpha_i^z, \alpha_i^s, 1\}.$$

Define μ_i^{pc} as,

$$\mu_i^{pc} = \frac{1}{m}(z_i + \alpha_i^{pc}\Delta z_i^{pc})^T(s_i + \alpha_i^{pc}\Delta s_i^{pc}).$$

Let α_i be given by,

$$\alpha_i = \begin{cases} 1 & \text{if } \min\{\alpha_i^z, \alpha_i^s\} > 1, \\ \left(\frac{\gamma_f \mu_i^{pc}}{s_i(j_i^z) + \alpha_i^{pc}\Delta s_i^{pc}(j_i^z)} - z_i(j_i^z)\right)/\Delta z_i^{pc}(j_i^z) & \text{if } \alpha_i^z < \alpha_i^s, \\ \left(\frac{\gamma_f \mu_i^{pc}}{z_i(j_i^s) + \alpha_i^{pc}\Delta z_i^{pc}(j_i^s)} - s_i(j_i^s)\right)/\Delta s_i^{pc}(j_i^s) & \text{otherwise.} \end{cases}$$

Finally, if $\alpha_i < (1-\gamma_f)\alpha_i^{pc}$ then set $\alpha_i = (1-\gamma_f)\alpha_i^{pc}$, else set $\alpha_i \leftarrow (1-\gamma_f)\alpha_i$.

7. *Form the new iterates,*

$$\begin{aligned}
x_{i+1} &= x_i + \alpha_i \Delta x_i^{pc}, \\
y_{i+1} &= y_i + \alpha_i \Delta y_i^{pc}, \\
z_{i+1} &= z_i + \alpha_i \Delta z_i^{pc}, \\
s_{i+1} &= s_i + \alpha_i \Delta s_i^{pc}.
\end{aligned}$$

8. *Increment i by one and repeat from step 1.*

END. □

4.1 Initial Points

The issue of finding good initial conditions has received much attention in the interior point literature (see e.g. Wright 1997*a*, Ye 1997). Gertz and Wright (2001) suggest that initial points should be close to the central path whilst almost satisfying the equality conditions (8)–(10). The latter conditions are henceforth referred to as feasibility conditions. Let η_∞ denote the largest absolute value from the problem data, i.e.

$$\eta_\infty = \|(H, f, L, k, A, b)\|_\infty.$$

Initialise the variables as follows,

$$\begin{aligned}
x &= 0, \\
y &= 0, \\
z &= \sqrt{\eta_\infty} e, \\
s &= \sqrt{\eta_\infty} e.
\end{aligned}$$

Next take a full step in the predictor direction as defined in algorithm 1. This may produce negative values in z and/or s - which must therefore be adjusted. One proposal is to add a constant to all values in z and s such that the resulting pair has roughly equal positive components. A further goal is to choose this constant such that the complementarity gap (defined as $\mu = z^T s/m$) is significantly larger than the relative feasibility gap. One measure of feasibility (used in the OOQP code from Gertz and Wright (2001)) is as follows,

$$\vartheta = \frac{\|(r_1, r_2, r_3)\|_\infty}{\eta_\infty},$$

where r_1, r_2 and r_3 are given in (16). If μ/ϑ is large then x, y, z and s resulting from this procedure are deemed suitable initial points. In particular, one heuristic is as follows: after taking the full predictor step then define an offset value β as,

$$\beta \quad = \quad 1000 + 2\max\left\{\max\{-z\}, \max\{-s\}, 0\right\}.$$

then update z and s as,

$$\begin{aligned} z &\leftarrow z + \beta e, \\ s &\leftarrow s + \beta e. \end{aligned}$$

4.2 Termination Conditions

Let ϑ^k and μ^k denote the relative residual and complementarity measures at iteration k of the algorithm. Define the centrality measure γ^k as,

$$\gamma^k = \|Z_k S_k e\|_\infty.$$

Conditions for terminating Algorithm 1 can be categorised as follows:

- If the relative residual measure ϑ^k is sufficiently small and the complementarity measure μ^k is sufficiently close to 0 and the centrality measure γ^k is sufficiently small then terminate with a successful condition flag. In particular if $\vartheta^k \leq \epsilon_r$ and $\mu^k \leq \epsilon_\mu$ and $\gamma^k \leq \epsilon_c$ then terminate (ϵ_r, ϵ_μ and ϵ_c are typically in the order of 10^{-8}).
- If the primal relative residual measure $\vartheta_p^k = \|(r_2, r_3)\|_\infty/\eta_\infty$ is greater than ϵ_r and starts to grow or stalls then the primal problem *may be* infeasible. The algorithm should terminate with a condition flag of *probably* infeasible problem.
- If the dual relative residual measure $\vartheta_d^k = \|r_1\|_\infty/\eta_\infty$ is greater than ϵ_r and starts to grow or stalls then the primal problem *may be* unbounded below. The algorithm should terminate with a condition flag of primal problem *probably* unbounded below.
- If the number of iterations exceeds an upper limit then the algorithm should terminate with a condition flag of unknown but *probably* infeasible.

5 Exploiting MPC Structure

Both ($\mathcal{MPC}_E$) and ($\mathcal{MPC}_I$) can be expressed as a quadratic program. The algorithm given in Section 4 can be applied to either case. In this section we describe how interior-point methods can be geared to exploit the structure of ($\mathcal{MPC}_E$). Gopal and Biegler (1998) provide an authoritative

account of the relevant issues for this style of problem (albeit in the context of Sequential Quadratic Programming (SQP)). The following review is strongly aligned with their discussion.

Optimality conditions for $(\mathcal{MPC}_E)$ can be expressed on an interval-by-interval basis. When $i = 0$, we have the following optimality conditions,

$$\begin{aligned} x_0 - \bar{x} &= 0, \\ Q(x_0 - x_0^r) + \lambda_0 - A^T\lambda_1 &= 0, \\ R(u_0 - u_0^r) - B^T\lambda_1 + L_u^T z_0^u &= 0, \\ L_u u_0 + s_0^u - k_u &= 0, \\ S_0^u z_0^u &= 0, \\ s_0^x,\ s_0^u,\ z_0^u &\succeq 0. \end{aligned}$$

For $i = 1, \ldots, N-1$ we have the following,

$$\begin{aligned} x_i - Ax_{i-1} - Bu_{i-1} &= 0, \\ Q(x_i - x_i^r) + \lambda_i - A^T\lambda_{i+1} + L_x^T z_i^x &= 0, \\ L_x x_i + s_i^x - k_x &= 0, \\ S_i^x z_i^x &= 0, \\ R(u_i - u_i^r) - B^T\lambda_{i+1} + L_u^T z_i^u &= 0, \\ L_u u_i + s_i^u - k_u &= 0, \\ S_i^u z_i^u &= 0, \\ s_i^x,\ z_i^x,\ s_i^u,\ z_i^u &\succeq 0. \end{aligned}$$

Further when $i = N$ the system reduces to,

$$\begin{aligned} x_N - Ax_{N-1} - Bu_{N-1} &= 0, \\ P(x_N - x_N^r) + \lambda_N + L_F^T z_N^x &= 0, \\ L_F x_N + s_N^x - k_F &= 0, \\ S_N^x z_N^x &= 0, \\ s_N^x,\ z_N^x &\succeq 0. \end{aligned}$$

Adopting a similar approach to Section 3.1, we can express a related system where the complementarity conditions $S_i^x z_i^x = 0$ and $S_i^u z_i^u = 0$ are replaced with $S_i^x z_i^x = \mu e$ and $S_i^u z_i^u = \mu e$ respectively. Again we are interested in solving a system in the form of (14). If we order the variables according to $\Delta p_i = [\Delta\lambda_i^T, \Delta x_i^T, (\Delta z_i^x)^T, (\Delta s_i^x)^T, \Delta u_i^T, (\Delta z_i^u)^T, (\Delta s_i^u)^T]^T$ (with obvious exceptions for $i = 0$ and $i = N$), then we the linear system has the following structure,

$$\begin{bmatrix} \ddots & & & & & & & & & \\ & & I & & & & & & & \\ & I & Q & L_x^T & & & & & -A^T & \\ & & L_x & & I & & & & & \\ & & & S_i^x & Z_i^x & & & & & \\ & & & & & R & L_u^T & & -B^T & \\ & & & & & L_u & & I & & \\ & & & & & & S_i^u & Z_i^u & & \\ & & -A & & & -B & & & & \\ & & & & & & & & & \ddots \end{bmatrix} \begin{bmatrix} \vdots \\ \Delta\lambda_i \\ \Delta x_i \\ \Delta z_i^x \\ \Delta s_i^x \\ \Delta u_i \\ \Delta z_i^u \\ \Delta s_i^u \\ \Delta\lambda_{i+1} \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ r_i^1 \\ r_i^2 \\ r_i^3 \\ r_i^4 \\ r_i^5 \\ r_i^6 \\ r_i^7 \\ r_{i+1}^1 \\ \vdots \end{bmatrix} \tag{22}$$

where,

$$\begin{aligned}
r_i^1 &= -(x_i - Ax_{i-1} - Bu_{i-1}) \\
r_i^2 &= -(Q(x_i - x_i^r) + \lambda_i - A^T\lambda_{i+1} + L_x^T z_i^x) \\
r_i^3 &= -(L_x x_i + s_i^x - k_x) \\
r_i^4 &= -(S_i^x z_i^x - \mu e) \\
r_i^5 &= -(R(u_i - u_i^r) - B^T\lambda_{i+1} + L_u^T z_i^u) \\
r_i^6 &= -(L_u u_i + s_i^u - k_u) \\
r_i^7 &= -(S_i^u z_i^u - \mu e)
\end{aligned}$$

Eliminating Δs_i^x and Δs_i^u from (22) results in the following system,

$$\begin{bmatrix}
\ddots & & & & & & & \\
 & & I & & & & & \\
 & I & Q & L_x^T & & & -A^T & \\
 & & L_x & -D_i^x & & & & \\
 & & & & R & L_u^T & -B^T & \\
 & & & & L_u & -D_i^u & & \\
 & & -A & & -B & & & \\
 & & & & & & & \ddots
\end{bmatrix}
\begin{bmatrix} \vdots \\ \Delta\lambda_i \\ \Delta x_i \\ \Delta z_i^x \\ \Delta u_i \\ \Delta z_i^u \\ \Delta\lambda_{i+1} \\ \vdots \end{bmatrix}
=
\begin{bmatrix} \vdots \\ r_i^1 \\ r_i^2 \\ r_i^3 - (Z_i^x)^{-1} r_i^4 \\ r_i^5 \\ r_i^6 - (Z_i^u)^{-1} r_i^7 \\ r_{i+1}^1 \\ \vdots \end{bmatrix}$$

where $D_i^x = (Z_i^x)^{-1} S_i^x$ and $D_i^u = (Z_i^u)^{-1} S_i^u$. The above system is symmetric block tri-diagonal,

$$\begin{bmatrix}
P_0 & V_0^T & & & & \\
V_o & P_1 & V_1^T & & & \\
 & V_1 & P_2 & V_2^T & & \\
 & & V_2^T & \ddots & \ddots & \\
 & & & \ddots & & V_{N-1}^T \\
 & & & & V_{N-1} & P_N
\end{bmatrix}
\begin{bmatrix} \Delta p_0 \\ \Delta p_1 \\ \Delta p_2 \\ \vdots \\ \vdots \\ \Delta p_N \end{bmatrix}
=
\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ \vdots \\ \vdots \\ q_N \end{bmatrix} \qquad (23)$$

Here,

$$P_0 = \begin{bmatrix} & I & & \\ I & Q & & \\ & & R & L_u^T \\ & & L_u & -D_0^u \end{bmatrix}, \quad \Delta p_0 = \begin{bmatrix} \Delta\lambda_0 \\ \Delta x_0 \\ \Delta u_0 \\ \Delta z_0^u \end{bmatrix}, \quad q_0 = \begin{bmatrix} \bar{x} - x_0 \\ r_0^2 \\ r_0^5 \\ r_i^6 - (Z_i^u)^{-1} r_i^7 \end{bmatrix}$$

For $i = 1, \ldots, N-1$. we have,

$$P_i = \begin{bmatrix} & I & & & \\ I & Q & L_x^T & & \\ & L_x & -D_i^x & & \\ & & & R & L_u^T \\ & & & L_u & -D_i^u \end{bmatrix}, \quad \Delta p_i = \begin{bmatrix} \Delta\lambda_i \\ \Delta x_i \\ \Delta z_i^x \\ \Delta u_i \\ \Delta z_i^u \end{bmatrix}, \quad q_i = \begin{bmatrix} r_i^1 \\ r_i^2 \\ r_i^3 - (Z_i^u)^{-1} r_i^4 \\ r_i^5 \\ r_i^6 - (Z_i^u)^{-1} r_i^7 \end{bmatrix},$$

and,

$$P_N = \begin{bmatrix} & I & \\ I & P & L_F^T \\ & L_F & -D_N^x \end{bmatrix}, \quad \Delta p_N = \begin{bmatrix} \Delta\lambda_N \\ \Delta x_N \\ \Delta z_N^x \end{bmatrix}, \quad q_N = \begin{bmatrix} r_N^1 \\ r_N^2 \\ r_i^3 - (Z_i^u)^{-1} r_i^4 \end{bmatrix}$$

Furthermore,

$$V_0 = \begin{bmatrix} 0 & -A & -B & 0 \\ & & & 0 \\ & & & 0 \\ & & & 0 \\ & & & 0 \end{bmatrix}, \quad V_{N-1} = \begin{bmatrix} 0 & -A & 0 & -B & 0 \\ & & & & 0 \\ & & & & 0 \end{bmatrix}$$

and,

$$V_i = \begin{bmatrix} 0 & -A & 0 & -B & 0 \\ & & & & 0 \\ & & & & 0 \\ & & & & 0 \\ & & & & 0 \end{bmatrix}, \qquad \text{for } i = 1, \ldots, N-2.$$

5.1 Riccati Recursion

By further eliminating Δz_i^x and Δz_i^u we can express (23) in a similar form to that considered in (Rao *et al.* 1998), where a Riccati recursion is used to solve this system. As a result of their approach, the number of arithmetic operations grows linearly in the prediction horizon. In particular, the number of operations is given by $O(N(n_x + n_u)^3)$. We proceed as follows,

$$\begin{aligned} \Delta z_i^u &= (D_i^u)^{-1}\left(L_u \Delta u_i + (Z_i^u)^{-1} r_i^7 - r_i^6\right), \quad \text{for } i = 0, \ldots, N-1, \\ \Delta z_i^x &= (D_i^x)^{-1}\left(L_x \Delta x_i + (Z_i^x)^{-1} r_i^4 - r_i^3\right), \quad \text{for } i = 1, \ldots, N-1, \\ \Delta z_N^x &= (D_N^x)^{-1}\left(L_F \Delta x_N + (Z_N^x)^{-1} r_N^4 - r_N^3\right). \end{aligned}$$

Therefore (23) can be expressed as,

$$\begin{bmatrix} \bar{P}_0 & \bar{V}_0^T & & & & \\ \bar{V}_o & \bar{P}_1 & \bar{V}_1^T & & & \\ & \bar{V}_1 & \bar{P}_2 & \bar{V}_2^T & & \\ & & \bar{V}_2^T & \ddots & \ddots & \\ & & & \ddots & & \bar{V}_{N-1}^T \\ & & & & \bar{V}_{N-1} & \bar{P}_N \end{bmatrix} \begin{bmatrix} \Delta\bar{p}_0 \\ \Delta\bar{p}_1 \\ \Delta\bar{p}_2 \\ \vdots \\ \vdots \\ \Delta\bar{p}_N \end{bmatrix} = \begin{bmatrix} \bar{q}_0 \\ \bar{q}_1 \\ \bar{q}_2 \\ \vdots \\ \vdots \\ \bar{q}_N \end{bmatrix} \tag{24}$$

Let $\bar{r}_i^2$ and $\bar{r}_i^3$ be given by,

$$\begin{aligned} \bar{r}_i^2 &= r_i^2 - L_x^T (D_i^x)^{-1}\left((Z_i^x)^{-1} r_i^4 - r_i^3\right) \\ \bar{r}_i^3 &= r_i^5 - L_u^T (D_i^u)^{-1}\left((Z_i^u)^{-1} r_i^7 - r_i^6\right). \end{aligned}$$

For $i = 0, \ldots, N-1$. we have that

$$\bar{P}_i = \begin{bmatrix} & I & \\ I & \bar{Q}_i & \\ & & \bar{R}_i \end{bmatrix}, \qquad \Delta\bar{p}_i = \begin{bmatrix} \Delta\lambda_i \\ \Delta x_i \\ \Delta u_i \end{bmatrix}, \qquad \bar{q}_i = \begin{bmatrix} r_i^1 \\ \bar{r}_i^2 \\ \bar{r}_i^3 \end{bmatrix}$$

In the above $\bar{Q}_i$ and $\bar{R}_i$ are given by,

$$\begin{aligned} \bar{Q}_0 &= Q, \\ \bar{Q}_i &= Q + L_x^T (D_i^x)^{-1} L_x, \quad \text{for } i = 1, \ldots, N-1, \\ \bar{R}_i &= R + L_u^T (D_i^u)^{-1} L_u, \quad \text{for } i = 0, \ldots, N-1 \end{aligned}$$

For $i = N$ we have that

$$\bar{P}_N = \begin{bmatrix} I & \\ & \bar{Q}_N \end{bmatrix}, \qquad \Delta\bar{p}_N = \begin{bmatrix} \Delta\lambda_N \\ \Delta x_N \end{bmatrix}, \qquad \bar{q}_N = \begin{bmatrix} r_N^1 \\ \bar{r}_N^2 \end{bmatrix}$$

In the above $\bar{Q}_N$ is given by

$$\bar{Q}_N = P + L_F^T (D_N^x)^{-1} L_F.$$

Furthermore,

$$\bar{V}_i = \begin{bmatrix} 0 & -A & -B \\ & & 0 \\ & & 0 \end{bmatrix}, \qquad \text{for } i = 0, \ldots, N-2$$

and

$$\bar{V}_{N-1} = \begin{bmatrix} 0 & -A & -B \\ & & 0 \end{bmatrix}$$

System (24) can be solved recursively as follows. Consider the last four block rows of (24),

$$\begin{bmatrix} I & \bar{Q}_{N-1} & & -A^T & \\ & & \bar{R}_{N-1} & -B^T & \\ & -A & -B & & I \\ & & & I & \bar{Q}_N \end{bmatrix} \begin{bmatrix} \Delta\lambda_{N-1} \\ \Delta x_{N-1} \\ \Delta u_{N-1} \\ \Delta\lambda_N \\ \Delta x_N \end{bmatrix} = \begin{bmatrix} \bar{r}^2_{N-1} \\ \bar{r}^3_{N-1} \\ r^1_N \\ \bar{r}^2_N \end{bmatrix} \tag{25}$$

We can eliminate Δx_N and $\Delta\lambda_N$ which results in,

$$\begin{bmatrix} I & \bar{Q}_{N-1} + A^T\bar{Q}_N A & A^T\bar{Q}_N B \\ & B^T\bar{Q}_N A & \bar{R}_{N-1} + B^T\bar{Q}_N B \end{bmatrix} \begin{bmatrix} \Delta\lambda_{N-1} \\ \Delta x_{N-1} \\ \Delta u_{N-1} \end{bmatrix} = \begin{bmatrix} \bar{r}^2_{N-1} + A^T\bar{r}^2_N - A^T\bar{Q}_N r^1_N \\ \bar{r}^3_{N-1} + B^T\bar{r}^2_N - B^T\bar{Q}_N r^1_N \end{bmatrix}$$

Lastly, by eliminating Δu_{N-1} we get the following relation,

$$\Delta\lambda_{N-1} + W_{N-1}\Delta x_{N-1} = w_{N-1}$$

where

$$W_{N-1} = \bar{Q}_{N-1} + A^T\bar{Q}_N A - A^T\bar{Q}_N B\left(\bar{R}_{N-1} + B^T\bar{Q}_N B\right)^{-1} B^T\bar{Q}_N A, \tag{26}$$

$$w_{N-1} = \bar{r}^2_{N-1} + A^T\bar{r}^2_N - A^T\bar{Q}_N r^1_N \tag{27}$$

$$-A^T\bar{Q}_N B\left(\bar{R}_{N-1} + B^T\bar{Q}_N B\right)^{-1}\left(\bar{r}^3_{N-1} + B^T\bar{r}^2_N - B^T\bar{Q}_N r^1_N\right). \tag{28}$$

We may now obtain a similar relation for $\Delta\lambda_{N-2}$ and Δx_{N-2}. Indeed, using equations (26) and (28) we have,

$$\begin{bmatrix} I & \bar{Q}_{N-2} & & -A^T & \\ & & \bar{R}_{N-2} & -B^T & \\ & -A & -B & & I \\ & & & I & W_{N-1} \end{bmatrix} \begin{bmatrix} \Delta\lambda_{N-2} \\ \Delta x_{N-2} \\ \Delta u_{N-2} \\ \Delta\lambda_{N-1} \\ \Delta x_{N-1} \end{bmatrix} = \begin{bmatrix} \bar{r}^2_{N-2} \\ \bar{r}^3_{N-2} \\ r^1_{N-1} \\ w_{N-1} \end{bmatrix}$$

This is in the same form as (25). More generally, if we define W_N and w_N as,

$$W_N = \bar{Q}_N,$$
$$w_N = \bar{r}^2_N,$$

then W_{i-1} and w_{i-1} may be obtained recursively via,

$$W_{i-1} = \bar{Q}_{i-1} + A^T W_i A - A^T W_i B\left(\bar{R}_{i-1} + B^T W_i B\right)^{-1} B^T W_i A,$$
$$w_{i-1} = \bar{r}^2_{i-1} + A^T w_i - A^T W_i r^1_i$$
$$-A^T W_i B\left(\bar{R}_{i-1} + B^T W_i B\right)^{-1}\left(\bar{r}^3_{i-1} + B^T w_i - B^T W_i r^1_i\right).$$

After obtaining W_i and w_i for $i = N, \ldots, 1$ we can solve the following system,

$$\begin{bmatrix} & I & & & \\ I & Q & & -A^T & \\ & & \bar{R}_0 & -B^T & \\ & -A & -B & & I \\ & & & I & W_1 \end{bmatrix} \begin{bmatrix} \Delta\lambda_0 \\ \Delta x_0 \\ \Delta u_0 \\ \Delta\lambda_1 \\ \Delta x_1 \end{bmatrix} = \begin{bmatrix} r^1_0 \\ \bar{r}^2_0 \\ \bar{r}^3_0 \\ r^1_1 \\ w_1 \end{bmatrix}$$

Now for $i = 1, \ldots, N$ we can recover Δu_i, Δx_i and $\Delta \lambda_i$ via,

$$\begin{aligned}
\Delta u_i &= \left(\bar{R}_{i-1} + B^T W_{i+1} B\right)^{-1} \left[\bar{r}^3_{i-1} + B^T w_{i+1} - B^T W_{i+1} r^1_i - B^T W_{i+1} A \Delta x_i\right], \\
\Delta x_{i+1} &= A\Delta x_i + B\Delta u_i + r^1_{i+1}, \\
\Delta \lambda_{i+1} &= w_{i+1} - W_{i+1}\Delta x_{i+1}.
\end{aligned}$$

6 Simulation Example

In this section we look at computational aspects of both the implicit and explicit formulations for a specific plant model with different prediction horizons. The simulations were performed in Matlab versions 5.3 and 6.1 running on an Intel Pentium IV 2GHz machine with 512M of memory.

Consider the following linear state-space model

$$\begin{aligned}
x_{t+1} &= Ax_t + Bu_t, \\
y_t &= Cx_t.
\end{aligned}$$

In the above, the matrices A,B and C were chosen randomly and are given in Appendix A. The number of states n_x is equal to 10 and the number of inputs n_u and outputs n_y are both equal to 2. The system has one unstable pole.

The explicit and implicit MPC problem formulations were considered separately. Further, two interior-point solvers were used. The first is the OOQP package by Gertz and Wright (2001) which uses the MA47 solver from the Harwell library (see `http://hsl.rl.ac.uk/`) to solve the linear system in the form of (18). The second is the authors' Matlab implementation of Algorithm 1 using the `luinc` Matlab function to factor the left hand matrix in (15) into lower and upper triangular components.

CPU execution time and floating point operations (flops) were used to measure algorithm efficiency. Both these measures are notoriously unreliable. However, we only compare like with like. In particular, for the simulations using the OOQP package, CPU time was used to compare explicit and implicit approaches. For the simulations using the authors' Matlab solver, flops were used to compare explicit and implicit approaches.

Figures 1 and 2 show results using the OOQP package. Execution time was determined using the `tic` and `toc` Matlab functions. Figures 3 and 4 shows results for the authors' Matlab implementation. Flop counts were determined using the `flops` Matlab function.

The aim here is not to compare the interior-point solvers, but to observe that in both cases the explicit formulation appears to have linear growth in the prediction horizon whilst the implicit formulation appears to have cubic growth in the prediction horizon. Note that the MPC structure has not been directly exploited when solving the linear sub-system (15); in particular, the block-tridiagonal structure is not exploited.

7 Discussion and Conclusion

Linear MPC involves (at least) one Quadratic Programming problem at each time interval. In the literature there appear to be two prominent formulations for this QP: the explicit form as per

Section 2.1 and the implicit form as per Section 2.2. Typically, the latter tends to be more dense than the former.

If we adopt an interior-point strategy (which may or may not be the most suitable approach) then we are asked to solve a linear system of equations (at least once per iteration),

$$Ax = b. \tag{29}$$

Typically, A will become increasingly ill-conditioned towards later stages of the algorithm. Therefore, care must be taken when solving (29) in this case.

We have seen various structures for the co-efficient matrix including:

1. Sparse and indefinite - see (15).
2. Sparse, indefinite and block tridiagonal - (22).
3. Sparse, symmetric and indefinite - see (18) and (19).
4. Sparse, symmetric, indefinite and block tridiagonal - see (23) and (24).
5. Dense, symmetric and positive definite - see (21).

Applicable factorisation methods will depend on the chosen structure. In any event, sparsity should be exploited if possible.

Assuming that an interior-point approach is used, it may be difficult (*ceteris paribus*) to prescribe one particular structure that performs well for an arbitrary MPC problem. For example, if the system has many states and only a few inputs, then even for moderate prediction horizons it is reasonable to expect that the implicit formulation will perform better. As a converse example, if the prediction horizon is required to be large then it is reasonable to expect that an explicit formulation will perform better - as the Riccati-recursion method and the simulation example testifies.

Although the Riccati recursion approach is very appealing, direct methods for solving the linear subproblems found in items 1-4 should produce acceptable results (even for large horizons). Indeed, packages like MA27, MA47 and MA57 from the Harwell library (`http://hsl.rl.ac.uk/`) can offer remarkable computational savings for the symmetric cases in items 3 and 4. These packages use a multifrontal technique which combines frontal methods (Duff *et al.* 1986) with pivot ordering routines to preserve sparsity. The MA57 (Duff 2002) package reports good results for practical optimisation problems. Due to the adaptive nature of these algorithms, it is difficult to specify their computational cost *apriori*. However, preliminary trials on a limited number of examples show near linear growth in the prediction horizon – even without exploiting the MPC structure as discussed in section 5.

It is the authors' opinion that achieving optimal performance for each MPC optimisation problem requires individual consideration. *Inter-alia*, the linear dynamical system, sparsity structure, system dimensions, presence of state and/or input constraints, weighting matrices and the prediction horizon can all affect the performance. Furthermore, with recent advances in both optimisation science and computational speed, the class of control problems for which MPC is tractable has increased and, via simple extrapolation, is likely to continue to do so.

References

Albuquerque, J., V. Gopal, G. Staus, L. T. Biegler and B. E. Ydstie (1997). Interior point SQP strategies for structured process optimization problems. *Computers & Chemical Engineering* **2**, S853–S859.

Barnard, R. D. (1976). Continuous-time implementation of optimal-aim controls. *IEEE Transactions on Automatic Control* **21**, 375.

Bartlett, R. A., A. Wachter and L. T. Biegler (2000). Active set vs. interior point strategies for model predictive control. In: *Proceedings of the American Control Conference*. Chicago, Illinois. pp. 4229–4233.

Bemporad, A., M. Morari and V. Dua E. N. Pistikopoulos (2002). The explicit linear quadratic regulator for constrained systems. *Automatica* **38**(1), 3–20.

Bitmead, R. R., M. Gevers and V. Wertz (1990). *Adaptive Optimal Control -The Thinking Man's GPC*. Prentice Hall, Inc.. Englewood Cliffs, New Jersey.

Boyd, S. and L. Vandenberghe (2002). Convex optimization. To appear. `http://www.stanford.edu/~boyd/cvxbook.html`.

Brézis, H. (1973). *Opérateurs maximaux monotones et sémigroups de contractions dans les espaces de Hilbert*. Mathematical Studies 5. North Holland.

Chen, H. and F. Allgöwer (1998). A quasi-infinite nonlinear model predictive control scheme with guaranteed stability. *Automatica* **34**(10), 1205–1217.

Clarke, D. W., C. Mohtadi and P. S. Tuffs (1987). Generalized predictive control, parts 1 and 2. *Automatica* **23**(2), 137–148.

Duff, I. S. (2002). MA57 - A new code for the solution of sparse symmetric definite and indefinite systems. Technical report. Report RAL-TR-2002-024, Computational Science and Engineering Department, Atlas Centre, Rutherford Appleton Laboratory.

Duff, I. S., A. M. Erisman and J. K. Reid (1986). *Direct Methods for Sparse Matrices*. Oxford University Press. Oxford.

Dunn, J. C. and D. P. Bertsekas (1989). Efficient dynamic programming implementations of Newton's method for unconstrained optimal control problems. *Journal of Optimization Theory and Applications* **63**, 23–38.

Fletcher, R. (1987). *Practical Methods of Optimization*. John Wiley & Sons Inc.. New York.

Garcia, C. E., D. M. Prett and M. Morari (1989). Model predictive control: Theory and practice - A survey. *Automatica* **25**(3), 335–348.

Gertz, E. M. and S. J. Wright (2001). Object-oriented software for quadratic programming. Technical report. Optimization Technical Report 01-02, Computer Sciences Department, University of Wisconsin-Madison. `http://www.cs.wisc.edu/~swright/papers/`.

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

Golub, G. H. and C. F. Van Loan (1996). *Matrix Computations, Third Edition*. The Johns Hopkins University Press. Baltimore, Maryland.

Gopal, V. and L. T. Biegler (1998). Large scale inequality constrained optimization and control. *IEEE Control Systems Magazine* **18**(6), 59–68.

Hansson, A. (2000). A primal-dual interior-point method for robust optimal control of linear discrete-time systems. *IEEE Transactions on Automatic Control* **45**(9), 1639–1655.

Heath, W. P. and A. G. Wills (2002). Recentred barriers for model predictive control with state constraints. *Non-linear Predictive Control Workshop at the University of Oxford*.

Kassmann, D. E., T. A. Badgwell and R. B. Hawkins (2000). Robust steady-state target calculation for model predictive control. *AIChE Journal* **46**(5), 1007–1024.

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

Kouvaritakis, B., M. Cannon and J. A. Rossiter (2002). Who needs qp for linear mpc anyway?. *Automatica* **38**(5), 879–884.

Lieu, B. T. and P. Huard (1966). La methode des centres dans un espace topologique. *Numerische Mathematik* **8**, 56–67.

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

Maciejowski, J. M. (2002). *Predictive Control with Constraints*. Pearson Education Limited. Harlow, Essex.

Mangasarian, O. L. (1969). *Nonlinear Programming*. McGraw-Hill Book Company. New York.

Mayne, D. Q., J. B. Rawlings, C. V. Rao and P. O. M. Scokaert (2000). Constrained model predictive control: Stability and optimality. *Automatica* **36**, 789–814.

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

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

Nesterov, Y. E. and M. J. Todd (1997). Self-scaled cones and interior–point methods in nonlinear programming. *Mathematics of Operations Research* **22**, 1–42.

Nesterov, Y. E. and M. J. Todd (1998). Primal–dual interior–point methods for self–scaled cones. *SIAM Journal on Optimization* **8**, 324–364.

Polak, E., T. H. Yang and D. Q. Mayne (1990). A method of centers based on barrier functions for solving optimal control problems with continuum state and control constraints. In: *Proceedings of the 29th Conference on Decision and Control*. Vol. 4. Honolulu, Hawaii. pp. 2327–2332.

Qin, S. Joe and T. A. Badgwell (1997). An overview of industrial model predictive control technology. *AIChE Symposium Series, 5th International Symposium on Chemical Process Control* **93**, 232–256.

Rao, C. V., J. B. Rawlings and D. Q. Mayne (2003). Constrained state estimation for nonlinear discrete-time systems: Stability and moving horizon approximations. *IEEE Transactions on Automatic Control* **48**(2), 246–258.

Rao, C. V., S. J. Wright and J. B. Rawlings (1998). Application of interior point methods to model predictive control. *Journal of Optimization Theory and Applications* **99**(3), 723–757.

Rossiter, J. A. and B. Kouvaritakis (1993). Constrained stable generalised predictive control. *Control Theory and Applications, IEE Proceedings D [see also IEE Proceedings-Control Theory and Applications]* **140**(4), 243–254.

Scokaert, P. O. M. and J. B. Rawlings (1999). Feasibility issues in linear model predictive control. *AiChE Journal* **45**(8), 1649–1659.

Seron, M. M., G. C. Goodwin and J. A. De Dona (2002). Finitely parameterised implementation of receding horizon control for constrained linear systems. In: *Proceeding of the 2002 American Control Conference*. Vol. 6. pp. 4481–4486.

Soroush, M. and S. Valluri (1999). Optimal directionality compensation in processes with input saturation non-linearities. *International Journal of Control* **72**(17), 1555–1564.

Syaichu-Rohman, A., R. H. Middleton and M. M. Seron (2003). A multivariable nonlinear algebraic loop as a qp with application to mpc. In: *European Control Conference*. Accepted for publication.

Tenny, M. J., S. J. Wright and J. B. Rawlings (2002). A feasible trust-region sequential quadratic programming algorithm. Technical report. Optimization Technical Report 02-05, Computer Sciences Department, University of Wisconsin-Madison. `http://www.optimization-online.org/DB_HTML/2002/08/520.html`.

Tøndel, P., T. A. Johansen and A. Bemporad (2003). An algorithm for multi-parametric quadratic programming and explicit MPC solutions. *Austomatica* **39**(3), 489–497.

Vandenberghe, L., S. Boyd and M. Nouralishahi (2002). Robust linear programming and optimal control. In: *15th Triennial World Congress of the International Federation of Automatic Control*. Barcelona.

Vanderbei, R. J. (1999). LOQO: An interior point code for quadratic programming. *Optimization Methods and Software* **11**, 451–484.

Wills, A. G. (2003). Barrier Function Based Model Predictive Control. PhD thesis. School of Electrical Engineering and Computer Science. Submitted March 2003.

Wills, A. G. and W. P. Heath (2002*a*). A recentred barrier for constrained receding horizon control. In: *Proceedings of the 2002 American Control Conference*. Vol. 5. Anchorage, Alaska. pp. 4177–4182.

Wills, A. G. and W. P. Heath (2002*b*). Using a modified predictor-corrector algorithm for model predictive control. In: *Proceedings of the 15th IFAC World Congress on Automatic Control*. Barcelona, Spain.

Wills, A. G. and W. P. Heath (2003). An exterior/interior-point approach to infeasibility in model predictive control. Technical report. School of Electrical Engineering and Computer Science, University of Newcastle Australia. Report number EE03014.

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

Wright, S. J. (1995). Stability of linear equations solvers in interior-point methods. *SIAM Journal of Matrix Analysis and Applications* **16**, 1287–1307.

Wright, S. J. (1997*a*). Applying new optimization algorithms to model predictive control. *Chemical Process Control-V, CACHE, AIChE Symposium Series* **93**(316), 147–155.

Wright, S. J. (1997*b*). Stability of augmented system factorizations in interior-point methods. *SIAM Journal of Matrix Analysis and Applications* **18**, 191–222.

Wright, S. J. (1999). Modified Cholesky factorizations in interior-point algorithms for linear programming. *SIAM Journal on Optimization* **9**, 1159–1191.

Ye, Y. (1997). *Interior Point Algorithms - Theory and Analysis*. John Wiley & Sons Inc.. New York.

Zafiriou, E. (1990). Robust model predictive control of processes with hard constraints. *Computers & Chemical Engineering* **14**(4–5), 359–371.

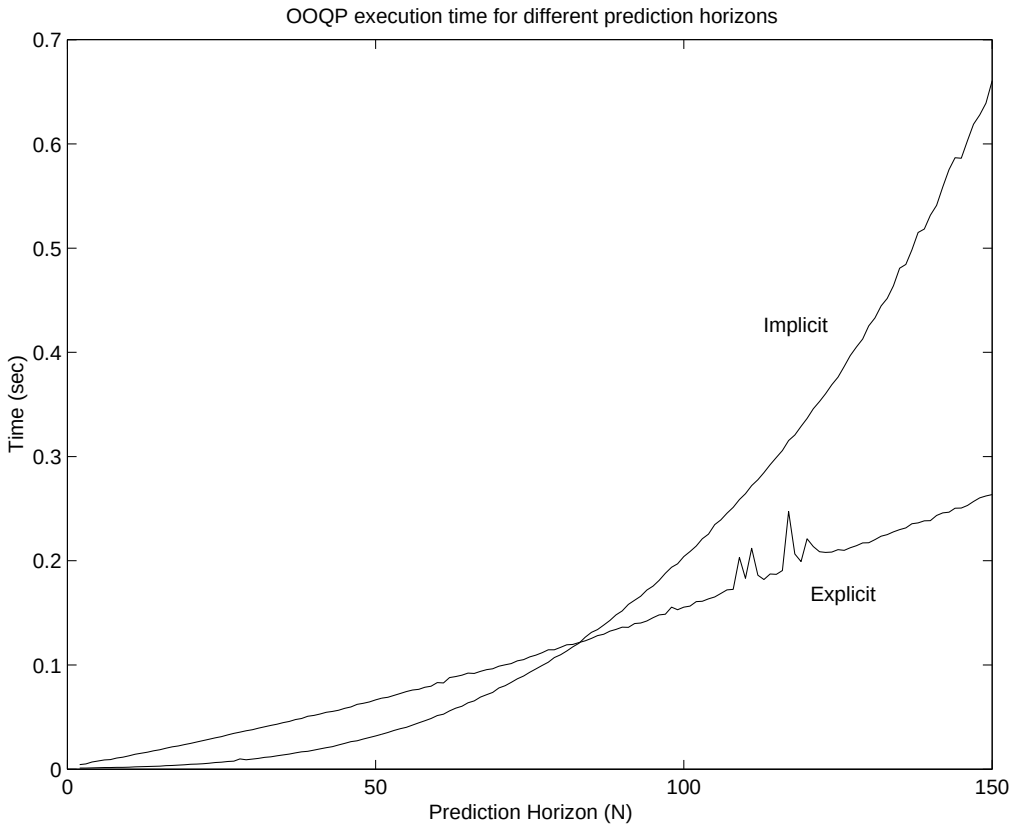

Figure 1: CPU execution time for simulations using OOQP. Note that both scales are linear. It seems as though execution time increases rapidly for the implicit formulation with large prediction horizons. Meanwhile, execution time for the explicit formulation appears to be linear.

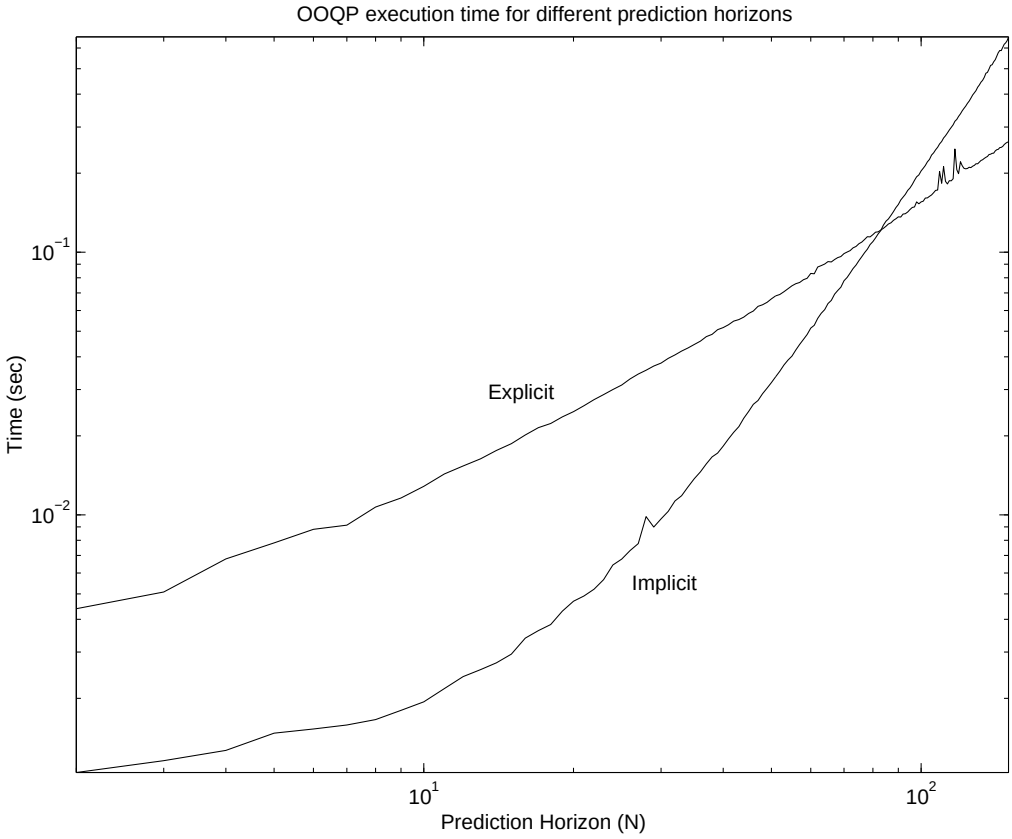

Figure 2: CPU execution time for simulations using OOQP. Note that both scales are logarithmic with base 10. Also note that the implicit curve approaches a straight line with gradient 3 for large values of N, thus indicating a cubic relationship between computation time and prediction horizon. Meanwhile, the explicit curve approaches a straight line with gradient 1 for large N, indicating a linear relationship with prediction horizon.

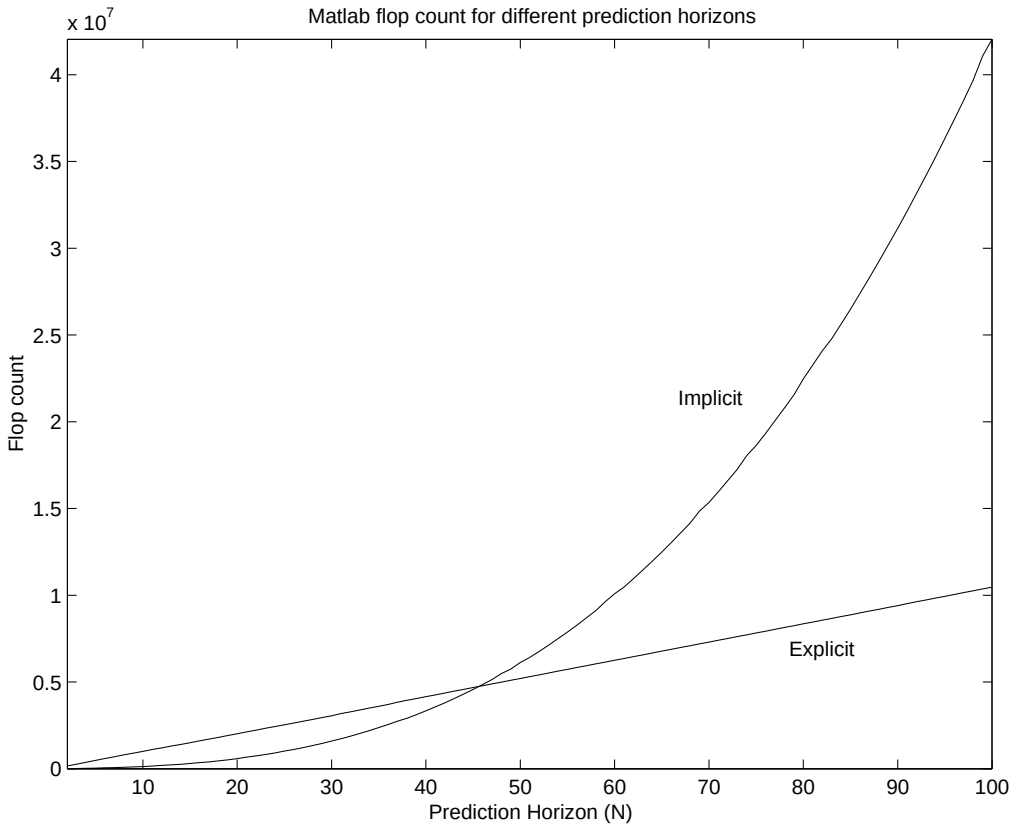

Figure 3: Flop counts for the authors' code. Note that both scales are linear. Again, it seems as though execution time increases rapidly for the implicit formulation with large prediction horizon. Meanwhile, execution time for the explicit formulation appears to be linear.

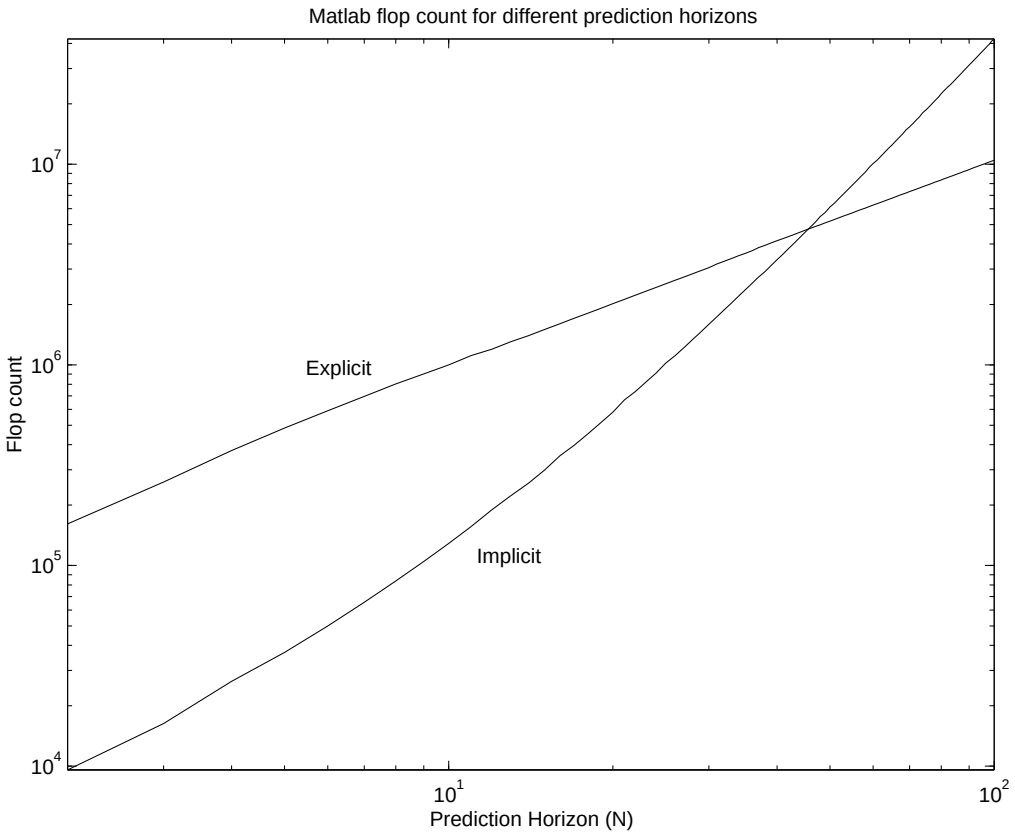

Figure 4: Flop counts for the authors' code. Note that both scales are logarithmic with base 10. Also note that the implicit curve approaches a straight line with gradient 3 for large values of N, thus indicating a cubic relationship between computation time and prediction horizon. Meanwhile, the explicit curve approaches a straight line with gradient 1 for large N, indicating a linear relationship with prediction horizon.

A Simulation Plant Model

The simulation plant model used in Section 6 is given by the discrete state-space model

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t, \\ y_t &= Cx_t. \end{aligned}$$

The matrices A, B and C are given by

$$A = \begin{bmatrix} A_1 & A_2 \end{bmatrix},$$

with

$$A_1 = \begin{bmatrix} -0.0757 & 0.0707 & -0.0758 & -0.2869 & 0.0015 \\ -0.0028 & -0.1624 & -0.1282 & -0.3003 & -0.0292 \\ -0.0242 & 0.0008 & -0.3699 & 0.5390 & -0.1968 \\ 0.1322 & 0.2438 & 0.1325 & 0.1121 & 0.0512 \\ 0.2739 & 0.2921 & -0.2940 & -0.0136 & -0.3752 \\ 0.3680 & 0.0344 & -0.3855 & -0.1598 & -0.6043 \\ 0.2748 & 0.4128 & -0.0159 & 0.0039 & 0.2141 \\ 0.4141 & -0.4212 & 0.1400 & 0.3686 & -0.2818 \\ 0.0373 & -0.1722 & -0.0037 & 0.3825 & -0.0804 \\ -0.2583 & 0.3087 & -0.0983 & -0.3360 & -0.0574 \end{bmatrix},$$

and

$$A_2 = \begin{bmatrix} -0.2646 & -0.6016 & -0.1465 & -0.0444 & -0.4932 \\ 0.1320 & 0.0746 & -0.1074 & -0.1470 & -0.3455 \\ 0.4283 & 0.2133 & 0.0538 & -0.3077 & 0.4440 \\ 0.4604 & 0.0779 & -0.1259 & -0.0502 & 0.5208 \\ -0.4473 & 0.0951 & -0.0026 & -0.1630 & -0.0545 \\ 0.1425 & 0.3621 & -0.4882 & 0.2946 & 0.2315 \\ 0.3140 & -0.1437 & -0.2575 & -0.4190 & -0.2806 \\ 0.3464 & -0.0608 & 0.0633 & 0.1676 & 0.6007 \\ 0.1750 & 0.0995 & 0.2716 & 0.0716 & -0.3700 \\ 0.3221 & -0.1388 & 0.1236 & -0.2490 & 0.1371 \end{bmatrix},$$

$$B = \begin{bmatrix} -0.4219 & -1.2467 \\ -0.5673 & 0.9652 \\ 0.4219 & -1.1380 \\ 0.1378 & 0.3378 \\ -0.2565 & -0.1151 \\ 0.0800 & -1.2943 \\ -1.3375 & 1.0550 \\ 0.4100 & 0.2415 \\ -0.4270 & 1.0341 \\ 1.0498 & -0.5008 \end{bmatrix}, \qquad C^T = \begin{bmatrix} 1.5758 & 0.8104 \\ 2.7527 & 0.2012 \\ 0.6896 & 0.2228 \\ -0.8408 & -1.9292 \\ -0.3912 & 1.8396 \\ 0.0718 & -0.3245 \\ -1.8085 & -1.5548 \\ 0.8649 & 0.3693 \\ -0.9647 & 0.8245 \\ -1.6555 & -0.8378 \end{bmatrix}.$$