Dev. Chem. Eng. Mineral Process. 12(5/6), pp. 573-614, 2004.

A Tutorial on Model Predictive Control: Using a Linear Velocity-Form Model

Liuping Wang*

School of Electrical and Computer Engineering, RMIT University, Melbourne, Victoria 3000, Australia

Model Predictive Control (MPC) has a long history in the field of control engineering. It is one of the few areas that has received on-going interest from researchers in both industry and universities. It has been recognised that there are three major branches of MPC algorithms consisting of step-response model based design: Dynamic Matrix Control (DMC); transfer function model based design: Generalised Predictive Control (GPC); and a general state space model based design. The DMC and GPC algorithms can also be caste in the state space framework. Along the general lines of state space methods, there are two mainstreams: one solves for the optimal control signal while the other solves for the increment of the optimal control signal. The latter can be implemented in a velocity form analogous to the implementation of a PID controller on an industrial plant. Motivated by this advantage, and that integral action is naturally embedded in the algorithm, this tutorial paper focuses on an introduction to Model Predictive Control based on the state space approach using a linear velocity-form model.

Keywords: *Discrete time systems; model predictive control; least squares solution; quadratic programming.*

1. Introduction

Model Predictive Control (MPC) refers to a class of algorithms that compute the trajectory of manipulated variable adjustment to optimise the future behaviour of a plant. This class of algorithms has been developed since the 1970's and has received wide application in the process industries. There are several major reviews published for MPC which clarify the economic benefits of this class of control algorithms when applied to the process industries (Garcia et al., 1989; Richalet, 1993; Qin and Badgwell, 1996; Morari and Lee, 1999). Mayne et al. (2000) gives an extensive theoretical review of model predictive control. There are also several excellent tutorial papers published in the area of model predictive control (Ricker, 1991; Shah, 1995; Rawlings, 2000; Allgower, 1999), in addition to several recent books in this area (Camacho and Bordons, 1999; Maciejowski, 2002).

There are three general approaches to model predictive control design. Each approach uses a unique model structure. In earlier formulations of model predictive control, Finite Impulse Response (FIR) models and Step Response models were

* *Author for correspondence (liuping.wang@rmit.edu.au).*

favoured. FIR model/step response model based design algorithms include Dynamic Matrix Control (DMC) (Cutler and Ramaker, 1979) and the quadratic DMC formulation of Garcia and Morshedi (1986). The FIR type of models are appealing to process engineers because the model structure gives a transparent description of process time delay, response time and gain. However, they are limited to stable plants and often require large model orders. This model structure typically requires 30 to 60 impulse response coefficients depending on the process dynamics and choice of sampling intervals. Transfer function models give a more parsimonious description of process dynamics and are applicable to both stable and unstable plants. Representatives of transfer function model based predictive control include the predictive control algorithm of Peterka (Peterka, 1984) and the Generalized Predictive Control algorithm of Clarke and colleagues (Clarke et al. 1987 a,b). The transfer function model based predictive control is often considered to be less effective in handling multivariable plants. Recent years have seen the growing popularity of predictive control design using state space design methods. The early history of predictive control design, predominantly in the electrical engineering field, focused on receding horizon control, including the work of Kleinman (Kleinman, 1970) and Thomas (Thomas, 1975), who showed that a state feedback stabilizing controller can be found by solving a finite horizon optimization problem. The computation was off-line and did not involve on-line optimization. The extension of these early results was provided by Kwon and his colleagues (Kwon and Pearson, 1977 and Kwon, Bruckstein and Kailath, 1983) where, for each time instant t, the state feedback controller was determined by solving a matrix Riccati differential equation over the prediction interval. This procedure required on-line computation. A receding horizon control strategy for nonlinear systems was presented in Mayne and Michalska (1990). In the chemical engineering field, the SMOC algorithm developed at Shell France represented the early development of model predictive control using state space design methods (Marquis and Broustail, 1988). Recent papers on model predictive control using state space models by chemical engineers include the tutorial paper by Ricker (1991), the papers by Muske and Rawlings (1993a, b), the paper by Scokaert and Rawlings (1998). Morari and Lee (1991) showed that the FIR model structure can be caste in a state space formulation. GPC has also been analyzed using the framework of state space methods (Bitmead et al. 1990).

The literature mentioned above, by no means gives an adequate citation of all the papers in the vast field of model predictive control. A more detailed account on the history of model predictive control from a process engineer's point of view is referred to Qin and Badgwell (1996), Morari and Lee (1999) and Allgower et al. (1999). Nevertheless, there is a rich source of literature to support the design and analysis of model predictive control using the state space approach, including dealing with stability issues and optimization problems. Taking account of these factors, this paper chooses to present the topic of model predictive control design based on a state space approach. Among the state space approaches, there are two mainstreams in the formulation of the MPC problem: one approach is to solve the control variable (Rawlings, 2000) while the other is to target the difference of the control variable (Ricker, 1991). However, the latter formulation also called model predictive control algorithm using velocity form model, leads to natural embedding of integral action and a simplified form for implementation of the model predictive control on a real plant.

There are three major aspects of model predictive control which make the design methodology attractive to both academics and engineers. The first aspect is the design formulation which has a complete multivariable system feature, specifically the performance parameters of the multivariable control system are related to the engineering aspects of the process, hence they can be understood and 'tuned' by engineers. This aspect is often overlooked by researchers. Nevertheless, the cost of commission and maintenance is a very important issue for a company to decide whether or not to use advanced control. The second aspect is the ability to handle both 'soft' constraints and hard constraints in the design. This is attractive to industry where tight profit margins and limits on the operation of the process are inevitably present. The third aspect is the ability to perform on-line optimization.

It is useful to have the background of the readers in mind when preparing a tutorial paper. The tutorial paper by Shah (1995) gave a basic introduction to constrained control using DMC as a vehicle, while Rawlings (2000) overviewed many important issues in model predictive control using the state space approach that targets the control variables. Perhaps, this tutorial paper aims the readers somewhere in-between Shah (1995) and Rawlings (2000). The focus of the paper is on model predictive control algorithms using a velocity form model. The main reason for such a decision is due to consideration of the simplified implementation procedure when this particular form of the algorithm is used, which hopefully will encourage the readers to try out the MPC algorithms on real plants. To understand the material presented in the paper, it is helpful for the readers to have some basic knowledge of state space methods. The text books for introduction to state space methods include Franklin et al. (1994), Kailath (1980) and Goodwin et al. (2000). It is the intention of this author to present, in a simplified and compact way, the basic ideas and concepts in model predictive control design, as well as some practical issues in the implementation of the algorithm. The tutorial paper was originally based on a set of lecturer's notes prepared for an advanced control course blended with the author's practical experience in implementation of model predictive control on several real plants.

The structure of the paper is as follows. The tutorial begins in Section 2 with observer based receding horizon control, in which we show that without using 'hard' constraints the MPC system is essentially a state estimate feedback control system. In section 3, we show how to incorporate 'hard' constraints into model predictive control. We address the solution in the form of quadratic programming. Section 4 discusses stability issues in model predictive control using terminal constraints. Section 5 addresses the numerical problem that exists with this class of the algorithm when the prediction horizon is large and proposes the strategies that might be used to overcome the numerical problem. Section 6 discusses a recent work in model predictive control design using a set of basis functions to reduce number of parameters in the optimization procedure. Section 7 presents two simulation examples. In the second simulation example, the tutorial takes the readers through an application of model predictive control on food extruder model using Matlab programming platform.

2. Model Predictive Control Design Using A Velocity Form Model

2.1 State space models

The general design criterion of model predictive control is to compute a trajectory of future manipulated variable u to optimize the future behaviour of the plant output y. The optimization is performed within a limited time window. To this end, we introduce the following terminology.

- Moving horizon window: the time dependent window from an arbitrary time k_i to $k_i + N_p$. The length of the window N_p (measured by the number of samples) remains constant.
- Prediction horizon: equals the length of the moving horizon window N_p and dictates how 'far' we wish the future to be predicted for.
- Control horizon: denoted by N_c and measured by number of samples, dictates how 'long' we wish the control signal to be active for before it reaches a steady state.
- Receding horizon control: although the optimal trajectory of future control signal is completely captured within the moving horizon window, the actual control input to the plant only takes the first sample of the control signal while neglecting the rest of the trajectory.

The model to be used in the control system design is taken to be a state space model. Specifically, we assume the plant to be controlled has p ($p \geq 1$) inputs and q outputs and is described by a discrete state space model of the form:

$$\begin{aligned} x_m(k+1) &= A_m x_m(k) + B_m u(k) + \xi(k) \\ y(k) &= C_m x_m(k) + \eta(k) \end{aligned} \tag{1}$$

where

- u is the manipulated variable (dimension of p);
- y is the process output (dimension of q);
- x_m is the state variable (dimension of m);
- ξ is the disturbance;
- η is the measurement noise.

A_m, B_m and C_m are matrices with dimension $m \times m$, $m \times p$ and $q \times m$ respectively. We assume that the model is stabilizable and detectable throughout the paper (i.e. all of its uncontrollable poles and all of its unobservable poles are stable). The state space model (1) can be obtained from either conversion of a transfer function model (for example, Kailath 1980) or direct identification of a state space model using subspace methods (Viberg, 1995).

The design of model predictive control using a velocity form model is based on optimization of future incremental control variable Δu. To obtain Δu as the manipulated variable for the future plant output, we augment the original state space model (1) as follows. Let $\Delta x_m(k) = x_m(k) - x_m(k-1)$ and $\bar{y}(k) = C_m x_m(k)$. Choosing a new state variable vector $x(k) = [\Delta x_m(k)\ \ \bar{y}(k)]^T$, we have:

$$
\begin{aligned}
\begin{bmatrix} \Delta x_m(k+1) \\ \bar{y}(k+1) \end{bmatrix} &= \begin{bmatrix} A_m & 0_1 \\ C_m A_m & I_{q\times q} \end{bmatrix} \begin{bmatrix} \Delta x_m(k) \\ \bar{y}(k) \end{bmatrix} + \begin{bmatrix} B_m \\ C_m B_m \end{bmatrix} \Delta u(k) \\
&\quad + \begin{bmatrix} \xi(k) - \xi(k-1) \\ 0_2 \end{bmatrix} \\
y(k) &= \begin{bmatrix} 0_m & I_{q\times q} \end{bmatrix} \begin{bmatrix} \Delta x_m(k) \\ \bar{y}(k) \end{bmatrix} + \eta(k)
\end{aligned}
\tag{2}
$$

where 0_1, 0_m and 0_2 are zero matrices with dimensions $m \times q$, $q \times m$ and $q \times 1$ respectively, and $I_{q\times q}$ is a unit matrix with dimension q. In the formulation of (2), we have assumed that the disturbance $\xi(k)$ is a random walk type of disturbance, more specifically, $\Delta\xi(k) = \xi(k) - \xi(k-1)$ is zero mean white noise. For notational simplicity, we denote (2) by

$$
\begin{aligned}
x(k+1) &= Ax(k) + B\Delta u(k) + \bar{\xi}(k) \\
y(k) &= Cx(k) + \eta(k)
\end{aligned}
\tag{3}
$$

where A, B and C are matrices corresponding to the forms given in (2). In the sequel, the dimensionality of the augmented state space equation is taken to be n, and (3) is referred as the composite design model.

There are two points that are worth mentioning here. The first is related to the eigenvalues of the composite design model. Note that the characteristic equation of the composite model is

$$
\begin{aligned}
\rho(\lambda) &= det \begin{bmatrix} \lambda I - A_m & 0_1 \\ -C_m A_m & (\lambda - 1) I_{q\times q} \end{bmatrix} \\
&= (\lambda - 1)^q det(\lambda I - A_m)
\end{aligned}
\tag{4}
$$

Hence, the eigenvalues of the composite model are the union of the eigenvalues of the plant model and the q eigenvalues being on the unit circle of the complex plane. The second point is that it can be verified that the z-transfer function of the composite model is

$$
C(zI - A)^{-1} B = \frac{z}{z-1} C_m (zI - A_m)^{-1} B_m \tag{5}
$$

Hence, the composite model is detectable and stabilizable if the plant model is detectable and stabilizable, and has no transmission zeros on the unit circle.

2.2 Prediction of Future States and Output

We begin by assuming that at the sampling instant k_i, $k_i > 0$, the state variable vector $x(k_i)$ is available through measurement. The more general situation where the state is not directly measured will be discussed later. The future control trajectory is captured by $\Delta u(k_i), \Delta u(k_i+1), \ldots, \Delta u(k_i+N_c-1)$. Then the prediction of the future state variables, $x(k_i+1/k_i), x(k_i+2/k_i), \ldots, x(k_i+N_p/k_i)$ $(N_c < N_p)$ is given by

$$\begin{aligned} x(k_i+1/k_i) &= Ax(k_i)+B\Delta u(k_i) && (6)\\ x(k_i+2/k_i) &= Ax(k_i+1/k_i)+B\Delta u(k_i+1) \\ &= A^2x(k_i)+AB\Delta u(k_i)+B\Delta u(k_i+1) && (7)\\ &\vdots \\ x(k_i+N_p/k_i) &= A^{N_p}x(k_i)+A^{N_p-1}B\Delta u(k_i)+A^{N_p-2}B\Delta u(k_i+1) \\ &+ \ldots + A^{N_p-N_c-1}B\Delta u(k_i+N_c-1) && (8)\end{aligned}$$

Define vectors
$X = \left[\, x(k_i+1/k_i)^T \quad x(k_i+2/k_i)^T \quad x(k_i+3/k_i)^T \quad \ldots \quad x(k_i+N_p/k_i)^T \,\right]^T$ and
$\Delta U = \left[\, \Delta u(k_i)^T \quad \Delta u(k_i+1)^T \quad \Delta u(k_i+2)^T \quad \ldots \quad \Delta u(k_i+N_c-1)^T \,\right]^T$ Then equations (6)-(8) can be written in a compact matrix form as

$$X = Fx(k_i)+\Phi\Delta U \tag{9}$$

where

$$F = \begin{bmatrix} A \\ A^2 \\ A^3 \\ \vdots \\ A^{N_p} \end{bmatrix}; \Phi = \begin{bmatrix} B & 0 & 0 & \ldots & 0 \\ AB & B & 0 & \ldots & 0 \\ A^2B & AB & B & \ldots & 0 \\ \vdots & & & & \\ A^{N_p-1}B & A^{N_p-2}B & A^{N_p-3}B & \ldots & A^{N_p-N_c-1}B \end{bmatrix}$$

If the control system is designed for disturbance rejection (regulation purpose), then a cost function J_r can be chosen as

$$J_r = X^T\bar{Q}_rX+\Delta U^T\bar{R}\Delta U \tag{10}$$

where $\bar{Q}_r$ is an $(n\times N_p)\times(n\times N_p)$ matrix, and $\bar{R}$ is a $(p\times N_c)\times(p\times N_c)$ matrix. Specifically, $\bar{Q}_r$ and $\bar{R}$ are block diagonal matrices with Q_r $(n\times n)$ and R $(p\times p)$ on their diagonals respectively. Q_r and R are both symmetric positive definite matrices. Substituting Equation (9) into Equation (10), minimisation of the quadratic cost function (10) in the absence of constraints yields the future optimal control trajectory given by

$$\Delta U = -(\Phi^T\bar{Q}_r\Phi+\bar{R})^{-1}\Phi^T\bar{Q}_rFx(k_i) \tag{11}$$

If set-point following is also required in the control system design, then the future plant output is predicted using

$$Y = \bar{C}X = \bar{C}Fx(k_i)+\bar{C}\Phi\Delta U \tag{12}$$

where $\bar{C}$ $((q\times N_p)\times(n\times N_p))$ is an N_p- block diagonal matrix with the C matrix in Equation (3) sitting on its diagonal. Assume that the future set-point trajectory is governed by $s(k_i+1), s(k_i+2), s(k_i+3), \ldots, s(k_i+N_p)$, and we write

$$S = \left[\; s(k_i+1)^T \quad s(k_i+2)^T \quad s(k_i+3)^T \quad \ldots \quad s(k_i+N_p)^T \;\right]^T$$

Then the cost function for set-point following can be chosen as follows:

$$J_s = (S - Y)^T \bar{Q}(S - Y) + \Delta U^T \bar{R} \Delta U \tag{13}$$

where $\bar{Q}$ has dimension of $(q \times N_p) \times (q \times N_p)$ and $\bar{Q} > 0$. By substituting Equation (12) into Equation (13), we obtain the optimal control trajectory as

$$\Delta U = -(\Phi^T \bar{C}^T \bar{Q} \bar{C} \Phi + \bar{R})^{-1} \Phi^T \bar{C}^T \bar{Q}(\bar{C} F x(k_i) - S) \tag{14}$$

For the quadratic cost functions J_r and J_s, without hard constraints, the resultant control system is in the form of linear time invariant state feedback control (see Equation (11) or (14)). This point becomes clearer when we let

$$K_d = (\Phi^T \bar{Q}_r \Phi + \bar{R})^{-1} \Phi^T \bar{Q}_r F$$

$$K_s = (\Phi^T \bar{C}^T \bar{Q} \bar{C} \Phi + \bar{R})^{-1} \Phi^T \bar{C}^T \bar{Q} \bar{C} F$$

$$R_s = (\Phi^T \bar{C}^T \bar{Q} \bar{C} \Phi + \bar{R})^{-1} \Phi^T \bar{C}^T \bar{Q} S$$

Notice that the above matrices only depend on the plant model parameters and the set point trajectory within the prediction horizon (assuming invariant for the majority of applications), thus they are constant matrices for linear time invariant systems. From the receding horizon control principle, the actual incremental control signal at time k_i is

$$\Delta u(k_i) = -k_d x(k_i) \quad (regulation) \tag{15}$$

or

$$\Delta u(k_i) = -k_s x(k_i) + r_s \quad (setpoint\ \ tracking) \tag{16}$$

where k_d, k_s and r_s are the first p rows of the matrices K_d, K_s and R_s respectively. Equation (15) or (16) is in the standard form of linear time invariant state feedback control.

Note that to guarantee stability in a systematic way, terminal state constraint is often recommended to be used in the algorithm (see section 4). Nevertheless, in the long history of model predictive control, closed-loop stability is traditionally achieved through tuning of the performance parameters such as the prediction horizon, control horizon and the weighting matrices $\bar{Q}$ and $\bar{R}$ (see for example, Clarke et al, 1987 a,b, Ricker, 1991). In this situation, it is often necessary to carefully select the tuning parameters so that the eigenvalues of the $A - Bk_s$ (or $A - Bk_d$) are inside the unit circle of the complex plane and on the desired locations. There is little computational demand in the (unconstrained) form of MPC since all gain matrices can be calculated off-line and verified through simulation studies.

2.3 State Estimation

Note that Equations (15) or (16) contains the state vector $x(k_i)$ at the sampling instant k_i. Thus the control strategy is essentially a state feedback control when the state variable $x(k)$ is measurable. In the situation where $x(k)$ is not measurable, an observer is required to estimate $x(k)$. Methods for estimating the state variable $x(k)$ have been described in the literature (see for example Kailath, 1980, Astrom and Wittenmark, 1984, Goodwin et al. 2000).

Essentially, the state variable $x(k_i)$ is estimated via an observer of the form:

$$\hat{x}(k_i+1) = A\hat{x}(k_i) + B\Delta u(k_i) + J_{ob}(y(k_i) - C\hat{x}(k_i)) \tag{17}$$

where J_{ob} is called the observer's gain. If the pair (A, C) is observable, then for a single output case, a pole -assignment strategy can be used to determine J_{ob} such that the eigenvalues of the observer error, i.e. of the matrix $A - J_{ob}C$ are at desired locations. Here, the assumption that the pair (A, C) is detectable indicates that all of its unobserverable poles are stable and its remaining poles can be arbitrarily positioned inside the unit circle. For a multi-output system, J_{ob} can be calculated recursively using a Kalman filter. This requires iterative equations, $m = 0, 1, \ldots,$

$$J_{ob}(m) = AP(m)C^T(\Sigma + CP(m)C^T)^{-1} \tag{18}$$

$$P(m+1) = A\{P(m) - P(m)C^T(\Sigma + CP(m)C^T)^{-1}CP(m)\}A^T + \Omega \tag{19}$$

where Ω and Σ are the covariance matrices of the disturbance $\bar{\xi}(k)$ and $\eta(k)$ (see Equation (3)). More specifically,

$$E\{\bar{\xi}(k)\bar{\xi}(\tau)^T\} = \Omega\delta(k-\tau)$$

$$E\{\eta(k)\eta(\tau)^T\} = \Sigma\delta(k-\tau)$$

$P(0)$ satisfies

$$E\{[x(0) - \hat{x}(0)][x(0) - \hat{x}(0)]^T\} = P(0)$$

Assuming that the system (C, A) is detectable from the output $y(k)$ (i.e. there are no unstable states whose response can not be 'seen' from the output) and $(A, \Omega^{1/2})$ is stabilizable, then, as $k \to \infty$, the steady state solutions of Equations (18) and (19) satisfy the discrete- time-algebraic-Riccati-Equation:

$$P(\infty) = A\{P(\infty) - P(\infty)C^T(\Sigma + CP(\infty)C^T)^{-1}CP(\infty)\}A^T + \Omega \tag{20}$$

and

$$J_{ob}(\infty) = AP(\infty)C^T(\Sigma + CP(\infty)C^T)^{-1} \tag{21}$$

Also the eigenvalues of $A - J_{ob}(\infty)C$ are guaranteed to be inside the unit circle (i.e. stable). To avoid confusion, it is emphasised that the iterative solution of the Riccati Equation (19) is not required in real-time. The observer gain is calculated off-line.

Tuning Observer Dynamics

It is often that the covariance matrices Ω and Σ are unknown. Thus, in practice, we choose an Ω, a Σ and an initial $P(0)$ to calculate an observer gain J_{ob} by solving the Riccati equation iteratively until the solution converges to a constant matrix. Then the closed-loop system obtained is analyzed with respect to location of eigenvalues contained in $A - J_{ob}C$, the transient response of the observer, robustness and effect of noise on the response. The elements of the covariance matrices are modified until a desired result is obtained. Such a trial-and-error procedure can be time consuming and frustrating, and is one of the challenges we face when using Kalman filter based multivariable system design.

In some circumstances, however, it is possible to specify a region in which the closed-loop observer error system poles should reside and to enforce this in the solution. We propose a simple approach, along a similar line to the classic approach in Anderson and Moore (1971, 1979), in which the closed-loop observer poles are assigned inside a circle with a pre-specified radius α $(0 < \alpha < 1)$. The procedure is summarized as follows. Let the error of the estimated state $\tilde{x}(k) = x(k) - \hat{x}(k)$. Then the observer error system is

$$\tilde{x}(k+1) = (A - J_{ob}C)\tilde{x}(k) \tag{22}$$

We perform the transformation $\hat{A} = A/\alpha$ and $\hat{C} = C/\alpha$ where $0 < \alpha < 1$, leading to a transformed system

$$\tilde{x}_t(k+1) = \frac{1}{\alpha}(A - \hat{J}_{ob}C)\tilde{x}_t(k) = (\hat{A} - \hat{J}_{ob}\hat{C})\tilde{x}_t(k) \tag{23}$$

Solving the iterative Equations (18) and (19), or the steady state Riccati Equation (20) by using $\hat{A}$ and $\hat{C}$ to replace A and C matrices, the eigenvalues of $\hat{A} - \hat{J}_{ob}(\infty)\hat{C}$ are guaranteed to be inside the unit circle (i.e. stable). The resultant observer gain $\hat{J}_{ob}$ is then applied to the original observer system (22), leading to the closed-loop characteristic equation

$$det(zI - (A - \hat{J}_{ob}C)) = det(zI - (\hat{A} - \hat{J}_{ob}\hat{C}) \times \alpha) \tag{24}$$

Therefore, we conclude that the eigenvalues of $(A - \hat{J}_{ob}C)$ are equal to the eigenvalues of $\hat{A} - \hat{J}_{ob}\hat{C}$ multiplying by the factor α, which guarantees that the eigenvalues of the observer error system with $\hat{J}_{ob}$ reside inside the circle of α. This procedure makes the direct connection to the observer dynamics via the choice of α. The trial-and- error procedure can be reduced to choose a suitable α along with Ω and Σ to achieve desired closed-loop performance.

The simulation study in Section 7 will demonstrate the relationship between the choice of performance parameter matrices and effect of measurement noise. It will also show that in the second example, with the choice of α, the trial-and-error procedure can be simplified for multi-output systems.

As a note, in the absence of 'hard' constraints on the control and process variables, the proposed model predictive control system reduces to linear state estimate feedback control with $x(k_i)$ replaced by the estimated state $\hat{x}(k_i)$. Hence the certainty equivalence principle will apply; i.e. the control system (assuming a perfect model) will be stable if the observer error system and the closed-loop system with MPC are both independently stable, since the eigenvalues of the combined control system are the union of the eigenvalues of these two separate systems. With this insight, we can tune the predictive controller and the observer separately before implementation.

2.4 Implementation of the MPC strategy

Receding horizon control is realized by calculating ΔU for every sampling instant based on Equation (13) with $x(k_i)$ replaced by $\hat{x}(k_i)$, and then implementing the increment control signal only at the first sampling instant, i.e. we apply (assuming p inputs)

$$\Delta u(k) = \Delta U(1:p) \tag{25}$$

where $\Delta U(1:p)$ is the first p elements of ΔU and the rest of the elements in ΔU are ignored. Then as time progresses from k_i to $k_i + 1$, a new estimate $\hat{x}(k_i + 1)$ is obtained, ΔU is re-calculated based on this new state information. The advantage of the MPC algorithm using a velocity form lies in its simplicity when implementing. Two final year undergraduate students from Newcastle University successfully implemented the continuous time version of the model predictive control algorithm (Wang, 2001a) on a distillation column (Johnston, 1999) and a food extruder (Smith, 2000). In the sequel, we will further discuss the implementation issue.

To avoid confusion, we clarify several types of signals that distinguish simulation studies from implementation on a real plant. Let u_{ss} and y_{ss} denote the steady state input and output signals on a real plant. u_{ss} and y_{ss} are determined by the operating conditions of the plant. When the plant operating condition changes, u_{ss} and y_{ss} may change. Let u_{act} and y_{act} denote the actually measured plant input and output signals. The input and output signals used in the state space model (1) are called deviation variables, defined as $u(k) = u_{act}(k) - u_{ss}$ and $y(k) = y_{act}(k) - y_{ss}$. In contrast to simulation studies where the steady-state signals u_{ss} and y_{ss} are often assumed to be zero, in implementation of a control system, the steady-state signals u_{ss} and y_{ss} play important roles and must be taken into account carefully.

The implementation procedure is summarised as follows.

Step one (initialization of the control system): At the process steady state operation, set the initial conditions of the estimated state variable $\hat{x}(0) = 0$, the incremental control signal $\Delta u(0) = 0$, the steady state value in the process output is y_{ss} and the change in the process output $y(0) = 0$, the measured plant input signal is u_{act}.
Step two: Calculate the estimated state variable $\hat{x}(k+1)$ using

$$\hat{x}(k+1) = A\hat{x}(k) + B\Delta u(k) + J_{ob}(y(k) - C\hat{x}(k)) \tag{26}$$

Step three: Calculate the control signal

$$u_p(k+1) = u_{act} + \Delta u(k) \tag{27}$$

Step four: Send this control signal u_p to the actuator.
Step five: Calculate

$$\Delta U = -(\Phi^T \bar{C}^T \bar{Q} \bar{C} \Phi + \bar{R})^{-1} \Phi^T \bar{C}^T \bar{Q} (\bar{C} F \hat{x}(k+1) - S) \tag{28}$$

Step six: Calculate $\Delta u(k+1) = \Delta U(1:p)$ (the first p elements in ΔU).
Step seven: Measure the actual plant output y_{act} and input u_{act}, and $y(k+1) = y_{act} - y_{ss}$.

Step eight: Go to step two with update on $\Delta u(k+1)$ and $y(k+1)$.

Note that in the initialization procedure, $\hat{x}(0) = 0$. This initial 'guess' is close to reality in the sense that the composite state variable $x(k)$ (see Equation (2)) consists of $\Delta x_m(k)$ and $y(k)$, and both quantities can be approximated by zero when plant is operating at steady state. In addition, in step three the control signal to the plant is calculated based on the actual measurement of the plant input signal instead of the previously calculated one. This is because in some circumstance, for instance in the situation of saturation, the calculated control signal may differ from the actual plant input signal.

When 'hard' constraints are imposed, the implementation procedure is essentially the same except that ΔU should be calculated using an on-line optimization algorithm discussed in the next section. Even with this seemly simple implementation procedure, the challenge for implementing a real time control system is tremendous. The issues of safety, reliability and robustness demand huge effort from control engineers in practice.

3. Model Predictive Control Design with Hard Constraints

One of the main features of MPC is the ability to handle hard constraints. This section shows how to incorporate hard constraints in the design.

3.1 Formulation of Constrained Control

The typical constraints are given as follows:

(1) Manipulated variable constraints. These are hard limits on the control signal $u(k)$, for example of valve saturation, $u(k)$ is typically specified as

$$u_{low} \leq u(k) \leq u_{high} \tag{29}$$

(2) Manipulated variable rate constraints. These are hard constraints on the size of the control signal moves. i.e. on the rate of change of the manipulated variables. We specify this in the form:

$$\Delta u_{low} \leq \Delta u(k) \leq \Delta u_{high} \tag{30}$$

(3) Output variable constraints. Hard or soft limits on the outputs of the system are imposed to avoid overshoot and undershoot. Typically the specification can be formulated as follows:

$$y_{low} \leq y(k) \leq y_{high} \tag{31}$$

The constraints can change with respect to time and operating region. The limits should be updated according to variations of the constraints. Among the three types of constraints, input constraints are most frequently encountered and can be imposed naturally through the physical system, however, the output constraints can result in infeasibility problems and the constraints can be violated because of model-plant mismatch (Zafiriou, 1991).

Traditionally the constraints are imposed for all future sampling instants. All constraints are expressed in terms of the parameter vector ΔU. In the case of manipulated variable constraint, we express:

$$\begin{bmatrix} u(k+1) \\ u(k+2) \\ u(k+3) \\ \vdots \\ u(k+N_c) \end{bmatrix} = \begin{bmatrix} I \\ I \\ I \\ \vdots \\ I \end{bmatrix} u_{act}(k) + \begin{bmatrix} I & 0 & 0 & \dots & 0 \\ I & I & 0 & \dots & 0 \\ I & I & I & \dots & 0 \\ \vdots \\ I & I & \dots & I & I \end{bmatrix} \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \Delta u(k+2) \\ \vdots \\ \Delta u(k+N_c-1) \end{bmatrix} \quad (32)$$

Writing Equation (32) in a compact matrix form, with C_1 and C_2 corresponding to the appropriate matrices, then the constraints for the control movement are imposed as

$$-(C_1 u_{act}(k) + C_2 \Delta U) \leq -U_{low} \quad (33)$$

$$(C_1 u_{act}(k) + C_2 \Delta U) \leq U_{high} \quad (34)$$

where U_{low} and U_{high} are column vectors with N_c elements of u_{low} and u_{high}, respectively. Similarly, for the increment of the control signal, we have the constraints:

$$-\Delta U \leq -\Delta U_{low} \quad (35)$$

$$\Delta U \leq \Delta U_{high} \quad (36)$$

where ΔU_{low} and ΔU_{high} are column vectors with N_c elements of Δu_{low} and Δu_{high}, respectively. For the output constraints, we have

$$-(\bar{C} F \hat{x}(k_i) + \bar{C}\Phi \Delta U) \leq -Y_{low} \quad (37)$$

$$\bar{C} F \hat{x}(k_i) + \bar{C}\Phi \Delta U \leq Y_{high} \quad (38)$$

where Y_{low} and Y_{high} are column vectors with N_p elements of y_{low} and y_{high}, respectively. Finally, the model predictive control problem with respect to hard constraints is proposed as (for the setpoint following case):
Minimize

$$J = \Delta U^T (\Phi^T \bar{C}^T \bar{Q} \bar{C} \Phi + \bar{R}) \Delta U + 2 \Delta U^T \Phi^T \bar{C}^T \bar{Q} (\bar{C} F \hat{x}(k_i) - S) + constant \quad (39)$$

subject to the inequality constraints

$$\begin{bmatrix} M_1 \\ M_2 \\ M_3 \end{bmatrix} \Delta U \leq \begin{bmatrix} N_1 \\ N_2 \\ N_3 \end{bmatrix} \quad (40)$$

where $M_1 = \begin{bmatrix} -C_2 \\ C_2 \end{bmatrix}$; $N_1 = \begin{bmatrix} -U_{low} + C_1 u_{act}(k) \\ U_{high} - C_1 u_{act}(k) \end{bmatrix}$; $M_2 = \begin{bmatrix} -I \\ I \end{bmatrix}$; $N_2 = \begin{bmatrix} -\Delta U_{low} \\ \Delta U_{high} \end{bmatrix}$; $M_3 = \begin{bmatrix} -\bar{C}\Phi \\ \bar{C}\Phi \end{bmatrix}$; $N_3 = \begin{bmatrix} -Y_{low} + \bar{C} F \hat{x}(k_i) \\ Y_{high} - \bar{C} F \hat{x}(k_i) \end{bmatrix}$.

When the constraints are fully imposed, the number of constraints is equal to $4 \times p \times N_c + 2 \times q \times N_p$ which is greater than the dimension of the decision variable ΔU.

To solve this quadratic programming problem in real time requires a reliable and efficient numerical algorithm and a large computational capacity. Because the receding horizon control law implements the first control movement and ignores the rest of the calculated future control signals, a question naturally arises whether it is necessary to impose constraints on all future trajectories of both the control signals and system output. Generally speaking, the stability condition for the constrained model predictive control requires all constraints in the future to be satisfied. From a performance point of view, the constraints to be satisfied in the future are, sometimes, necessary as indicated by an example in Goodwin et al. (2000). However, a recent study done by Zheng (1999) has indicated that for stable systems, reducing the number of constraints on the future control movements could cause little performance deterioration. The problem of whether one can reduce the number of constraints in the formulation is still open as demonstrated in the simulation example presented later.

3.2 Numerical Solutions to the Constrained Control Problem

Complete analytical solutions to the constrained control problem have not been obtained yet, although some progresses have been made towards this goal (Bemporad et al., 1999, Seron et al. 2000). Numerical solutions to this type of problems have been studied extensively in the literature (see for example, Luenberger, 1984, Fletcher, 1981).

For simplicity of notation, we write the optimization problem (see Equations (39) and (40)) as minimizing

$$J = \frac{1}{2}\Delta U^T E \Delta U + \Delta U^T F \tag{41}$$

subject to

$$M\Delta U \leq \gamma \tag{42}$$

where M is a matrix of $m \times n$, and $m > n$, i.e. the number of constraints is greater than the number of variables. If all constraints were active in model predictive control, then the regularity condition ($m < n$) (Luenberger, 1984) would not be satisfied.

There are four types of methods devised for solving this constrained control problem. They are primal methods, dual methods, penalty and barrier methods and Lagrange methods. They, respectively, work in the spaces of dimension $n - m$, m, n and $n + m$. In the literature of model predictive control, the primal methods have dominated the numerical solutions (see for example, Muske and Rawlings, 1993b, Ricker, 1985, Zafiriou, 1991) until recent years specially tailored interior-point methods applicable to MPC have appeared (Rao et al. 1998, Gopal and Biegler, 1998, Hansson, 2000). These algorithms solve the constrained control problem by utilizing the special structure of the control system. Another interesting approach to the constrained control problem was proposed by Rossiter and Kouvaritakis (1993) where it was solved iteratively using a weighted least squares type of algorithm (Lawson's algorithm). To improve computational speed when implementing on-line, an alternative to quadratic programming is to use linear programming for large scale systems (Doyle III et al., 1997).

A primal method (active set)

By a primal method of solution, we mean a search method that works on the original problem directly by searching through the feasible region for the optimal solution. Each point in the process is feasible and the value of the objective function constantly decreases. Primal methods possess several significant advantages that recommend their use as general procedures applicable to the constrained control problem. First since each search point generated in the search procedure is feasible, if the process is terminated before reaching the solution, the terminal point is feasible. Thus this final point is a feasible and probably nearly optimal solution to the original problem and therefore may represent an acceptable solution to the constrained control problem. A second attractive feature of primal methods is that, often, it can be guaranteed that if they generate a convergent sequence, the limit point of that sequence must be at least a local constrained minimum. In other words, the global convergence characteristics of these methods are often satisfactory. As our original problem where the number of constraints is greater than the number of variables, active set methods (among the family of primal methods) are appropriate. The idea underlying active set methods is to partition the inequality constraints into two groups: those that are to be treated as active and those that are to be treated as inactive. The constraints treated as inactive are essentially ignored. Since, in the constrained control problem, not all constraints are active at the same time, we could then take a guess at the active constraints and ignore the inactive constraints so that the regularity condition (number of constraints is less than the number of variables) is satisfied, at least at each computational step.

The necessary conditions for this optimization problem (Kuhn-Tucker condition) are

$$\begin{aligned} E\Delta U + F + M^T\lambda &= 0 \\ M\Delta U - \gamma &\leq 0 \\ \lambda^T(M\Delta U - \gamma) &= 0 \\ \lambda &\geq 0 \end{aligned} \tag{43}$$

where the vector λ contains the Lagrange multipliers. These conditions can be expressed in a simpler form in terms of the set of active constraints. Let S_{act} denote the index set of active constraints. Then the necessary conditions become

$$\begin{aligned} E\Delta U + F + \sum_{i \subset S_{act}} \lambda_i M_i^T &= 0 \\ M_i\Delta U - \gamma_i &= 0 \quad i \subset S_{act} \\ M_i\Delta U - \gamma_i &< 0 \quad i \not\subset S_{act} \\ \lambda_i &\geq 0 \quad i \subset S_{act} \\ \lambda_i &= 0 \quad i \not\subset S_{act} \end{aligned} \tag{44}$$

where M_i is the ith row of the M matrix. It is clear that if the active set were known, the original problem could be replaced by the corresponding problem having equality constraints only. Alternatively, suppose an active set is guessed and the corresponding equality constrained problem is solved. Then if the other constraints are satisfied and the Lagrange multipliers turn out to be nonnegative, that solution would be correct.

The idea of active set methods is to define at each step of an algorithm a set of constraints, termed the working set, that is to be treated as the active set. The working set is chosen to be a subset of the constraints that are actually active at the current point, and hence the current point is feasible for the working set. The algorithm then proceeds to move on the surface defined by the working set of constraints to an improved point. At each step of the active set methods, an equality constraint problem is solved. If all the Lagrange multipliers $\lambda_i \geq 0$, then the point is a local solution to the original problem. If, on the other hand, there exists a $\lambda_i < 0$, then the objective can be decreased by relaxing the constraint i (i.e. taking it out from the constraint equation). During the course of minimization, it is necessary to monitor the values of the other constraints to be sure that they are not violated since all points defined by the algorithm must be feasible. It often happens that while moving on the working surface, a new constraint boundary is encountered. It is convenient to add this constraint to the working set, then proceeding on the redefined working surface. This active set strategy can be automated using a gradient projection method (Luenberger, 1984).

Primal-Dual method in MPC

In the active set methods, the active constraints need to be 'guessed', if there are many constraints such as the situation that there are many input and output variables, the computational load is quite large. A dual method can be used to identify the constraints that are not active. They can then be eliminated in the solution.

The dual problem to the original quadratic problem (Luenberger, 1969) is described as follows. Assuming feasibility (i.e. there is an ΔU such that $M\Delta U < \gamma$), the problem is equivalent to

$$\max_{\lambda\geq 0}\min_{\Delta U}[\frac{1}{2}\Delta U^T E\Delta U + \Delta U^T F + \lambda^T(M\Delta U - \gamma)] \tag{45}$$

The minimization over ΔU is unconstrained and is attained by

$$\Delta U = -E^{-1}(F + M^T\lambda) \tag{46}$$

Substituting Equation (46) into (45), the dual problem becomes

$$\max_{\lambda\geq 0}(-\frac{1}{2}\lambda^T H\lambda - \lambda^T K - \frac{1}{2}F^T E^{-1}F) \tag{47}$$

where $H = ME^{-1}M^T$ and $K = \gamma + ME^{-1}F$. Thus the dual is also a quadratic programming problem. Equation (47) is equivalent to

$$\min_{\lambda\geq 0}(\frac{1}{2}\lambda^T H\lambda + \lambda^T K + \frac{1}{2}\gamma^T E^{-1}\gamma) \tag{48}$$

Note that the dual problem may be much easier to solve than the primal problem because the constraints are simpler. A simple algorithm, called Hildreth's Quadratic Programming Procedure (Luenberger, 1969), was proposed for solving this dual problem.

In this algorithm, the direction vectors were selected to be equal to the basis vectors $e_i = (0,0,\ldots,1,\ldots,0,0)$. Then the λ vector can be varied one component at a time. At a given step in the process, having obtained a vector $\lambda \geq 0$, we fix our attention on a single component λ_i. The objective function may be regarded as a quadratic function in this single component. We adjust λ_i to minimize the objective function. If that requires $\lambda_i < 0$, we set $\lambda_i = 0$. In any case, the objective function is decreased. Then we consider the next component λ_{i+1}.

If we consider one complete cycle through the components to be one iteration taking the vector λ^m to λ^{m+1}, the method can be expressed explicitly as

$$\lambda_i^{m+1} = max(0, w_i^{m+1}) \tag{49}$$

where

$$w_i^{m+1} = -\frac{1}{h_{ii}}[k_i + \sum_{j=1}^{i-1} h_{ij}\lambda_j^{m+1} + \sum_{j=i+1}^{n} h_{ij}\lambda_j^m] \tag{50}$$

Hildreth's Quadratic Programming Procedure is easy for computer programming and the convergence of the algorithm can also be proved. However, a drawback of this extreme simple algorithm is that the convergence rate is too slow for the nonnegative λ_is, depending on the distribution of the eigenvalues of H matrix, which is similar to other gradient based methods (Luenberger, 1984). It seems that for the negative and zero λ_is, the convergence rate is fast, in which several case studies have found that reliable estimation of the inactive constraints can be obtained with 10 iterations. Because Hildreth's Quadratic Programming algorithm converges, given a sufficient number of iterations, the algorithm will provide reliable results. Thus, what we propose is to use this simple algorithm to identify the set of inactive constraints. By deleting the inactive constraints, the original inequality constraint problem becomes an equality constraint problem, and both ΔU and the nonnegative λ_is can be calculated in a closed form solution. Let the set of active constraints be represented by M_{act} and γ_{act}. The optimal solution to the constrained control problem (see Equation (44)) is given by the solution of the linear equations:

$$\begin{bmatrix} E & M_{act}^T \\ M_{act} & 0 \end{bmatrix} \begin{bmatrix} \Delta U \\ \lambda_{act} \end{bmatrix} = \begin{bmatrix} -F \\ \gamma_{act} \end{bmatrix} \tag{51}$$

Explicitly:

$$\begin{aligned} \lambda_{act} &= -(M_{act}E^{-1}M_{act}^T)^{-1}(\gamma_{act} + M_{act}E^{-1}F) && (52) \\ \Delta U &= -E^{-1}(F + M_{act}^T\lambda_{act}) && (53) \end{aligned}$$

A comment follows immediately. A matrix eigenvalue property says (Anderson and Moore, 1979) that if a matrix A is $c \times d$ and B is $d \times c$ with $c \geq d$, then the eigenvalues of AB are the same as those of BA together with $c - d$ zero eigenvalues. Thus, in our case, it becomes obvious that if the number of active constraints is greater than the number of decision variables, the matrix $M_{act}E^{-1}M_{act}^T$ contains zero eigenvalue(s), hence becomes singular. As a result, we can check whether the number of rows of M_{act} is less than the number of columns as the first step to determine whether the Hildreth's programming procedure has converged. If so, we proceed to calculate λ_{act} and ΔU using Equations (52) and (53). However, if one or more elements in λ_{act} turn out to be negative in the closed form solution, a remedy is to go back to the Hildreth programming for recalculating the set of inactive constraints. Nevertheless, without much priori knowledge in numerical computation, this semi-closed form solution gives a means for on-line implementation of model predictive control in the presence of hard constraints. The convergence of Hildreth programming has been proved, thus given sufficient computational time, the algorithms is reliable.

An alternative as to what we propose here is to use Hildreth Quadratic programming procedure for identification of those 'obviously' inactive constraints (van Donkelaar et al. 1999). After having deleted the inactive constraints, the decision variable ΔU is parameterised in terms of those constraints that are left in the computational procedure. Because the dimensionality of the original problem is significantly reduced, the active set methods can be used efficiently to find the decision variable. In Tsang and Clarke (1988), optimal solutions were derived for constrained GPC of SISO systems with a control horizon of 1 or 2. The essence of their approach was to take a 'guess' at the active constraints for the two special cases and apply the closed- form solution. Hence, the approach by Tsang and Clarke (1988) and the primal and dual approach proposed here are similar in spirit.

Use of Barrier function Methods in MPC

Another class of optimization methods which have been applied in MPC is the Barrier function methods (McGovern and Feron, 1999). These are characterised by their property of preserving strict constraint feasibility at all times, by using a term which is infinite on the constraint boundaries. The sequence of minimizer is also feasible, hence the techniques are also referred to as interior point methods which have found favourable status in model predictive control design (Rao et al., 1998, Hansson, 2000). The two most important cases are the inverse barrier function

$$F(\Delta U,r) = J(\Delta U) + r\sum_i (M_i\Delta U - \gamma_i)^{-1} \tag{54}$$

and the logarithmic barrier function

$$F(\Delta U,r) = J(\Delta U) - r\sum_i log(M_i\Delta U - \gamma_i) \tag{55}$$

where the coefficient r is used to control the barrier function iteration. A sequence of $\{r^k\} \to 0$ is chosen, which ensures that the barrier term becomes more and more negligible except close to the boundary. The optimization procedure is traditionally implemented as follows.

(1) Choose a fixed sequence $\{r^k\} \to 0$, typically, $\{1, 0.1, 0.01, \ldots\}$.
(2) For each r^k find a local minimizer, $\Delta U(r^k)$, say to minimize $F(\Delta U, r)$.
(3) Terminate when $\sum_i (M_i\Delta U - \gamma_i)^2$ is sufficiently small.

The barrier function methods are simple to implement and the computation speed is also fairly fast because the optimization can be performed for the primal variables.

Unfortunately barrier function algorithms suffer from numerical difficulties in the limit due to ill -conditioning and large gradients. Moreover, the barrier function is undefined for infeasible points in the solutions. Another difficulty is that an initial interior feasible point is required, and this in itself, can be a non-trivial problem in model predictive control when output constraints are involved.

4. Stability Properties

When the constraints in model predictive control are activated, the control law effectively becomes a nonlinear control case. Thus, the stability properties of linear time invariant systems do not apply. However, a remarkable property of model predictive control is that one can establish stability of the closed-loop system under certain circumstances. Researchers have devoted considerable energy to this topic in the 1990's (see Mayne et al. (2000) for a summary of literature in this area). We illustrate the key ideas through the general consensus, followed by a close examination of the stability properties of the model predictive control algorithms using a velocity form model.

4.1 General approach

We make the following assumptions:
Assumption 1: An additional constraint is placed on the final state of the receding horizon optimization problem: $x(k+N_p)=0$, where $x(k+N_p)$ is the terminal state resulting from the control sequence $\Delta u(k), \Delta u(k+1), \Delta u(k+2), \ldots, \Delta u(k+N_c-1)$.
Assumption 2: For each sampling instant, k, there exists a solution ΔU such that the cost function J is minimized subject to the given inequality constraints (42) and terminal equality constraint $x(k+N_p)=0$.
Stability result: Subject to assumptions 1 and 2, the closed-loop model predictive control system is asymptotically stable.

In the following, we sketch the procedure for establishing this stability result. We consider the cost function (10) for regulation purpose. The key to the stability result is to construct a Lyapunov function for the model predictive control system. Choose the Lyapunov function V(x(k),k) as the minimum of the finite horizon cost function

$$V(x(k),k) = \hat{X}_0^T \bar{Q} \hat{X}_0 + \Delta \hat{U}_0^T \bar{R} \Delta \hat{U}_0 \tag{56}$$

where $\hat{X}_0 = Fx(k) + \Phi \Delta \hat{U}_0$ and $\Delta \hat{U}_0$ is, at time k, the parameter vector solution of the original cost function (10) respect to both inequality and equality constraints. It is easily seen that $V(x(k),k)$ is positive definite and $V(x(k),k)$ tends to infinity as $x(k)$ tends to infinity. Similarly,

$$V(x(k+1),k+1) = \hat{X}_1^T \bar{Q} \hat{X}_1 + \Delta \hat{U}_1^T \bar{R} \Delta \hat{U}_1 \tag{57}$$

and $\hat{X}_1 = Fx(k+1) + \Phi \Delta \hat{U}_1$ and $\Delta \hat{U}_1$ is the parameter vector solution at $k+1$. A feasible solution (not the optimal one) for $x(k+1)$ in the receding horizon is to shift the elements in $\Delta \hat{U}_0$ one step forward and replace the last element by zero, i.e. $\Delta \hat{u}(k+1), \Delta \hat{u}(k+$

$2),\ldots,\Delta\hat{u}(k+N_c-1),0$. We denote this vector by $\Delta\hat{U}_{01}$. Because of optimality in the solution of $\Delta\hat{U}_1$, it is seen that

$$V(x(k+1),k+1) \leq \bar{V}(x(k+1),k+1) \tag{58}$$

where $\bar{V}(x(k+1),k+1)$ is similar to Equation (57) except that $\Delta\hat{U}_1$ is replaced by $\Delta\hat{U}_{01}$. The difference between $V(x(k+1),k+1)$ and $V(x(k),k)$ is then bounded by

$$V(x(k+1),k+1) - V(x(k),k) \leq \bar{V}(x(k+1),k+1) - V(x(k),k) \tag{59}$$

Note that $\bar{V}(x(k+1,k+1) - V(x(k),k) = x(k+N_p)^T Qx(k+N_p) - x(k+1)^T Qx(k+1) - \Delta u(k)^T R\Delta u(k)$. By Assumption 1, we have

$$\bar{V}(x(k+1,k+1) - V(x(k),k) = -x(k+1)^T Qx(k+1) - \Delta u(k)^T R\Delta u(k) \tag{60}$$

Hence, the difference of the Lyapunov function is

$$V(x(k+1),k+1) - V(x(k),k) \leq -x(k+1)^T Qx(k+1) - \Delta u(k)^T R\Delta u(k) \tag{61}$$

which we see is negative. Notice that $V(x(k+1),k+1)$ and $V(x(k),k)$ are the minimums of the cost function at $k+1$ and k. Summing (61) from 1 to ∞ and noticing the cancellation on the left hand of the equation, we obtain

$$\begin{aligned} & \sum_{k=1}^{\infty}(V(x(k+1),k+1) - V(x(k),k)) = V(x(\infty),\infty) - V(x(1),1) \\ \leq \quad & -\sum_{k=1}^{\infty}(x(k+1)^T Qx(k+1) + \Delta u(k)^T R\Delta u(k)) \end{aligned} \tag{62}$$

Hence

$$V(x(\infty),\infty) + \sum_{k=1}^{\infty} x(k+1)^T Qx(k+1) + \Delta u(k)^T R\Delta u(k)) \leq V(x(1),1) \tag{63}$$

By assuming feasibility of the optimal solution, we have $V(x(1),1) < \infty$. With assumption of continuity in signals, the optimal control solution at k is a feasible solution at $k+1$. This, in turn, ensures feasibility of the optimal control solutions for all k. Hence the infinite sequence of left handside of the equation is bounded. Therefore, as $k \longrightarrow \infty$, $x(k) \longrightarrow 0$ and $\Delta u(k) \longrightarrow 0$. Stability of the closed-loop system is hence established.

4.2 Terminal state constraint for MPC algorithms using velocity model forms

The approaches to guarantee closed-loop stability slightly differ for the two mainstreams MPC algorithms. General consensus is, if the plant model is stable, the algorithms that compute the optimal control variable (Rawlings, 2000; Mayne, 2000) will produce a guaranteed stable closed-loop system if the prediction horizon approaches infinity. With finite horizon, if the plant model is stable and the optimal control signal is computed, a terminal state weight can be used to achieve the same effect as the case of infinity horizon model predictive control (Rawlings and Muske, 1993). Unfortunately, the stability treatment for the class of the MPC algorithm using the velocity form model is more complicated. In the velocity form model, the composite system matrix A has q eigenvalues on the unit circle of the complex plane (see Equation (4)), thus, it is impossible to take the prediction horizon to infinity. This is because $A^{N_p} \longrightarrow \infty$ as $N_p \longrightarrow \infty$. So it is a general agreement that to guarantee closed-loop stability in a systematic way for the class of algorithms using velocity form model, either terminal constraints or an inner loop stabiliser (Rossiter, et al. 1998) will have to be used.

The early approaches to terminal state constraints include Clarke and Scattolini (1991) and Mosca and Zhang (1992). The basic idea is illustrated as follows. Let x_T contain the terminal state information, for instance, if set-point is used, part of the terminal states corresponding to the output y are expected to equal the set-point while the rest of the state variables are expected to be zero. Then the constraints are specified as

$$A^{N_p}x(k_i) + \begin{bmatrix} A^{N_p-1}B & A^{N_p-2}B & \dots & A^{N_p-N_c-1}B \end{bmatrix} \Delta U - x_T = 0 \tag{64}$$

Without hard constraints, analytical solution of the optimal incremental control signal can be found by following the same procedure as the treatment of equality constraints. However, as in the case of primal- dual solution in 3.2, the number of equality constraints (the dimensionality of state variable) must be less than the number of decision variables (the dimensionality of ΔU), otherwise, the regularity condition can not be satisfied. In addition, if the hard constraints are specified in the design, the number of total active constraints, including the active inequality constraints and the terminal equality constraints, could be well above the number of decision variables, hence numerical difficulty could be encountered. To reduce the number of terminal state constraints, an improvement has been proposed for imposing the terminal state constraints only on the unstable modes of the composite model (Rawlings and Muske, 1993). The key idea is to force the restriction that $x(k_i + N_p)$ lies in the stable manifold of A so that the output converges to zero according to the stable modes of A beyond the prediction horizon N_p. Let W_s, V_s and Λ_s be the right/left eigenvectors and eigenvalues of the composite system matrix A, associated to the stable eigenvalues, and V_u be the left eigenvectors associated to the unstable eigenvalues. Then the terminal constraint, with respect to the unstable modes of the system, is

$$V_u x(k_i + N_p) = 0 \tag{65}$$

The remaining stable part of the system, after the prediction horizon N_p and for some arbitrary m, is

$$x(k_i + N_p + m) = A_s^m x(k_i + N_p) \tag{66}$$

where $A_s = W_s \Lambda_s V_s$. Since A_s contains the stable part of the composite model, a terminal state weight can be used in the cost function. The essence is to find a P matrix from the Lyapunov equation

$$P = A_s^T P A_s + C^T C \tag{67}$$

along with $y(k_i) = Cx(k_i)$, such that

$$\sum_{i=N_p}^{\infty} y(k_i+i)^T y(k_i+i) = x(k_i+N_p)^T P x(k_i+N_p) \tag{68}$$

Finally, with some modification to the cost function to include terminal state weight, we arrive at

$$J = x(k_i+N_p)^T P x(k_i+N_p) + Y^T \bar{Q} Y + \Delta U^T \bar{R} \Delta U \tag{69}$$

subject to the terminal state constraint (65) and other inequality constraints in the performance specification. With terminal state constraint and state weight, the problem concerning closed-loop stability of model predictive control is converted to finding a feasible solution of the constrained optimization problem. Infeasibility in the optimization may occur if there is a conflict between the terminal constraint and the other inequality constraints specified in the algorithm.

5. Numerical Condition of the Algorithm

In the following analysis, we assume a single input and single output system, where Q becomes a diagonal matrix with equal elements on its diagonal, thus it does not have an effect on the numerical condition of the Hessian matrix. To this end, we note that

$$\bar{C}\Phi = \begin{bmatrix} h_0 & 0 & 0 & \dots & 0 \\ h_1 & h_0 & 0 & \dots & 0 \\ h_2 & h_1 & h_0 & \dots & 0 \\ \vdots & & & & \\ h_{N_p-1} & h_{N_p-2} & h_{N_p-3} & \dots & h_{N_p-N_c-1} \end{bmatrix} \tag{70}$$

where $h_i = CA^iB$ is the ith impulse response coefficient of the composite model (3). We pay particular attention to the Toeplitz matrix $\Phi^T \bar{C}^T \bar{C}\Phi$ which is the Hessian matrix in the MPC algorithm. This type of Toeplitz matrix is also called the correlation matrix which has been studied by Grenander and Szego (1958). The analysis presented in (Grenander and Szego, 1958) showed that assuming that $|\lambda_i(A)| < 1$ for all i, for infinite N_p, as N_c increases, the minimum and maximum eigenvalues of a correlation matrix asymptotically converge to the minimum and maximum of the power spectrum of the signal that generated this correlation matrix. This result has been used in recent years for analysing numerical conditions of identification and filter design algorithms (see for example, Wahlberg, 1991, Beaufays, 1995). In the model predictive control algorithm, the signal that generated this correlation matrix is the set of impulse response coefficients of the composite model. Hence the spectrum of the signal is given by

$$S_{hh}(w) = |C(e^{iw}I - A)^{-1}B|^2 \tag{71}$$

However, because the composite model used for prediction contains an integrator, there is at least one eigenvalue of A existed on the unit circle, the result from Grenander and

Szego (1958) does not apply. Nevertheless, simulation result in Wang (2001c) demonstrated that indeed both minimum and maximum eigenvalues of the Hessian matrix tend to infinite when the prediction horizon increases.

The analysis presented above indicates that the key to improving the numerical condition of MPC algorithm is to use a stable model for prediction. Rossiter et al. (1998) proposed to use state feedback control as a mechanism to improve the numerical condition of the MPC algorithm using velocity form model. The central idea is to let

$$\Delta u(k) = -K_f x(k) + \delta u(k)$$

then the composition model used for model predictive control design becomes

$$\begin{aligned} x(k+1) &= (A - BK_f)x(k) + B\delta u(k) + \bar{\xi}(k) \\ y(k) &= Cx(k) + \eta(k) \end{aligned} \tag{72}$$

where the state feedback controller K_f is chosen such that the eigenvalues of the state space model (72), $\lambda_i(A - BK_f) < 1 - \varepsilon$, $\varepsilon > 0$. In this case, for infinite prediction horizon, as N_c increases, $\lambda_{min}(H) \longrightarrow \min_w |C(e^{iw}I - (A - BK_f))^{-1}B|^2$ and $\lambda_{max}(H) \longrightarrow \max_w |C(e^{iw}I - (A - BK_f))^{-1}B|^2$.

6. Model Predictive Control Design Using Basis Functions

In essence, the core technique in the design of discrete MPC is based on modelling the future control trajectory, either the control signal ($u(k)$) itself or the difference of the control signal ($\Delta u(k)$), by pulse operators to obtain the linear-in-the-parameter relation for predicted output. As a consequence, in case of rapid sampling, complicated process dynamics and /or high demands on closed-loop performance, satisfactory approximation of the control signal requires a very large number of forward shift operators, and leads to poorly numerically conditioned solutions and heavy computational load when implemented on-line. Recent work by Wang (2001a, 2001b, 2003) introduced a more appropriate expansion related to Laguerre networks in the design of model predictive control. In this framework, for a set of Laguerre functions, $l_1(k), l_2(k), \ldots, l_N(k)$, then:

$$\Delta u(k_i + k) \approx \sum_{m=1}^{N} c_m(k_i) l_m(k) \tag{73}$$

where $\Delta u(k_i + k) = u(k_i + k) - u(k_i + k - 1)$ with k_i being the initial time of the moving horizon window and k being the future sampling instant; N is the number of terms used in the expansion and c_m, $m = 1, 2, \ldots, N$, are the coefficients, and they are functions of the initial time of the moving horizon window, k_i. The set of discrete Laguerre functions satisfies the following difference equation, with the scaling factor $0 \leq a < 1$:

$$L(k+1) = \Omega L(k) \tag{74}$$

where $\beta = 1 - a^2$ and

$$L(k) = \begin{bmatrix} l_1(k) & l_2(k) & \dots l_N(k) \end{bmatrix}^T$$

$$\Omega = \begin{bmatrix} a & 0 & 0 & \dots & 0 \\ \beta & a & 0 & \dots & 0 \\ -a\beta & \beta & a & \dots & 0 \\ a^2\beta & -a\beta & \beta & \dots & 0 \\ \vdots & & & & \\ (-1)^{N-2}a^{N-2}\beta & (-1)^{N-3}a^{N-3}\beta & \dots & \beta & a \end{bmatrix}$$

with initial condition

$$L(0)^T = \sqrt{(1-a^2)} \begin{bmatrix} 1 & -a & a^2 & -a^3 & \cdots & (-1)^{N-1}a^{N-1} \end{bmatrix}$$

When the scaling factor a is chosen to be zero, the set of Laguerre functions becomes a set of pulse functions, hence the design is identical to the design earlier in the paper. However, a larger value of a can be selected to achieve a long control horizon with a fewer number of parameters N that is required in the optimization procedure. It was shown that the number of terms could be reduced to a fraction of that required by the usual procedure (Wang 2001a, Wang 2004).

Within the framework of using basis functions, the prediction of the future state variable, $x(k_i + m/k_i)$ at sampling instant m, becomes

$$x(k_i + m/k_i) = A^m x(k_i) + \sum_{i=0}^{m-1} A^{m-i-1} BL(i)^T \eta \tag{75}$$

where $\eta = \begin{bmatrix} c_1 & c_2 & \dots & c_N \end{bmatrix}^T$ To compute the prediction, the convolution sum $S_c(m) = \sum_{i=0}^{m-1} A^{m-i-1} BL(i)^T$ can be solved sequentially through the relation:

$$S_c(m) = AS_c(m-1) + S_c(1)(\Omega^{m-1})^T \tag{76}$$

where $S_c(1) = BL(0)^T$ and $m = 2,3,4,\dots,N_p$. Although the derivation of the model predictive control algorithm in section 2 is easy to understand and the implementation is simple, it requires a large storage of computer memory to form the system matrices. This problem is overcome when using the basis function approach. More specifically, when using basis function approach, the cost function (13) can be equivalently represented by

$$J_s = \sum_{m=1}^{N_p} [s(k_i+m) - y(k_i+m/k_i)]^T Q [s(k_i+m) - y(k_i+m/k_i)] + \eta^T R \eta \tag{77}$$

where Q and R are positive definite matrices. With the future prediction of the state variable $x(k_i + m/k_i)$ defined by (75), the quadratic cost function J_s can be written explicitly as

$$J_s = \eta^T \left(\sum_{m=1}^{N_p} \phi(m) Q \phi(m)^T + R \right) \eta$$

$$+2\eta^T(-\sum_{m=1}^{N_p}\phi(m)Q(r(k_i+m)+CA^m x(k_i))+constant \quad (78)$$

where $\phi(m)^T = CS_c(m)$, with $S_c(m)$ iteratively computed using Equation (76). This computation procedure can be performed sequentially, avoiding usage of large computer memory.

7. Simulation Example

7.1 Model Predictive Control of Double Integrating Plant

Consider a continuous time double integrator plant that is sampled at an interval $\Delta t = 1$ second. For pedagogical reasons, we choose this 'toy' example with the intention to make the design procedure more transparent for beginners.

The corresponding discrete-time model is of the form

$$\begin{aligned} x_m(k+1) &= A_m x_m(k) + B_m u(k) \\ y(k) &= C_m x_m(k) \end{aligned} \quad (79)$$

where
$A_m = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}; B_m = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}; C_m = \begin{bmatrix} 1 & 0 \end{bmatrix}$
The augmented model for this plant is given by

$$\begin{aligned} x(k+1) &= Ax(k) + B\Delta u(k) \\ y(k) &= Cx(k) \end{aligned} \quad (80)$$

where $A = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}; B = \begin{bmatrix} 0.5 \\ 1 \\ 0.5 \end{bmatrix}; C = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$

In the following, we demonstrate two aspects of model predictive controller design: (a) constrained control and (b) observer design . We are not to use terminal state constraint in the design. Thus, closed-loop stability is achieved through choice of tuning parameters. As there are three eigenvalues on the unit circle, if the terminal state constraint were to be used, the control horizon would be required to be at least 4 (and above) to incorporate one constraint on the control signal.

a. Constrained control

A pole-assignment strategy is used to design an observer based on the model (80). The set of observer's poles are chosen to be (0.01,0.0105,0.011) corresponding to a fast dynamic response. The resultant observer gain is $J_{ob} = [1.9685 \;\; 0.9688 \;\; 2.9685]^T$. In the model predictive control design, the prediction horizon, N_p, is selected to be 150 samples and the control horizon N_c is chosen to be 3. The weighting on the output error, Q, is the unit matrix and the weighting on Δu is chosen to be 0.1. With this set of tuning parameters, the state feedback control gain is $k_s = [1.4723 \;\; 1.7304 \;\; 0.9794]$

which effectively yields a set of closed-loop eigenvalues $\lambda_1 = -0.0469 + j0.1989$, $\lambda_2 = -0.0469 - j0.1989$ and $\lambda_3 = 0.1375$. As all closed-loop poles are close to the origin of the complex plane, the closed-loop dynamic response is expected to be fast without constraint. Figure 1 shows that without constraints on the input signal, the closed-loop response for a step input disturbance is indeed a fast response. In this case, to reject the disturbance, the steady state of the control signal converges to 10. Therefore, we require $u(k) > 10$ at steady state. Two cases of constraints are presented. One is $-14 \leq u(k) \leq 14$ and the another is $-10.5 \leq u(k) \leq 10.5$. The results are compared in Figure 2. It can be seen that the responses are very different from the ones mentioned in the unconstrained case. As the amplitude of the constraint in the control signal reduces, the closed-loop performance deteriorates significantly. In fact, the closed-loop system becomes unstable, when the control signal is constrained to be $(-10, 10)$. As discussed before, if we could reduce the number of constraints for future control signal movement, then the computational demand in MPC would be reduced significantly. However, the situation is more complicated than it appears to be at the first sight. We change the disturbance from a step disturbance to a pulse input disturbance (see Figure 3) and the other design specifications remain the same except that the constraint for the control signal is $-12 \leq u(k) \leq 12$. We compare the cases where this constraint is imposed on $u(k)$ and the constraint is imposed on both $u(k)$ and $u(k+1)$. It is seen from Figure 3 that the responses are different for these two cases. Apparently, to impose constraints in both current and future control movement improves the closed-loop performance, although the future control movement we have calculated is not implemented due to the receding control principle. It is also discovered that the closed-loop responses are indifferent from imposing constraints on $u(k)$ and $u(k+1)$ if another constraint $u(k+2)$ is imposed.

From this example, it is seen that when constraints become active, the closed-loop system is a nonlinear control system, its performance deviates from the performance of linear unconstrained control and can be difficult to predict. To guarantee closed-loop stability, the constraints must be feasible. The problem of how far the constraints should be imposed into the future movements of both input and output signals is still open. In theory, the constraints should be imposed for all future movements. It is seen that the closed-loop performance is difficult to predict when the constraints are active.

b. Observer design

It is worth mentioning that the observer design is a dual problem to the state feedback controller design, hence the matlab functions in the commercial software package such as 'place' (for pole placement) and 'lqr' (linear quadratic regulator) can be directly used for calculating the observer gain J_{ob} where we replace A by A^T and B by C^T. Part of the challenges in observer design is to select the Ω and Σ matrices (see Equations (18) and (19)) to reflect the disturbance characteristics in a given system. Since the disturbance characteristics are seldom known, a trial and error procedure is often used. For a single output system, an easy way is to select a set of desired closed-loop poles as the performance parameters. The rule of thumb is that the observer dynamics should be faster than the MPC dynamic response if the measurement noise is not severe and permits such a choice. In other words, we try to select a set of closed-loop poles for

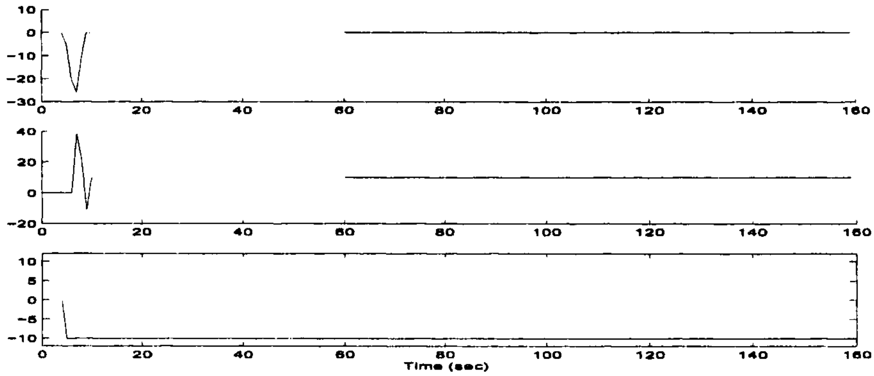

Figure 1: *Closed-loop response without constraints: top figure- output response; middle figure- control signal response; bottom figure- input disturbance*

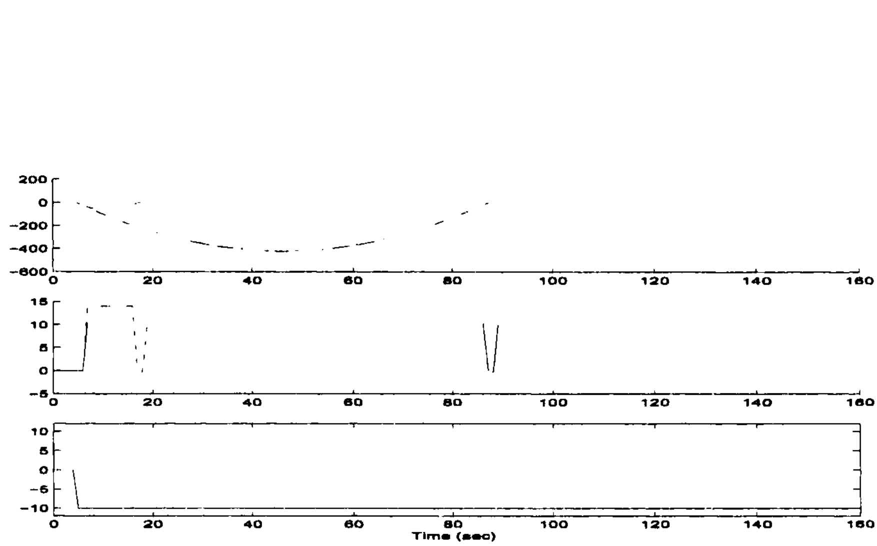

Figure 2: *Closed-loop response with constraints (solid line:* $-10.5 \leq u(k) \leq 10.5$*; dash-dotted line:* $-14 \leq u(k) \leq 14$*.): top figure- output response; middle figure- control signal response; bottom figure- input disturbance.*

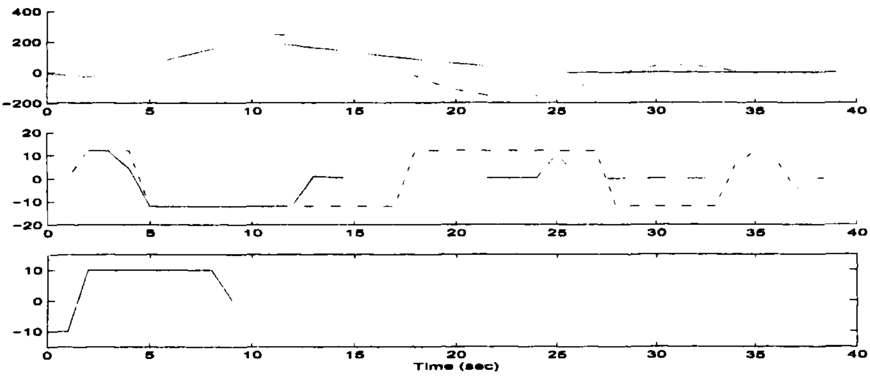

Figure 3: *Closed-loop response with constraints (solid line: constraint on current control; dash-dotted line: constraint on both current and future control): top figure- output response; middle figure- control signal response; bottom figure- input disturbance.*

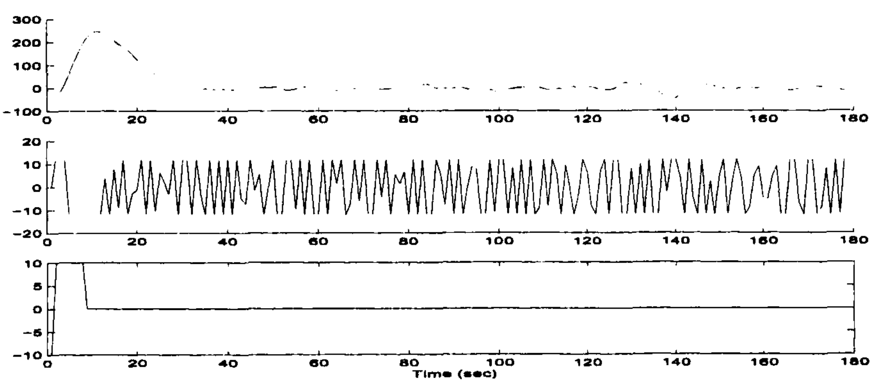

Figure 4: *Closed-loop response with constraints using high observer gain: top figure- output response; middle figure- control signal response; bottom figure- input disturbance.*

the observer that are closer to the origin (inside the unit circle) than the set of closed-loop poles produced by the model predictive controller ($\lambda_i(A - Bk_s)$). The observer poles can also be selected using an optimal criterion via symmetric root locus (Kailath, 1980). For the case of multi-output system, we often use Equations (18) and (19). One way to choose Ω and Σ is to select Ω to be an identify matrix and vary the elements in Σ. In general, smaller elements in Σ correspond to larger observer gain J_{ob}, and larger elements in Σ correspond to smaller observer gain J_{ob}. Larger observer gain means faster observer dynamic response, however, it also amplifies measurement noise.

To demonstrate the effect of measurement noise, we add a random noise sequence with standard deviation of one to the process output. It is seen from Figure 4 that this measurement noise severely affects the control signal. To reduce the effect of measurement noise, we select the closed-loop poles to be $[0.8 \quad 0.84 \quad 0.88]$, which effectively yields the observer gain $J_{ob} = [0.0714 \quad 0.0038 \quad 0.48]^T$. In comparison to the previous case, the observer gain has been reduced significantly. Figure 5 shows that the effect of measurement noise on the control signal has been reduced dramatically by comparing to Figure 4.

7.2 Model Predictive Control of Extrusion Process

Extrusion is a continuous process in which a rotating screw is used to force the food material through the barrel of the machine and out through a narrow die opening. In this process the material is simultaneously transported, mixed, shaped, stretched and sheared under elevated temperature and pressure. The extruder in the study is an APV-MPF40 co-rotating twin-screw extruder. A project on system identification of food extruder was performed by the author and her colleagues (Wang, et al., 2003). In the identification project, a multi-frequency relay feedback control system (Wang and Cluett, 2000) was implemented on the food extruder to ensure safe operation of the process when doing identification experiments and to obtain experimental data that have relevant frequency content for dynamic modelling. Continuous time transfer function models were estimated using the state-variable filter approach presented in Wang and Gawthrop (2001). Suppose that u_1, u_2, y_1 and y_2 represent screw speed, liquid pump speed, SME and motor torque respectively. Then the continuous time model for the food extruder is

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{81}$$

where

$$G_{11} = \frac{0.21048s + 0.00245}{s^3 + 0.302902s^2 + 0.066775s + 0.002186}$$

$$G_{12} = \frac{-0.001313s^2 + 0.000548s - 0.000052}{s^4 + 0.210391s^3 + 0.105228s^2 + 0.00777s + 0.000854}$$

$$G_{21} = \frac{0.000976s - 0.000226}{s^3 + 0.422036s^2 + 0.091833s + 0.003434}$$

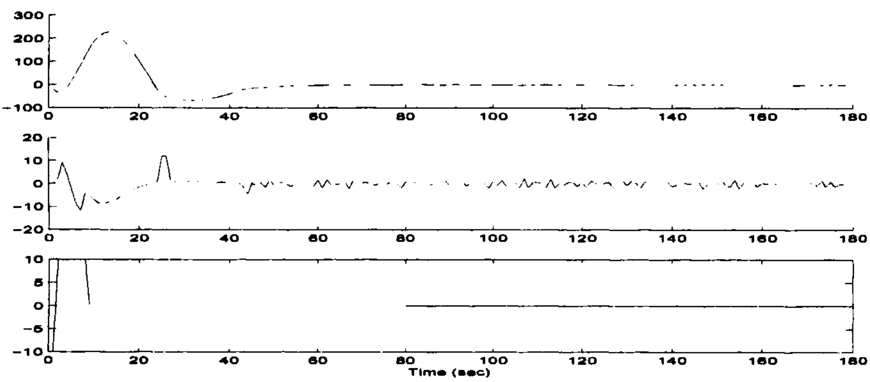

Figure 5: *Closed-loop response with constraints using low observer gain: top figure- output response; middle figure- control signal response; bottom figure- input disturbance.*

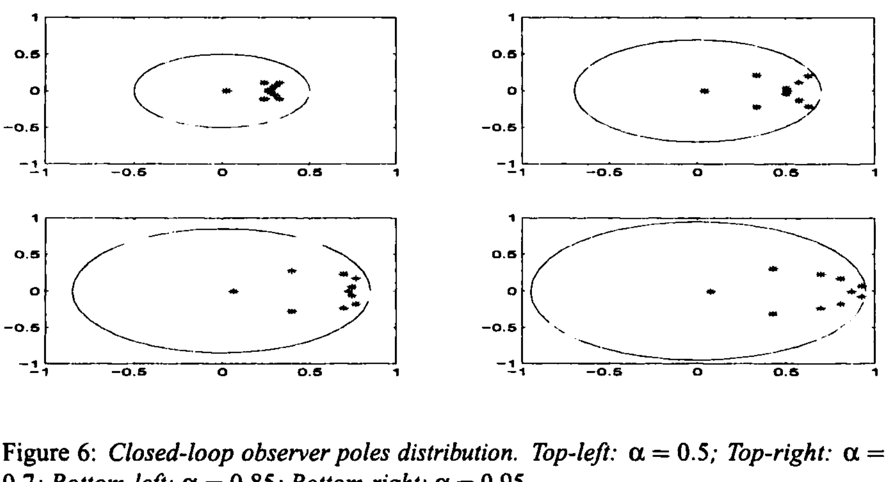

Figure 6: *Closed-loop observer poles distribution. Top-left:* $\alpha = 0.5$*; Top-right:* $\alpha = 0.7$*; Bottom-left:* $\alpha = 0.85$*; Bottom-right:* $\alpha = 0.95$.

$$G_{22} = \frac{-0.000017}{s^2 + 0.060324s + 0.006836}$$

This continuous time transfer function model is used here to demonstrate the design and implementation procedure in model predictive control.

The first step in the design is to convert the transfer function model (81) to a state-space model. This conversion can be performed systematically by using either an observer canonical form or a controller canonical form (Kailath, 1980). This conversion can also be performed using matlab function "tf" and then "ssdata", which put the mathematical model into state space form. With sampling interval $\Delta t = 1(sec)$, the continuous time model is discretized to a discrete state space model.

Design of the Observer

Since this is a multi-output system, the pole-assignment method for designing observer is not guaranteed to produce a valid design. Therefore, we propose to use the Kalman Filtering approach. To demonstrate the observer design with constraints on the location of closed-loop observer poles, the steady state Riccati equation is solved using matlab function "dlqr'. The covariance matrices $\Sigma = 0.1 \times I, \Omega = I$. The dynamic performance of the observer is related to the choice of α circle in which the closed-loop observer poles reside. Figure 6 shows that the eigenvalues of the observer closed-loop system $(A - J_{ob}C)$ for the cases of $\alpha = 0.5, 0.7, 0.85, 0.95$. To examine the effect of observer closed-loop poles in the presence of measurement noise, two cases are considered here. In the simulation, two sets of independent white noise sequences with variance 0.01 are added to SME (y_1) and motor torque (y_2). System steady state operating conditions are: screw speed $u_{1ss} = 300$ rpm; liquid pump speed $u_{2ss} = 2000$ rpm; SME $y_{1ss} = 650$ and Motor torque $y_{2ss} = 42$. A unit step input disturbance is added to the control signal u_1. It is seen that when $\alpha = 0.75$, both control signals are severely corrupted by noise (see Figure 7), however, when we increase α to 0.95, the effect of noise is significantly reduced (see Figure 8). Generally speaking, if there is little measurement noise in the process, the observer poles can be chosen to be close to the origin of the complex plane (equivalent to a small α)to facilitate a fast dynamic response. However, if there is much measurement noise in the process, the observer poles should be chosen to correspond to a slower dynamic response speed (equivalent to a larger α) so that the effect of measurement noise can be reduced.

Case A. Constrained Control: input step disturbance rejection.

We consider a noise free case study. In the design of observer, $\alpha = 0.75$ the covariance matrices $\Sigma = 0.1 \times I, \Omega = I$. The performance parameters in the model predictive control algorithm are specified as: (1) the control horizon $N_c = 4$ for both inputs; (2) the prediction horizon $N_p = 200$ for y_1 and 100 for y_2; (3) The weight matrices in the cost function of model predictive control algorithm are $Q = I$ and $R = I$. The model predictive control system is to maintain steady state operation for SME= 650, Motor Toque = 42. An input step disturbance with amplitude being equal to 3 is added to the screw speed u_1. The operation constraints are specified as (1) screw speed: $295 \leq u_1 \leq 305$ (rpm); (2) $1999.9 \leq u_2 \leq 2000.1$ (rpm)–trying to maintain liquid pump speed at a constant rate; (3) the change rate on screw speed: $-0.1 \leq \Delta u_1 \leq 0.1$; and (4) the change rate

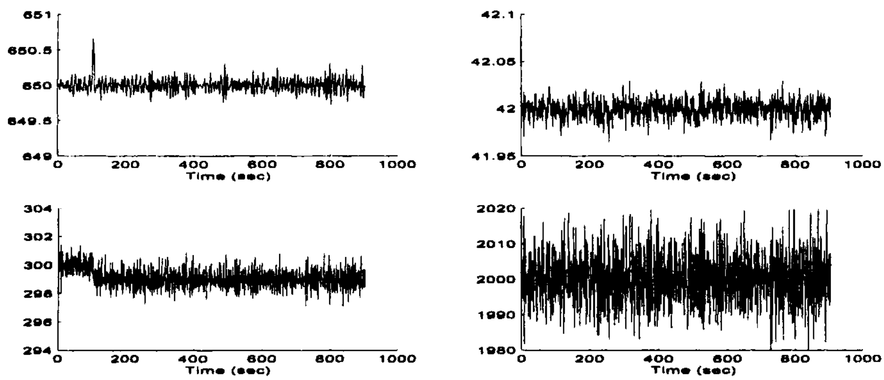

Figure 7: *Closed-loop system response with input disturbance and measurement noise* $\alpha = 0.75$. *Top-left:* y_1; *Top-right:* y_2; *Bottom-left:* u_1; *Bottom-right:* u_2

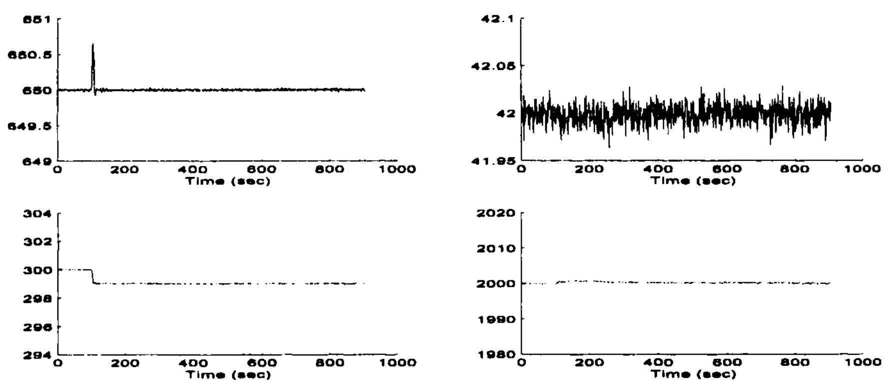

Figure 8: *Closed-loop system response for step input disturbance and measurement noise* $\alpha = 0.75$. *Top-left:* y_1; *Top-right:* y_2; *Bottom-left:* u_1; *Bottom-right:* u_2

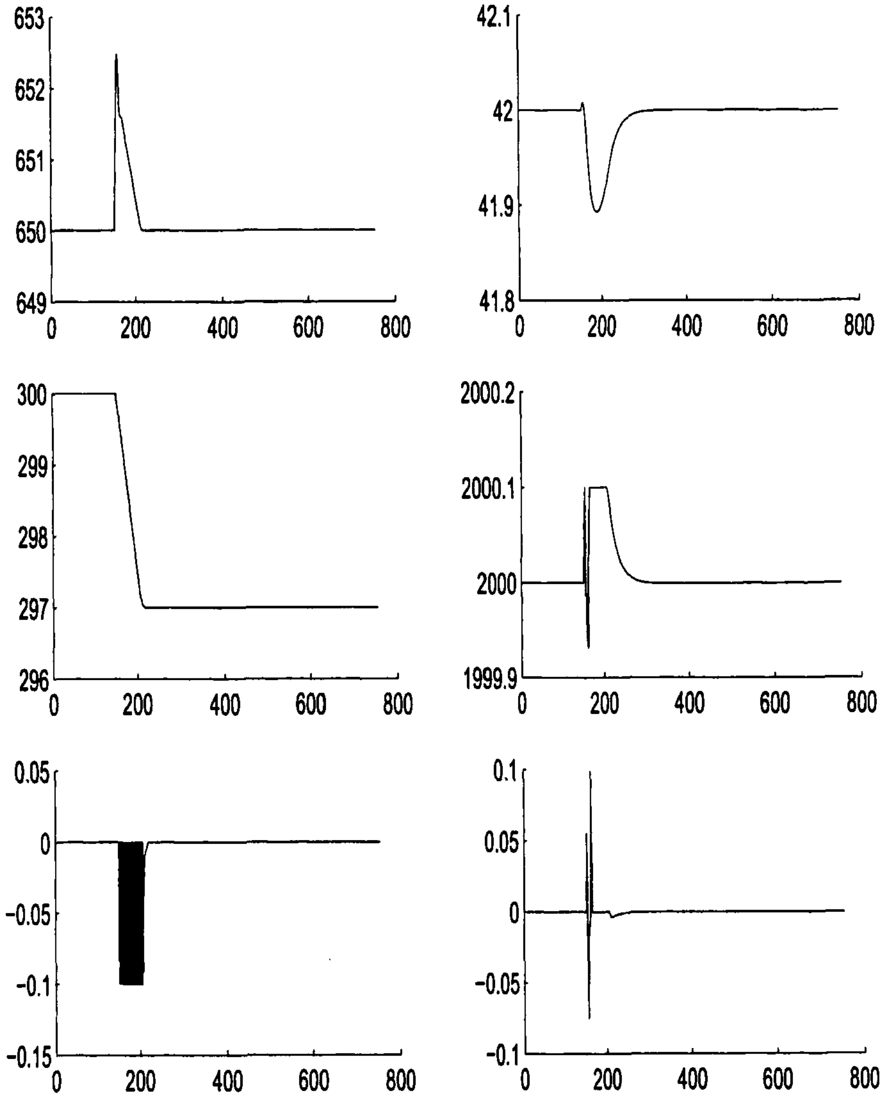

Figure 9: *Constrained Control. Closed-loop system response for step input disturbance. Top-left:* y_1*; Top-right:* y_2*; Middle-left:* u_1*; Middle-right:* u_2*; Bottom-left:* Δu_1*; Bottom-right:* Δu_2

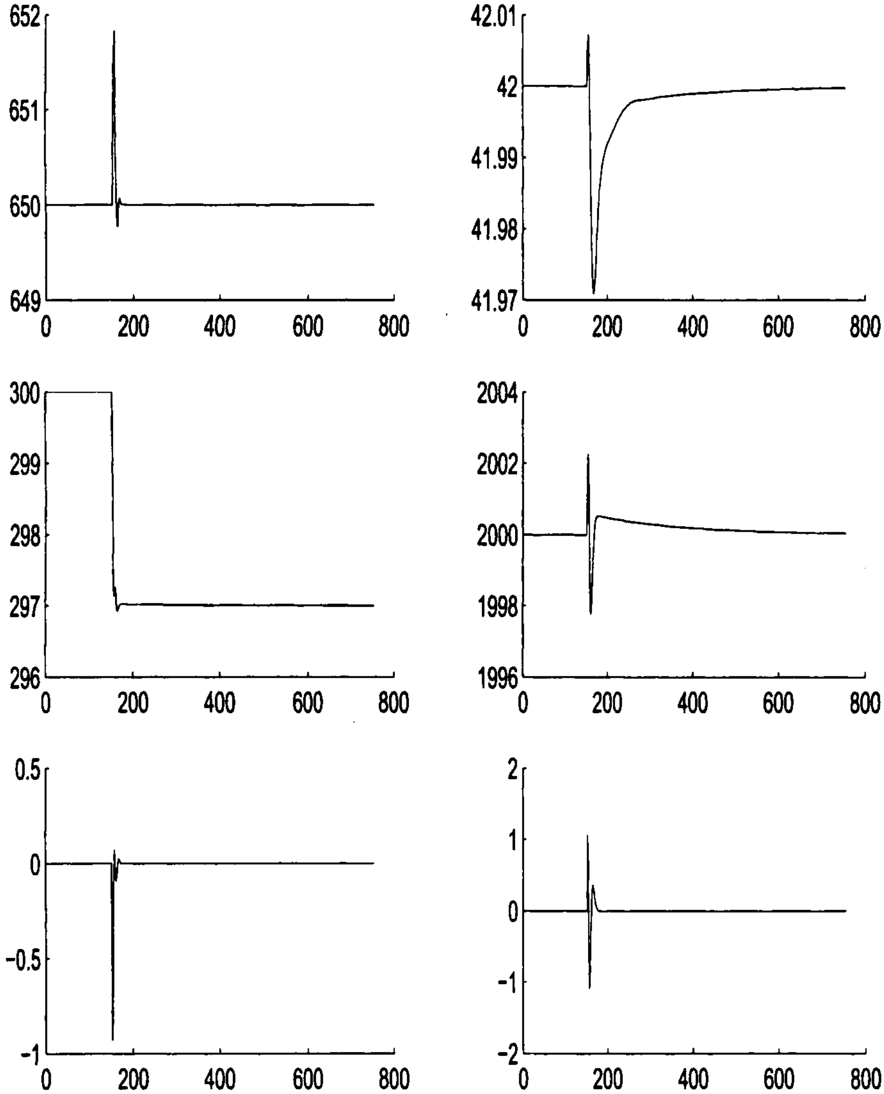

Figure 10: *Unconstrained Control. Closed-loop system response for step input disturbance. Top-left:* y_1*; Top-right:* y_2*; Middle-left:* u_1*; Middle-right:* u_2*; Bottom-left:* Δu_1*; Bottom-right:* Δu_2

on liquid pump speed: $-0.1 \leq \Delta u_2 \leq 0.1$. The simulation results for the constrained control case are presented in Figure 9. It is seen from the figure that all operation constraints are satisfied by the control system. As for comparison purpose, we also present the simulation results for unconstrained control case (see Figure 10)–i.e. the operation constraints are enlarged so that they are not activated in the simulation. Both constrained and unconstrained cases have successfully rejected the input step disturbance.

Case B. Constrained Control: set-point changes.
All design parameters for the observer and the model predictive controller remain the same as the case in input disturbance rejection. However, the setpoint for SME (y_1) has been changed from its steady state operation 650 to 651 to 652, and back to 650 while the setpoint for motor torque (y_2) is maintained as constant. The operation constraints are specified as (1) screw speed: $292 \leq u_1 \leq 302$ (rpm); (2) $1970 \leq u_2 \leq 2001$ (rpm); (3) the change rate on screw speed: $-0.5 \leq \Delta u_1 \leq 0.5$; and (4) the change rate on liquid pump speed: $-0.5 \leq \Delta u_2 \leq 0.5$. Figure 11 shows the simulation results. It is seen that the setpoint changes have been achieved in the constrained control framework, except that the motor torque has not returned to its steady state value in the second step of the setpoint change due to the activated constraint on the liquid pump speed (u_2). To compare, Figure 12 shows the simulation results for the unconstrained control case. It is seen that in order for the motor torque to return to its steady state value in the second step of the step change, the liquid pump speed needs to be dropped to 1964 (rpm).

Case C. Constrained Control: input disturbance rejection with measurement noise.
The simulation conditions and design parameters remain the same as in Case A, except that two sets of white noise sequences with variance of 0.01 are added to the SME and motor torque. It was found that the observer with pole-location constrained (the parameter $\alpha = 0.75$ as chosen in Case A) resulted in infeasibility solutions in the quadratic programming procedure, due to the existence of the noise. By choosing $\alpha = 0.95$, the solutions become feasible. The simulation results are presented in Figure 13. It is seen that the simulation results basically follow the same pattern as the noise-free case, and the disturbance has been rejected and all constraints are satisfied.

8. Conclusions

This paper has given an introduction to model predictive control based on a state space approach. It is seen that when no constraints are imposed, the control system reduces to linear feedback of state estimate and stability is determined by the location of closed-loop poles from both observer and controller. When constraints are introduced, stability is guaranteed with a terminal constraint and a terminal weight. Feasibility of the optimal solution is necessary in achieving closed-loop stability. Two simulation examples are used to demonstrate the design procedure.

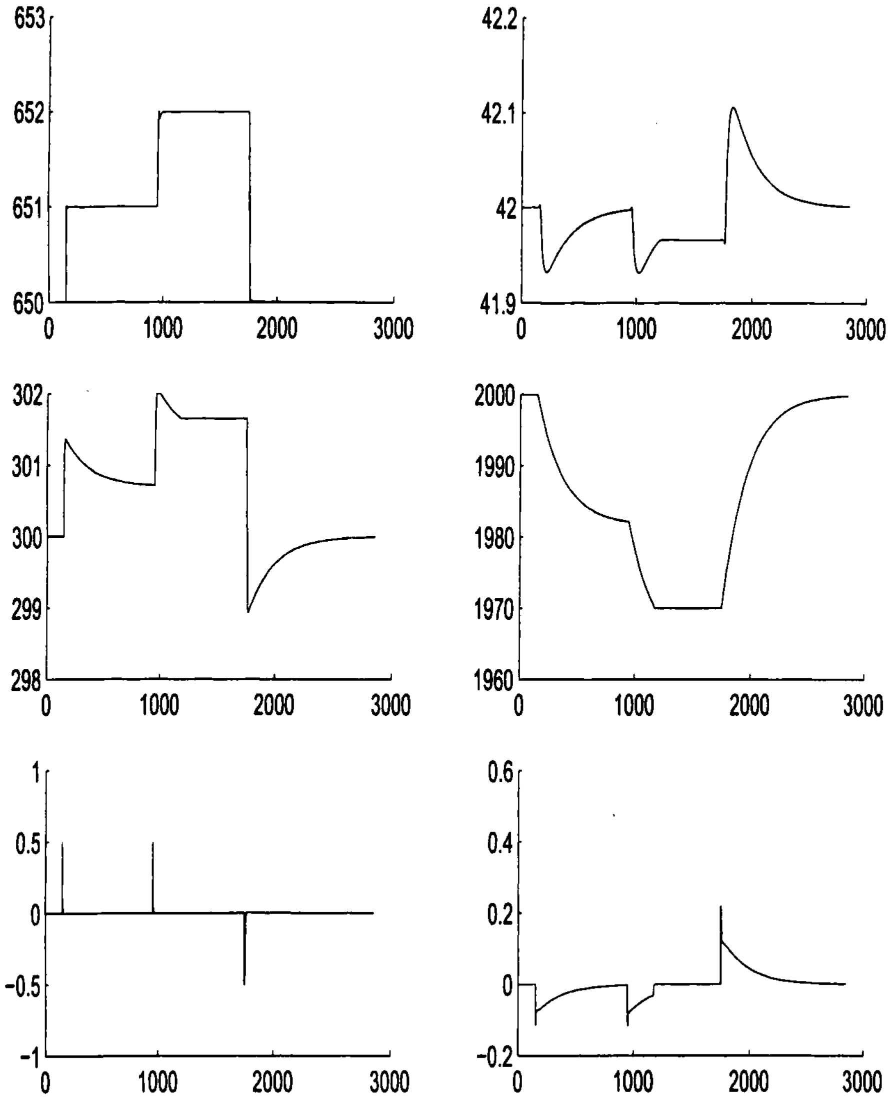

Figure 11: *Constrained Control. Closed-loop system response for setpoint change. Top-left: y_1; Top-right: y_2; Middle-left: u_1; Middle-right: u_2; Bottom-left: Δu_1; Bottom-right: Δu_2*

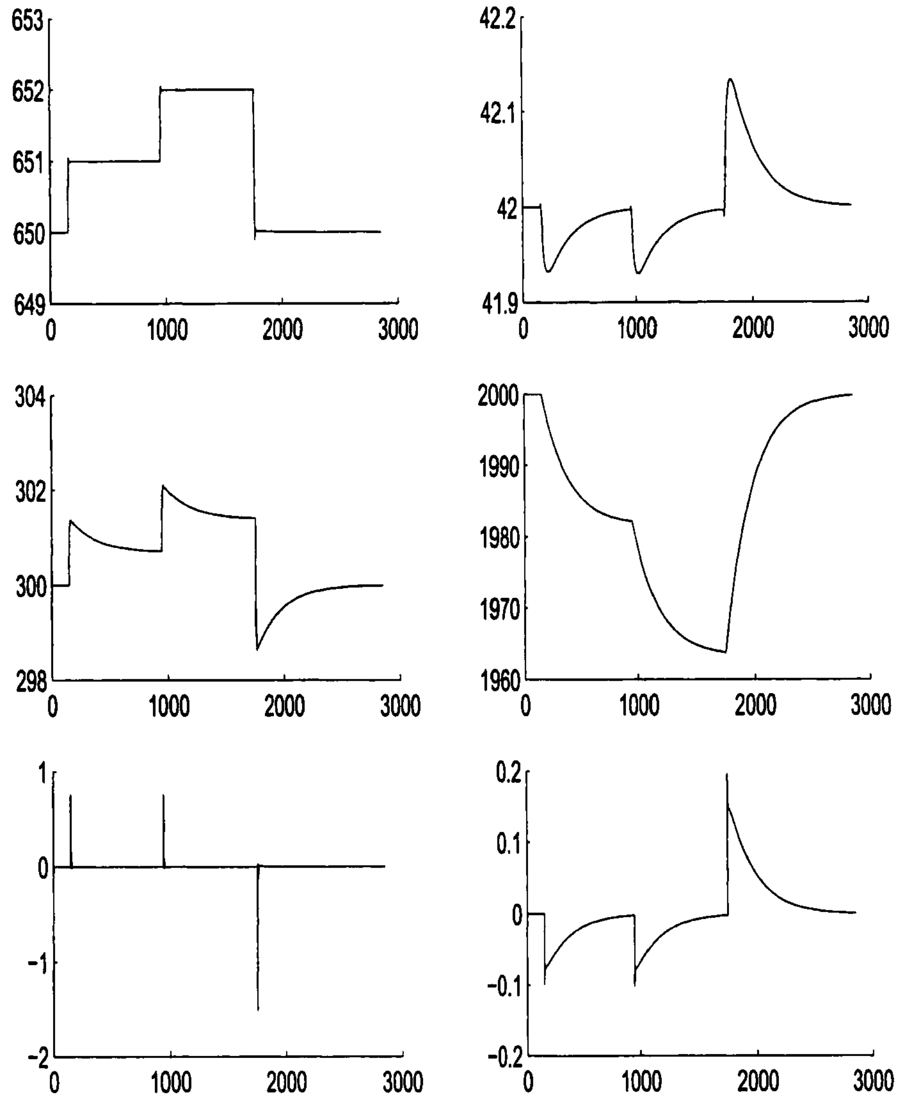

Figure 12: *Unconstrained Control. Closed-loop system response for setpoint change. Top-left:* y_1*; Top-right:* y_2*; Middle-left:* u_1*; Middle-right:* u_2*; Bottom-left:* Δu_1*; Bottom-right:* Δu_2

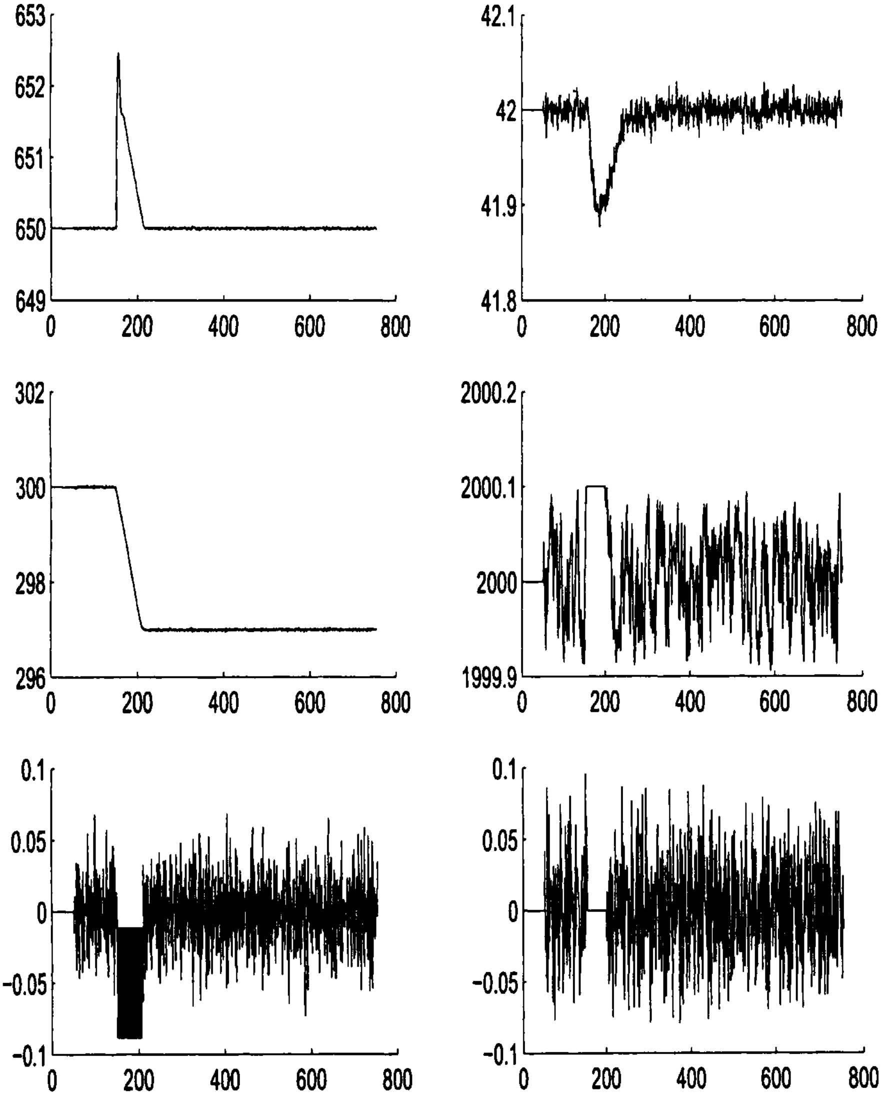

Figure 13: *Constrained Control with measurement noise. Closed-loop system response for input disturbance rejection. Top-left:* y_1*; Top-right:* y_2*; Middle-left:* u_1*; Middle-right:* u_2*; Bottom-left:* Δu_1*; Bottom-right:* Δu_2

References

1. Allgower, F., T.A. Badgwell, S.J. Qin, J.B. Rawlings and S.J. Wright. 1999. Nonlinear predictive control and moving horizon estimation- an introductory overview. *in Advances in Control: Highlights of ECC'99 (editor: Paul M. Frank), London, Springer* pp. 391–449.

2. Anderson, B. D.O. and J.M. Moore. 1971. *Linear Optimal Control.* Prentice-Hall, Hemel Hempstead.

3. Anderson, B. D.O. and J.M. Moore. 1979. *Optimal Filtering.* Prentice-Hall, Englewood Cliffs, New Jersey.

4. Astrom, K.J. and B.Wittenmark. 1984. *Computer Controlled Systems.* Prentice-Hall, Englewood Cliffs, N. J.

5. Beaufays, F. 1994. Transform- domain adaptive filters: an analytical approach. *IEEE Transactions on Signal Processing.*

6. Bemporad, A., M. Morari, V. Dua and E.N. Pistikopoulos. 1999. The explicit linear quadratic regulators for constrained control. *Technical report AUT99-16, ETH Zurich.*

7. Bitmead, R.R., M. Gevers and V. Wertz. 1990. *Adaptive Optimal Control the thinking man's GPC.* Prentice Hall, New York.

8. Camacho, E.F. and C. Bordons. 1999. *Model Predictive Control.* Springer, New York.

9. Clarke, D.W. and R. Scattolini. 1991. Constrained receding horizon predictive control. *Proc. IEE, PtD* **Vol.138**, 347–354.

10. Clarke, D.W., C. Mohtadi and P.S. Tuffs. 1987. Generalized predictive control.part 1: The basic algorithm. part 2: Extensions and interpretations. *Automatica* **Vol.23**, 137–160.

11. Cutler, C.R. and B.L. Ramaker. 1979. Dynamic matrix control-a computer control algorithm. *Presented at the Meeting of the American Institute of Chemical Engineers, Houston, Texas.*

12. Fletcher, R. 1981. *Practical Methods of Optimization, Volume 2, Constrained Optimization.* John Wiley and Sons, New York.

13. Franklin, G.F., J.D. Powell and A. Emami-Naeini. 1991. *Feedback Control of Dynamic Systems.* Addison-Welsley Publishing Company, Second Edition.

14. Garcia, C.E. and A.M. Morshedi. 1986. Quadratic programming solution to Dynamic Matrix Control (QDMC). *Chemical Engineering Communication* **46**, 73–87.

15. Garcia, C.E., D.M. Prett and M. Morari. 1989. Model predictive control: theory and practice - a survey. *Automatica* **Vol.25**, 335–348.

16. Goodwin, G.C., S. Graebe and M. Salgado. 2000. *Control System Design*. Prentice-Hall, Englewood Cliffs, NJ.

17. Gopal, V. and L.T. Biegler. 1998. Large scale inequality constrained optimization and control. *IEEE Control System Mag.* **18**, 59–68.

18. Grenander, U. and G. Szego. 1958. *Toeplitz Forms and Their Applications*. University of California Press, Berkeley and Los Angeles.

19. Hansson, A. 2000. A primal-dual interior-point method for robust optimal control of linear discrete-time systems. *IEEE transac. on Auto. Control.*

20. III, F. J. Doyle, J.F. Pekny, P. Dave and S. Bose. 1997. Specialized programming methods in the model predictive control of large scale systems. *Computers Chem. Engineering* **Vol.21**, 847–852.

21. Johnston, G. 1999. *Continuous time model predictive control of chemical process*. Final year thesis, Department of Electrical Engineering, University of Newcastle.

22. Kailath, T. 1980. *Linear Systems*. Prentice Hall, Inc., Englewood Cliffs, N.J.

23. Kleinman, B.L. 1970. An easy way to stabilize a linear constant system. *IEEE Trans. Automa. Control* **Vol.15**, 693.

24. Kwon, W.H., A.N. Bruckstein and T. Kailath. 1983. Stabilizing state feedback design via the moving horizon method. *Int. J. Control* **Vol. 37**, 631–643.

25. Kwon, W.H. and A.E. Pearson. 1977. A modified quadratic cost problem and feedback stabilization of a linear system. *IEEE Trans. Automat. Control* **Vol. AC-22**, 838–842.

26. Luenberger, D.G. 1969. *Optimization by Vector Space Methods*. John Wiley and Sons, New York.

27. Luenberger, D.G. 1984. *Linear and Nonlinear Programming*. Second edition, Addison-Wesley Publishing Company.

28. Maciejowski, J.M. 2002. *Predictive Control with Constraints*. Pearson Education Limited.

29. Marquis, P. and J.P. Broustail. 1988. AMOC, a bridge betweeen state space and model predictive controllers: Application to the automation of a hydrotreating unit. *Proceedings of the 1988 IFAC Workshop on Model Based Process Control, edited by T. J. McAvoy, Y. Arkun and E. Zafiriou, Pergamon Press.*

30. Mayne, D., J. Rawlings, C. Rao and P. Scokaert. 2000. Model predictive control: stability and optimality. *Automatica*.

31. Mayne, D. Q. and H. Michalska. 1990. Receding horizon control of nonlinear systems. *IEEE Trans. on Autom. Control.*

32. McGovern, L.K. and E. Feron 1999. Closed-loop stability of systems driven by real-time dynamic optimization algorithms. *38th Conference on Decision and Control, Phoenix, USA* pp. 3690–3696.

33. Morari, M. and J.H. Lee. 1991. Model predictive control: the good, the bad, and the ugly. *Proc. Fourth International Conference of Chemical Processes, Padre Island, Texas* pp. 419–444.

34. Morari, M. and J.H. Lee. 1999. Model predictive control: past, present and future. *Computers and Chemical Engineering* pp. 667–682.

35. Mosca, E. and J. Zhang. 1992. Stable redesign of predictive control. *Automatica* pp. 1229–1233.

36. Muske, K. and J.B. Rawlings. 1993*a*. Linear model predictive control of unstable processes. *Journal of Process Control.*

37. Muske, K. and J.B. Rawlings. 1993*b*. Model predictive control with linear models. *AIChE J.*

38. Peterka, V. 1984. Predictor-based self tuning control. *Automatica* **Vol.20**, 39–50.

39. Qin, S.J. and T.A. Badgwell. 1996. An overview of industrial model predictive control technology. *In Preprints of Chemical Process Control-CPC V, California.*

40. Rao, C.V., S.J. Wright and J.B. Rawlings. 1998. On the application of interior point methods to model predictive control. *Journal of Optimization Theory and Applications* **Vol.99**, 723–757.

41. Rawlings, J.B. 2000. Tutorial overview of model predictive control. *IEEE Control Systems Magazines* **Vol. 20**, 38–52.

42. Rawlings, J.B. and K.R. Muske. 1993. The stability of constrained receding horizon control. *IEEE Trans. Auto. Control* **Vol.38**, 1512–1516.

43. Richalet, J. 1993. Industrial applications of model based predictive control. *Automatica* **Vol.29**, 1251–1274.

44. Ricker, N.L. 1985. Use of quadratic programming for constrained internal model control. *Ind. Eng. Chem. Process Des. Dev.* **Vol.24**, 925–936.

45. Ricker, N.L. 1991. Model-predictive control: state of the art. *Proc. Fourth International Conference on Chemical Process Control, Padre Island, Texas* pp. 271–296.

46. Rossiter, J.A. and B.Kouvaritakis. 1993. Constrained stable generalised predictive control. *IEE Proceedings, Part D* **Vol. 140**, 243–254.

47. Rossiter, J.A., B.Kouvaritakis and M.J. Rice. 1998. A numerically stable robust state-space approach to stable-predictive control strategies. *Automatica* **Vol. 34**, 65–73.

48. Scokaert, P.O. and J.B. Rawlings. 1998. Constrained linear quadratic regulation. *IEEE Trans. Auto. Control.* **Vol. 43**, 1163–1169.

49. Seron, M.M., J.A. De Dona and G.C. Goodwin. 2000. Global analytical predictive control with input constraints. *39th Conference on Decision and Control.*

50. Shah, S.L. 1995. A tutorial introduction to constrained long range predictive control. *Pulp and Paper Canada* **Vol. 94**, 57–63.

51. Smith, Stephen. 2000. *Continuous time model predictive control of food cooking extruder.* Final year thesis, Department of Electrical and Computer Engineering, University of Newcastle, Australia.

52. Thomas, Y.A. 1975. Linear quadratic optimal estimation and control with receding horizon. *Electronics Letters* **Vol. 11**, 19–21.

53. Tsang, T. T.C. and D.W. Clarke. 1988. Generalized predictive control with input constraints. *Proc. IEE, Part D.* **Vol. 135**, 415–460.

54. van Donkelaar, E.T., O.H. Bosgra and P.M.J. Van den Hof. 1999. Constrained model predictive control with on-line input paramterization. In: *Proc. the 38th Conference on Decision and Control, Phoenix, Arizona.* pp. 3718–3721.

55. Viberg, M. 1995. Subspace-based methods for identification of linear time invariant systems. *Automatica* **Vol.31**, 1835–1851.

56. Wahlberg, B. 1991. System identification using Laguerre models. *IEEE Transactions on Automatic Control* **Vol.36**, 551–562.

57. Wang, L. 2001a. Discrete model predictive control design using laguerre functions. *Proceedings of American Control Conference.*

58. Wang, L. 2001b. Continuous time model predictive control using orthonormal functions. *International Journal of Control* **74**, 1588–1600.

59. Wang, L. 2001c. Use of exponential data weighting in model predictive control design. *Proceedings of the 40th IEEE Conference on Decision and Control.*

60. Wang, L. 2004. Discrete model predictive control design using laguerre functions. *Journal of Process Control* **14**, 131–142.

61. Wang, L. and P.J. Gawthrop. 2001. On the estimation of continuous time transfer functions. *International Journal of Control* **Vol. 74**, 889–904.

62. Wang, L. and W.R. Cluett. 2000. *From Plant Data to Process Control: Ideas for Process Identification and PID Design.* Taylor and Francis, London.

63. Wang, L., P. Gawthrop, C. Chessari and T. Podsiadley. 2003. Continuous time system identification of food extruder: experiment design and data analysis. *Proceedings of IFAC Symposium on System Identification, Holland.*

64. Zafiriou, E. 1991. On the effect of tuning parameters and constraints on the robustness of model predictive control. *Proc. Fourth International Conference on Chemical Process Control, Padre Island, Texas* pp. 363–393.

65. Zheng, A. 1999. Reducing on-line computational demands in model predictive control by approximating QP constraints. *Journal of Process Control* **Vol.9**, 279–290.