Analysis of the Convergence and Closed Loop Stability in EDMC

M. Haeri* and H. Zadehmorshed Beik

In this paper, the convergence and stability conditions of extended DMC in the control of nonlinear SISO and MIMO systems are investigated. The formulations are based on the ordinary DMC in which, with successive linearization of the nonlinear model and new interpretation of disturbance, the nonlinear extension is deduced. In addition, new convergence and stability criteria are derived for SISO and MIMO systems. These criteria include convergence and stability in the case of longer control \((M > 1)\) and prediction \((P > 1)\) horizons, as well as the finite and infinite sampling time. Finally, the simulation results for a MIMO \((3 \times 3)\) model, based on a power unit nonlinear plant, are presented.

INTRODUCTION

Model Predictive Control (MPC) and its industrial applications have become more and more popular during the last few decades \([1-3]\). MPC refers to a family of controllers that share three basic schemes \([4]\). In all model predictive controllers, an explicit model is employed to predict the future outputs of the process. This is why they are also categorized in model based control approaches. The control signal is determined, based on an optimization algorithm to optimize an objective function. This property enables the controllers to handle the constraints explicitly during the control design stage. Finally, the calculated control signals are applied, based on the receding horizon strategy, which is why they are also called receding horizon controllers.

So far, almost all types of model structure have been employed in MPC. Linear MPC refers to those controllers that use a linear model structure of the process. In the absence of constraints, use of linear models results in a closed form solution to the optimization problem. However, in constrained linear MPC, a quadratic programming is usually performed. In nonlinear MPC controllers, on the other hand, a nonlinear model of the process is employed. In this case, the optimization problem does not generally have a closed form solution. Usually, nonlinear programming is implemented to obtain results. Nonlinear programming is run, based on either a sequential \([5,6]\) or simultaneous \([7,8]\) strategy. In either case, high computational power is required \([9]\). This deficiency confines application of nonlinear MPC on processes that possess slow dynamics. In order to reduce computational requirements, linear approximations have been employed in some designs of nonlinear model predictive controllers. Multi Model adaptive Predictive Control (MMPC) \([10]\), Quadratic Dynamic Matrix Control (QDMC and NLQDMC) \([11,12]\), Extended Dynamic Matrix Control (EDMC) \([13,14]\) and Universal Dynamic Matrix Control (UDMC) \([15]\) have been introduced from this point of view. In the first three approaches, an approximated linear model is used to predict the effects of future control moves at each sample time. Free response, however, is computed by integrating the nonlinear model equations. While the linear models are determined off line in MMPC, in QDMC and EDMC they are obtained based on the Jacobian and the perturbation methods at each sample time. The main difference between QDMC and EDMC is the way of computation and consideration of the model/process mismatch and external disturbance. In UDMC, linear approximation is implemented only in the calculation of the Jacobians during the updating state of the optimization algorithm.

There is plenty of research work on the stability
consideration of linear [16-21] and nonlinear [22-23] MPC controllers. The backbone of ideas presented in the literature relies on the infiniteness of the prediction horizon and, therefore, use of the established stability property of the LQ control theory. Because of impracticality, in most cases, the infiniteness of the horizon is replaced by the stability constraints (including terminal constraints) in some improved schemes. Both hard [16,21] and soft constraints [26,27] were considered. By choosing the cost function in MPC as the candidate of the Lyapunov function and showing that it is a non-increasing function, the stability of the closed loop system is proved. The stability is derived for the nominal condition in most cases and is based on the state space model representation of the process, in which states are assumed to be measurable directly or determined through indirect measurements. It is also assumed that the implemented optimization algorithms are converged to the correct solution of the problem.

Due to their conceptual simplicity and lower computational requirements, predictive controllers, such as QDMC and EDMC, provide realistic advanced control strategies for industrial applications. It is shown in [12] that by tolerating a negligible loss in the closed loop performance, QDMC was six times faster than a nonlinear MPC run based on the sequential method. Unfortunately, there is not an analytical and explicit stability proof for QDMC or NLQDMC. Nevertheless, stable closed loop behavior has been illustrated in computer simulations, as well as actual implementation of the controllers, even for open loop integrating, as well as unstable processes [9,11,12]. To get a stable closed loop system, the state space representation of the process, along with a state estimator, has been employed in NLQDMC [12]. On the other hand, EDMC has proved to be stable, provided that some conditions on the control parameters and the steady state gain of the process are satisfied [13]. Proof of the convergence in the internal loop, as well as closed loop stability, was derived based on the contraction mapping theorem. Input/output representations of the process and the controller are used in the formulation of the problem. Imposing unrealistic conditions, such as infinite sample time and restricting assumptions, such as unity control and prediction horizons, make the mentioned proof unacceptable in practice. In this paper, the authors try to extend the applicability of the proof by relaxing the above mentioned restricting conditions. In this way, a nonlinear MPC (or time varying linear MPC) has been provided with less computational requirements, as well as acceptable and realistic stability conditions.

The paper is organized as follows. First, the DMC and its nonlinear extension are reviewed and nonlinear vector equations and some powerful solutions are presented. Then, the convergence and closed loop stability conditions for a SISO process case with \( M > 1 \), for finite and infinite sampling time, are proposed. After that, the EDMC formulation for the MIMO system is derived and the related convergence theorem and the stability criteria are presented, respectively. Finally, the simulation results for a nonlinear model \((3 \times 3)\) of a power unit plant are illustrated. Conclusions are given in the last section.

DMC FORMULATION AND ITS NONLINEAR EXTENSION FOR SISO SYSTEMS

Since EDMC formulation is based on ordinary DMC, a brief description of both controllers is presented in this section.

DMC

Considering a SISO system step response model, the output could be determined using the following discrete convolution:

\[
y^{\text{lin}}(k) = \sum_{i=1}^{N} a_i \Delta u(k-i) + a_N u(k-N-1) + d(k),
\]

(1)

where \( u(k) \) and \( \Delta u(k) \) are the input and its variation i.e., \( u(k)-u(k-1) \) in sample time \( k \), respectively. \( a_i \) is the step response coefficient at sample time \( i \). \( N \) is the number of sample time at which step response reaches its steady state. The following disturbance represents any difference between the measured output and the one predicted by the above model:

\[
d(k) = y^{\text{meas}}(k) - \sum_{i=1}^{N} a_i \Delta u(k-i) - a_N u(k-N-1).
\]

(2)

The future predictions of the process output are given in the following matrix-vector relation:

\[
\begin{bmatrix}
y^{\text{lin}}(k+1) \\
y^{\text{lin}}(k+2) \\
\vdots \\
y^{\text{lin}}(k+P)
\end{bmatrix} =
\begin{bmatrix}
a_1 & 0 & 0 & \ldots & 0 \\
a_2 & a_1 & 0 & \ldots & 0 \\
\vdots \\
a_M & a_M-1 & \ldots & a_1 \\
\vdots \\
a_P & a_P-1 & \ldots & a_{P-M+1}
\end{bmatrix}
\begin{bmatrix}
\Delta u(k) \\
\Delta u(k+1) \\
\vdots \\
\Delta u(k+M-1)
\end{bmatrix} +
\begin{bmatrix}
d(k) \\
\vdots \\
\vdots \\
d(k+P-1)
\end{bmatrix}
\]

\[
\begin{bmatrix}
y^{\text{lin}}(k+1) \\
y^{\text{lin}}(k+2) \\
\vdots \\
y^{\text{lin}}(k+P)
\end{bmatrix} =
\begin{bmatrix}
a_1 & 0 & 0 & \ldots & 0 \\
a_2 & a_1 & 0 & \ldots & 0 \\
\vdots \\
a_M & a_M-1 & \ldots & a_1 \\
\vdots \\
a_P & a_P-1 & \ldots & a_{P-M+1}
\end{bmatrix}
\begin{bmatrix}
\Delta u(k) \\
\Delta u(k+1) \\
\vdots \\
\Delta u(k+M-1)
\end{bmatrix} +
\begin{bmatrix}
d(k) \\
\vdots \\
\vdots \\
d(k+P-1)
\end{bmatrix}
\]
This can be written, equivalently, in the following vector form:
\[ y^{\text{lin}} = A \Delta u + y^{\text{past}} + d, \]  
(4)

\[ P \text{ and } M \text{ are the prediction and moving (control) horizons, respectively, } y^{\text{past}} \text{ denotes the effects of the past inputs on the predicted outputs. } A \text{ is a Toeplitz matrix, consisting of step response coefficients and is called the dynamic matrix of the process. Since future estimates of the mismatches are not available, it is customary to assume } d(k+i) = d(k) \text{ for } i = 1, 2, \ldots, P. \]

The control moves, \( \Delta u \), are determined according to the solution of the following optimization problem:
\[ \min_{\Delta u} \sum_{i=1}^{P} \left[ y^d(k+i) - y^{\text{lin}}(k+i) \right]^2 + \sum_{j=1}^{M} \lambda^2 \left[ \Delta u(k+M-j) \right]^2, \]  
(5)

\( y^d \) is the desired output trajectory and \( \lambda \) is the weighting coefficient on the input variations. Using least square estimation, the solution of the problem is given by:
\[ \Delta u = (A^T A + \lambda^2 I)^{-1} A^T (y^d - y^{\text{past}} - d). \]  
(6)

Usually, the first component of \( \Delta u \), i.e. \( \Delta u(k) \), will be applied to the process \( u(k) = u(k-1) + \Delta u(k) \) and the same procedure will be performed in future sampling intervals.

**Extended DMC**

To extend the application of DMC (which is originally based on a linear model of the process) to nonlinear systems, it is required to employ an approximated linear model for a nonlinear process in each sample interval. This is done via linearization of the process’s nonlinear model or by determination of the process response for a step perturbation. Also, a new interpretation of disturbance is introduced, due to the nonlinear character of the process. This is explained in more detail in the next section.

**Nonlinear Disturbance**

In this extension, a new interpretation of \( d \) is exploited [13]. In other words, \( d \) is split into two parts, the unknown parts, \( d^{\text{nl}} \), which are treated as in ordinary DMC and the known parts, \( d^{\text{el}} \), which represent the difference between approximated linear and nonlinear models of the process.

\[ d = d^{\text{ext}} + d^{\text{nl}}. \]  
(7)

To consider this partitioning, the predicted outputs of the process are written as:
\[ y^{\text{el}} = A \Delta u + y^{\text{past}} + d^{\text{ext}} + d^{\text{nl}}, \]  
(8)

\( d^{\text{ext}}(k+i) \) is assumed constant over the prediction horizon \( (i = 1, 2, \ldots, P) \) and \( d^{\text{nl}}(k+i) \) varies during the horizon. Solving Problem 5 results in the following relation for input variations:
\[ \Delta u(d^{\text{nl}}) = (A^T A + \lambda^2 I)^{-1} A^T (y^{sp} - y^{\text{past}} - d^{\text{ext}} - d^{\text{nl}}), \]  
(9)

\( d^{\text{nl}} \) is determined, in order to have the same predicted outputs from the nonlinear, \( y^{\text{nl}} \), and linear, \( y^{\text{el}} \), models.

\[ y^{\text{nl}}(d^{\text{nl}}) = y^{\text{el}} = A \Delta u(d^{\text{nl}}) + y^{\text{past}} + d^{\text{ext}} + d^{\text{nl}}. \]  
(10)

To solve the above problem, equation 10 is reformulated in the following equation, which is a root finding problem:
\[ f(d^{\text{nl}}) = y^{\text{nl}}(d^{\text{nl}}) - A \Delta u(d^{\text{nl}}) - y^{\text{past}} - d^{\text{nl}} - d^{\text{ext}} = 0. \]  
(11)

There are \( P \) nonlinear equations in \( P \) unknowns, \( d^{\text{nl}}(k+1), \ldots, d^{\text{nl}}(k+P) \). Nonlinearity of the equations arises from the nonlinear relation that exists between \( y^{\text{nl}} \) and \( u \) (and, therefore, \( \Delta u \)). In the following sub-section, some well established methods are summarized that can be applied to find the solution to the problem. One of the simplest methods is successive substitution via a fixed-point algorithm,
\[ d^{\text{nl}}_{k+1} = f(d^{\text{nl}}_k) = d^{\text{nl}}_k + \beta (y^{\text{nl}}_k - y^{\text{el}}_k) = d^{\text{nl}}_k + \beta f(d^{\text{nl}}_k). \]  
(12)

A block diagram of EDMC in Figure 1 shows the internal iteration and external DMC closed loop after convergence.
In Quasi Newton (QN) methods, such as Brodyen's method [29], the following steps are performed in each iteration: to obtain a good approximation of the Jacobian matrix and benefit from a convergence rate of one. Iteration steps are as follows:

\[ x_{k+1} = x_k - \{f'(x_k)\}^{-1}f(x_k). \]  

This method benefits from a convergence rate of two around the optimal solution. However, it requires a Jacobian matrix that is not easily available in most applications, like Problem 11. A simplified version of the Newton method is obtained by substituting \( f'(x) \) with a fixed matrix, \( C \). Sometimes, the initial value of the Jacobian matrix, \( f'(x_0) \), is used in this regard:

\[ x_{k+1} = x_k - C^{-1}f(x_k) = x_k - f'(x)^{-1}f(x_k). \]  

The fixed-point iteration method is obtained by replacing \( -C^{-1} \) with \( \beta I \), in which \( \beta \) is a small positive coefficient:

\[ x_{k+1} = x_k + \beta f(x_k), \quad \beta \in (0, 1). \]  

This method requires the least computational effort and benefits from a convergence rate of one. Iteration methods given in Equations 12 and 14 have been used in [13] and some good results have been obtained for certain conditions. In Quasi Newton (QN) methods, some approximation of the Jacobian matrix is employed. In Brodyen's method [29], the following steps are performed in each iteration:

\[ x_{k+1} = x_k + s_k, \]

\[ y_k = f(x_{k+1}) - f(x_k), \]

\[ w_k s_k = -f(x_k), \]

\[ w_{k+1} = w_k + \frac{(y_k - w_k s_k) s_k^T}{s_k^T s_k}, \]  

where \( w_k \) is an approximate for \( f'(x_k) \) and initialized by \( w_0 = f'(x_0) \). This method converges to the solution at an approximate rate of 1.6, which means super linear convergence. Some other methods of the QN family, such as Greenstadt, Barnes and Thomas, can be found in [30].

The two following subsections describe the contribution made by the present work. This includes an extension of the work in [13] to higher \( M \) and MIMO systems.

Convergence for SISO Systems with \( M > 1 \)

As in [13], convergence of the fixed-point iterations can be proved via the contraction-mapping theorem. It is shown that for a globally asymptotically stable open loop system, iterations in Equations 9 and 12 will converge if the sampling time and weighting factor are large enough and the relaxation factor, \( \beta \), is small enough. Since in the above-mentioned conditions, assumption \( M = 1 \) is not used, the statement is also valid for \( M > 1 \). With some mathematical manipulation of the results given in [13], it is shown that the following condition on the weighting factor (\( \lambda \)) is required to get the convergence:

\[ \lambda^2 > 2MP \max_i \{|a_i| \sigma_i^2|\}, \]  

in which \( a_i \) and \( \sigma_i^2 \) are the steady state gain of the linear and nonlinear models at the \( i \)th iteration. Since \( M \) is allowed to be higher than one, more performance improvement can be expected from the controller.

Stability for SISO Systems with \( M > 1 \)

The operator theory and contraction mapping [31] can be used to show the closed loop stability of the system for \( M > 1 \) and infinite/infinite sampling time (\( T \)) as follows.

**Infinite Sampling Time**

In this section, it is required to extend the stability criteria of an SISO system for a \( M > 1 \) case. Regarding stability, it is assumed that the iterative computation of \( d_{\text{in}} \) in sample time \( k \) has been converged and the goal is to derive a relation between present \( u_k \) and previous \( u_{k-1} \) inputs.

\[ u_k = u_{k-1} + \Delta u_k \]

\[ = u_{k-1} + e_i^T [(A^T A + \lambda^2 I)^{-1} A^T (y_{k+1}^{\text{ext}} - y_{k+1}^{\text{ref}} - d_{\text{in}}^{\text{est}})] \]

\[ e_i \] is a \( M \times 1 \) vector with all elements zero, except the \( i \)th element, which is one. Note that for a nominal system, \( d_{\text{ext}} = 0 \). To complete and simplify Equation 18, the converged value of \( d_{\text{in}}^{\text{est}} \) should be
determined. When the convergence has been reached, the following equality is satisfied:

\[ y_d = y^e, \quad y_d = [y^{nl}(k+1) \cdots y^{nl}(k+P)]^T, \]

and:

\[ y^e = [y^{nl}(k+1) \cdots y^{nl}(k+P)]^T. \]

The output of the extended linear model, \( y^{nl}_k \), is obtained as follows:

\[ y^{nl}_{k+1} = y^{past}_{k+1} + A \Delta u_k + d^{nl}_{k+1} \]

\[ = y^{past}_{k+1} + A (A^T A + \lambda^2 I)^{-1} \]

\[ A^T (y^{sp}_{k+1} - y^{past}_{k+1} - d^{nl}_{k+1}) + d^{nl}_{k+1}. \] (20)

Let one define:

\[ A_0 = A (A^T A + \lambda^2 I)^{-1} A^T. \]

Using Equations 19 and 20 and the above definition, one can derive the following relation:

\[ (I - A_0) d^{nl}_{k+1} = y^{nl}_{k+1} - (I - A_0) y^{past}_{k+1} - A_0 y^{sp}_{k+1}, \]

or:

\[ d^{nl}_{k+1} = (I - A_0)^{-1} y^{nl}_{k+1} - y^{past}_{k+1} - (I - A_0)^{-1} A_0 y^{sp}_{k+1}. \] (21)

Substituting \( d^{nl}_{k} \) from Equation 21 in Equation 18, one obtains:

\[ u_k = u_{k-1} + \Delta u_k \]

\[ = u_{k-1} + e_1^T [(A^T A + \lambda^2 I)^{-1} \]

\[ A^T (y^{sp}_{k+1} - y^{past}_{k+1} - (I - A_0)^{-1} A_0 y^{sp}_{k+1} \]

\[ + (I - A_0)^{-1} A_0 y^{sp}_{k+1})]. \] (22)

Using the following two matrix relationships for \( A_0 \):

\[ (I - A_0)^{-1} = I + \frac{1}{\lambda^2} A A^T, \]

\[ (I - A_0)^{-1} A_0 = \frac{1}{\lambda^2} A A^T. \] (23)

Equation 22 is simplified as:

\[ u_k = u_{k-1} + e_1^T (A^T A + \lambda^2 I)^{-1} \]

\[ A^T (I + \frac{1}{\lambda^2} A A^T) (y^{sp}_{k+1} - y^{nl}_{k+1}) \]

\[ = u_{k-1} + \frac{1}{\lambda^2} e_1^T A^T (y^{sp}_{k+1} - y^{nl}_{k+1}). \] (24)

When considering the infinite sampling time assumption \((T \to \infty)\), one can use the following definitions:

\[ y^{nl}_{k+1} = y^{nl}(k+1) I_p, \quad y^{sp}_{k+1} = y^{sp} I_p, \]

\[ A = a I_L, \quad e_1^T A^T I_p = p A, \]

where:

\[ I_p = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}, \quad I_L = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \cdots & 1 \end{bmatrix}_{p \times M}. \] (26)

Therefore, Equation 24 is further simplified as:

\[ u_k = u_{k-1} + \frac{a P}{\lambda^2} (y^{sp} - y^{nl}(k+1)). \] (27)

This equation is similar to the one derived in [13] but assumption \( M = 1 \) is not used in this case. Therefore, results given in [13] for the stability of the closed loop system are also applicable to \( M > 1 \). This results in the following theorem.

**Nominal Stability Theorem**

Suppose that the system to be controlled is globally asymptotically stable for all feasible inputs and, furthermore, suppose that the following is valid:

1. The steady state gain of the system does not change sign;
2. The weight on the change of the input is larger than zero;
3. The sampling time is long enough \((T \to \infty)\);
4. The set point is constant in the prediction horizon.

Then, the closed loop system is guaranteed to be nominally stable.

**Finite Sampling Time**

In this section, one considers relaxing assumption 3 in the above stability theory. According to the previous section, this condition is employed in two cases, first in definitions given in Equation 25 and second, in deriving the Jacobian of the input in Equation 27. In the following lines, analysis of the stability is continued without considering assumption 3. Recall the relation for \( u_k \) in Equation 24.

\[ u_k = u_{k-1} + \frac{1}{\lambda^2} e_1^T A^T (y^{sp}_{k+1} - y^{nl}_{k+1}). \] (28)
It is assumed that the internal iteration in EDMC has been converged. This relation is expanded using elements of \(e_1, A, y_{k+1}^p\) and \(y_{k+1}^p\):

\[
u_k = u_{k-1} + \frac{1}{N} \sum_{i=1}^{N} a_i (y_{k+1}^p(k + i) - y_{k+1}^p(k + i)). \tag{29}
\]

To derive the Jacobian of \(\frac{\partial u_k}{\partial u_{k-1}}\), derivatives of the future outputs, with respect to \(u_{k-1}\), are required.

\[
\frac{\partial u_k}{\partial u_{k-1}} = 1 - \frac{1}{N} \sum_{i=1}^{N} \frac{\partial y_{k+1}^p(k + i)}{\partial u_{k-1}} + a_2 \frac{\partial y_{k+1}^p(k + 2)}{\partial u_{k-1}} + \cdots + a_P \frac{\partial y_{k+1}^p(k + P)}{\partial u_{k-1}}.
\]

(30)

To solve the problem, the linear approximations of \(y_{k+1}^p(k + i)\) are used in the computation:

\[
g(k+1) = a_1 u(k) + (a_2 - a_1) u(k-1) + (a_3 - a_2) u(k-2) + \cdots + (a_N - a_{N-1}) u(k - N + 1),
\]

\[
g(k+2) = a_1 u(k+1) + (a_2 - a_1) u(k) + (a_3 - a_2) u(k-1) + \cdots + (a_N - a_{N-1}) u(k - N + 2),
\]

\[
\vdots
\]

\[
g(k + M) = a_1 u(k + M - 1) + (a_2 - a_1) u(k + M - 2) + \cdots + (a_M - a_{M-1}) u(k - 1) + (a_N - a_{N-1}) u(k - N + M),
\]

\[
g(k + M + 1) = a_2 u(k + M - 1) + (a_3 - a_2) u(k + M - 2) + \cdots + (a_{M+2} - a_{M+1}) u(k - 1) + (a_N - a_{N-1}) u(k - N + M + 1),
\]

\[
\vdots
\]

\[
g(k + P + 1) = a_{P+1} u(k + M - 1) + (a_{P+2} - a_{P+1}) u(k + M - 2) + \cdots + a_P u(k - 1) + \cdots + (a_N - a_{N-1}) u(k - N + P). \tag{31}
\]

Let the following definitions be used:

\[
A = [a_1 \ a_2 \ \cdots \ a_M],
\]

and:

\[
z_1 = \frac{\partial u_k}{\partial u_{k-1}}, \quad z_2 = \frac{\partial u_{k+1}}{\partial u_{k-1}}, \quad \ldots, \quad z_M = \frac{\partial u_{k+M-1}}{\partial u_{k-1}} \tag{32}
\]

Therefore, \(z_i\) are determined as follows:

\[
z_1 = 1 - \frac{1}{N} \alpha^T_{11}, \quad z_2 = 1 - \frac{1}{N} \alpha^T_{12}, \quad \ldots, \quad z_M = 1 - \frac{1}{N} \alpha^T_{1M} \tag{33}
\]

\(l\) is a column vector given as:

\[
[l_1 \ l_2 \ \cdots \ l_M] =
\begin{bmatrix}
a_1 z_1 + a_2 - a_1 & a_1 z_2 + (a_2 - a_1) z_1 + a_3 - a_2 & \cdots \\
a_2 z_M + (a_2 - a_1) z_{M-1} & a_2 z_{M+1} + (a_3 - a_2) z_{M-1} & \cdots \\
& \ddots & \ddots & \ddots
\end{bmatrix}
\]

\[
l =
\begin{bmatrix}
1 \\
\vdots \\
1
\end{bmatrix}
\]

(34)

Substitute for \(z_i\) in Equation 34 from Equations 33 and, with some manipulation, \(l\) is reformulated as follows:

\[
l = [a_2 - \frac{1}{N} a_1 \alpha^T_1 \ 1 \\
a_3 - \frac{1}{N} (a_2 \alpha^T_2 + a_1 \alpha^T_1) \ 1 \\
\vdots \\
a_{P+1} - \frac{1}{N} (a_P \alpha^T_P + \cdots + a_M \alpha^T_M) \ 1]
\]

(35)

Or, in simple form, it can be written as:

\[
l = \hat{B} - \frac{1}{N^2} \hat{B} l,
\]

where:

\[
\hat{B} =
\begin{bmatrix}
a_2 \\
a_3 \\
\vdots \\
a_{P+1}
\end{bmatrix},
\]

\[
\hat{B} =
\begin{bmatrix}
a_1 \alpha^T_1 \\
a_2 \alpha^T_2 + a_1 \alpha^T_1 \\
a_M \alpha^T_M + \cdots + a_1 \alpha^T_M \\
\vdots \\
a_P \alpha^T_P + \cdots + a_M \alpha^T_M
\end{bmatrix}.
\]

(36)
Equation 36 can be solved for $l$:

$$ l = (I + \frac{1}{\lambda}AA^T)^{-1}b. \quad (38) $$

Using the definition of $A_0$ in the previous section, since $(I + \frac{1}{\lambda}AA^T)^{-1} = I - A_0$, therefore:

$$ l = (I - A_0)b = \hat{b} - A_0\hat{b}. \quad (39) $$

From the first row of Equation 34, $z_1$ is determined as follows:

$$ z_1 = \frac{1}{a_1}(l_1 + a_1 - a_2) $$

$$ = \frac{1}{a_1}(e_1^Tl + a_1 - a_2) $$

$$ = \frac{1}{a_1}(e_1^Tb - e_1^TA_0b + a_1 - a_2) $$

$$ = \frac{1}{a_1}(a_1 - e_1^TA_0b). \quad (40) $$

Based on the contraction mapping theorem, closed loop stability requires $z_1 < 1$. In other words:

$$ \frac{1}{a_1}e_1^TA_0b > 0. \quad (41) $$

**Some Special Cases for Stability**

1. $P = M = 1$,

$$ A_0 = a_1(a_1^2 + \lambda^2)^{-1}a_1, \quad b = a_2, $$

$$ \frac{1}{a_1}e_1^TA_0b = \frac{1}{a_1}a_2^2 + \lambda^2a_2 = \frac{a_1a_2}{a_1^2 + \lambda^2} > 0 $$

$$ \Rightarrow a_1a_2 > 0. \quad (42) $$

2. $M = 1, P > 1$,

$$ A = [a_1a_2 \cdots a_P]^T, $$

$$ \frac{1}{a_1}e_1^TA_0b = \frac{A^Tb}{AA^T + \lambda^2} > 0 $$

$$ \Rightarrow A^Tb > 0. \quad (43) $$

Or, equivalently,

$$ a_1a_2 + a_2a_3 + \cdots + a_pa_{p+1} = \sum_{i=1}^{p} a_ia_{i+1} > 0. \quad (44) $$

3. $M = 2, P > 2$

$$ A^T A + \lambda^2 I = \begin{bmatrix} \sum_{i=1}^{p} a_i^2 + \lambda^2 & \sum_{i=1}^{p-1} a_ia_{i+1} \\ \sum_{i=1}^{p-1} a_ia_{i+1} & \sum_{i=1}^{p-1} a_i^2 + \lambda^2 \end{bmatrix}. \quad (45) $$

$$ \frac{1}{a_1}e_1^TA_0b = \frac{[\sum_{i=1}^{p-1} a_i^2 + \lambda^2 - \sum_{i=1}^{p-1} a_ia_{i+1}] [\sum_{i=1}^{p-1} a_ia_{i+1}] - [\sum_{i=1}^{p-1} a_ia_{i+2}]}{\det(A^TA + \lambda^2 I)} $$

$$ = \frac{[(\sum_{i=1}^{p-1} a_i^2 + \lambda^2)(\sum_{i=1}^{p-1} a_ia_{i+1}) - (\sum_{i=1}^{p-1} a_ia_{i+1})(\sum_{i=1}^{p-1} a_ia_{i+2})]}{[(\sum_{i=1}^{p-1} a_i^2 + \lambda^2)(\sum_{i=1}^{p-1} a_i^2 + \lambda^2) - (\sum_{i=1}^{p-1} a_ia_{i+1})^2]} \quad (46) $$

**DMC FORMULATION FOR LINEAR MIMO SYSTEMS**

For the sake of simplicity, formulations are given for $2 \times 2$ systems. The same line of calculations is used to determine formulas for $n \times n$ systems. Outputs of a stable linear time invariant $2 \times 2$ system can be represented, using its step response model, by the following equations:

$$ y_1(k) = \sum_{i=1}^{N} a_i \Delta u_1(k-i) + a_N u_1(k-N-1) $$

$$ + \sum_{i=1}^{N} b_i \Delta u_2(k-i) + b_N u_2(k-N-1) + d_1(k), $$

$$ y_2(k) = \sum_{i=1}^{N} c_i \Delta u_1(k-i) + c_N u_1(k-N-1) $$

$$ + \sum_{i=1}^{N} d_i \Delta u_2(k-i) + d_N u_2(k-N-1) + d_2(k), \quad (47) $$

where $u_i(k)$ and $\Delta u_i(k)$ are the $i$th input and its variation in sample time $k$. $a_i, b_i, c_i$ and $d_i$ are the step response coefficients at sample time $i$. $N$ is the sample time at which all the step responses reach their steady state. $d_i$ stands for any differences between the system output and the one predicted by the step response model. These errors account for model/system mismatches and external disturbances. Future predictions of the system outputs for prediction horizon $P$, based on control horizon $M$, are given in
the following matrix-vector relation:

$$
\begin{bmatrix}
    y_1(k + 1) \\
    \vdots \\
    y_1(k + P) \\
    \vdots \\
    y_2(k + 1) \\
    \vdots \\
    y_2(k + P)
\end{bmatrix} =
\begin{bmatrix}
    a_1 & 0 & \cdots & 0 & b_1 & 0 & \cdots & 0 \\
    a_2 & a_1 & \cdots & 0 & b_2 & b_1 & \cdots & 0 \\
    \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
    c_P & a_{P-1} & \cdots & a_1 & d_P & d_{P-1} & \cdots & d_1
\end{bmatrix}
\begin{bmatrix}
    \Delta u_1(k) \\
    \vdots \\
    \Delta u_1(k + M - 1) \\
    \vdots \\
    \Delta u_2(k) \\
    \vdots \\
    \Delta u_2(k + M - 1)
\end{bmatrix} +
\begin{bmatrix}
    a_{N+1} & b_2 & b_3 & \cdots & b_{N+1} \\
    0 & b_3 & b_4 & \cdots & b_{N+1} & 0 \\
    \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
    0 & b_{P+1} & \cdots & b_{N+1} & 0 \\
    c_{N+1} & d_3 & d_4 & \cdots & d_{N+1} & 0 \\
    0 & d_4 & d_4 & \cdots & d_{N+1} & 0 \\
    0 & d_{P+1} & \cdots & d_{N+1} & 0
\end{bmatrix}
\begin{bmatrix}
    \Delta u_1(k - 1) \\
    \Delta u_1(k - 2) \\
    \vdots \\
    \Delta u_1(k + N - 1) \\
    \Delta u_2(k - 1) \\
    \Delta u_2(k - 2) \\
    \vdots \\
    \Delta u_2(k + N - 1)
\end{bmatrix}
\begin{bmatrix}
    a_{N+1} & u_1(k - N) + b_{N+1} & u_2(k - N) \\
    a_{N+1} & u_1(k - N + 1) + b_{N+1} & u_2(k - N + 1) \\
    \vdots & \vdots & \vdots \\
    a_{N+1} & u_1(k - N + P - 1) + b_{N+1} & u_2(k - N + P - 1) \\
    c_{N+1} & u_1(k - N) + d_{N+1} & u_2(k - N) \\
    c_{N+1} & u_1(k - N + 1) + d_{N+1} & u_2(k - N + 1) \\
    \vdots & \vdots & \vdots \\
    c_{N+1} & u_1(k - N + P - 1) + d_{N+1} & u_2(k - N + P - 1)
\end{bmatrix}
\begin{bmatrix}
    d_1(k + 1) \\
    d_1(k + 2) \\
    \vdots \\
    d_1(k + P) \\
    d_2(k + 1) \\
    d_2(k + 2) \\
    \vdots \\
    d_2(k + P)
\end{bmatrix}
\]
one. Doing some simplifications, one can derive the following results (details of similar computations for SISO systems are given in [13]):

\[
\begin{align*}
\frac{\partial f_1}{\partial d^{nl}} &= I + \beta(\frac{\partial y^{nl}}{\partial d^{nl}} - \frac{\partial y^{el}}{\partial d^{el}}) \\
&= I + \beta(\frac{\partial y^{nl}}{\partial d^{nl}} \frac{\partial \Delta u}{\partial d^{nl}} - A \frac{\partial \Delta u}{\partial d^{nl}} - I) \\
&= (1 - \beta)I + \beta(B - A)'\frac{\partial \Delta u}{\partial d^{nl}} \\
&= I - \beta(B - A)(A^T A + \lambda^2 I)^{-1} A^T \\
&= (1 - \beta)I - \beta(B^T - A)(L^T A^T A I + \lambda^2 I)^{-1} L^T A^T \\
&= (1 - \beta)I - \beta(B^T - A)A L (L^T A^T A L + \lambda^2 I)^{-1} L^T A^T. \\
\end{align*}
\]

(56)

Using the matrix inversion lemma given in Equation 57, this relation is rearranged as in Equation 58,

\[
u(I + u^T u)^{-1} u^T = uu^T(I + uu^T)^{-1},
\]

(57)

\[
\frac{\partial f_1}{\partial d^{nl}} = (1 - \beta)I - \beta(B^T - A)A L (L^T A^T A L + \lambda^2 I)^{-1} L^T A^T.
\]

(58)

Using the following property:

\[
\frac{\partial f_1}{\partial d^{nl}} = \frac{\partial f_1}{\partial d^{nl}}\frac{\partial y^{nl}}{\partial d^{nl}} - A \frac{\partial \Delta u}{\partial d^{nl}} - I
\]

(59)

It is further simplified as:

\[
\frac{\partial f_1}{\partial d^{nl}} = \frac{\partial f_1}{\partial d^{nl}}(1 - \beta)I - \beta(B^T - A)A L (L^T A^T A L + \lambda^2 I)^{-1} L^T A^T.
\]

(60)

The maximum singular value of the matrix is considered as its norm here. Therefore,\[
\sigma^*(\frac{\partial f_1}{\partial d^{nl}}) \leq 1 - \beta
\]

(61)

By selecting a large \( \lambda \) and a small \( \beta \), the maximum singular value of \( \frac{\partial f_1}{\partial d^{nl}} \) would be less than 1 and, therefore, convergence of the fixed-point iterations in Equation 12 will be guaranteed. For some simplification, the following approximation can be used:

\[
\sigma^*(L) = 2 \times 2 \times MP = 4MP,
\]

\[
\sigma_s(L) \in (0.25, 0.5).
\]

(62)

**Stability for MIMO Systems (with Infinite Sampling Time Assumption)**

Stability of a closed loop \( 2 \times 2 \) MIMO system is guaranteed under above convergence conditions and the positive definiteness of \( D_0 = \begin{bmatrix} a & c \\ b & d \end{bmatrix} \), which is stated in Theorem 2.

**Theorem 2**

Suppose that the nonlinear \( 2 \times 2 \) MIMO system to be controlled is globally asymptotically stable for all feasible inputs, the weight on the change of the inputs is larger than zero, the sampling time is long enough and the steady state gain, \( G \), satisfies the following criteria, then the closed loop system is guaranteed to be nominally stable.

\[
D_0 = \begin{bmatrix} a & c \\ b & d \end{bmatrix} G \succ 0.
\]

(63)

**Proof**

As in [33], closed loop stability can be investigated via computing singular values of the derivative of the nonlinear operator, \( N \), that is defined as:

\[
u(k) = N(u(k - 1)).
\]

(64)

To determine \( N \), one starts from Equation 52. It is assumed that \( d^\text{ext}_k = 0 \) for a nominal system.

\[
u_k = u_{k-1} + e^T_1 (A^T A + \lambda^2 I)^{-1}
\]

\[
A^T (\Pi y^\text{sp}_{k+1} - \Pi y^\text{spar}_{k+1} - \Pi d^{nl}_{k+1}),
\]

(65)

where:

\[
e^T_1 = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 1 & 0 & \cdots & 0 \end{bmatrix}_{2 \times 2M},
\]

and:

\[
\Pi = \begin{bmatrix} 1_p & 0 \\ 0 & 1_p \end{bmatrix}_{2 \times 2p}.
\]

The convergent value of \( d^{nl}_{k+1} \) is calculated in a similar way to that found in the section of Infinite Sampling Time and, then, it is replaced in Equation 65.

\[
\Pi d^{nl}_{k+1} = (I - A_0)^{-1} \Pi y^\text{nl}_{k+1} - \Pi y^\text{spar}_{k+1} - (I - A_0)^{-1} A_0 \Pi y^\text{sp}_{k+1}.
\]

(66)

\[
y^\text{nl}_{k+1} = \begin{bmatrix} y^\text{nl}_{1}(k + 1) \\ y^\text{nl}_{2}(k + 1) \end{bmatrix},
\]

(67)

\[
y^\text{sp}_{k+1} = \begin{bmatrix} y^\text{sp}_{1}(k + 1) \\ y^\text{sp}_{2}(k + 1) \end{bmatrix}.
\]
Equation 66 is simplified as:
\[
\Pi d_{k+1}^{nl} = (\mathbf{I} + \frac{1}{\lambda^2} \mathbf{A} \mathbf{A}^T) \Pi y_{k+1}^{nl} - \frac{1}{\lambda^2} \mathbf{A} \mathbf{A}^T y_{k+1}^{sp} - \Pi y_{k+1}^{past}.
\] (68)

Substituting \( \Pi d_{k+1}^{nl} \) into Equation 65 results in the following relation.
\[
u_k = u_{k-1} + e^T \left( \mathbf{A}^T \mathbf{A} + \lambda^2 \mathbf{I} \right)^{-1} \mathbf{A}^T \left( \Pi y_{k+1}^{sp} - \Pi y_{k+1}^{past} \right)
- \left( \mathbf{I} + \frac{1}{\lambda^2} \mathbf{A} \mathbf{A}^T \right) \Pi y_{k+1}^{nl} + \frac{1}{\lambda^2} \mathbf{A} \mathbf{A}^T y_{k+1}^{sp} + \Pi y_{k+1}^{past}.
\] (69)

This relation is further simplified as:
\[
u_k = u_{k-1} + \frac{1}{\lambda^2} e^T \mathbf{A}^T \Pi (y_{k+1}^{sp} - y_{k+1}^{nl}).
\] (70)

Relations obtained so far are based on open loop stability and \( T \rightarrow \infty \) conditions. It can be shown that:
\[
e^T \mathbf{A}^T \Pi = \begin{bmatrix} a & c \\ b & d \end{bmatrix} P = \hat{A} P.
\] (71)

Then, \( u_k \) is:
\[
u_k = u_{k-1} - \frac{P}{\lambda^2} \begin{bmatrix} a & c \\ b & d \end{bmatrix} y_{k+1}^{nl} + \frac{P}{\lambda^2} \begin{bmatrix} a & c \\ b & d \end{bmatrix} y_{k+1}^{sp}
- \left( \mathbf{I} + \frac{1}{\lambda^2} \hat{A} \mathbf{G} \right) u_k = u_{k-1} + \frac{P}{\lambda^2} \hat{A} y_{k+1}^{sp},
\] (72)
or:
\[
u_k = \left( \mathbf{I} + \frac{P}{\lambda^2} \hat{A} \mathbf{G} \right)^{-1} u_{k-1} + \left( \mathbf{I} + \frac{P}{\lambda^2} \hat{A} \mathbf{G} \right)^{-1} \frac{P}{\lambda^2} \hat{A} y_{k+1}^{sp}.
\] (73)

Now, one can determine \( \mathbf{N}' \) from Equation 73.
\[
\mathbf{N}' = \frac{\partial \mathbf{u}_k}{\partial u_{k-1}} = \left( \mathbf{I} + \frac{P}{\lambda^2} \hat{A} \mathbf{G} \right)^{-1}.
\] (74)

It can be seen that when the matrix:
\[
\mathbf{D}_0 = \begin{bmatrix} a & c \\ b & d \end{bmatrix} \mathbf{G},
\] (75)
is positive definite, the closed loop system is stable. In the case of SISO systems, \( \mathbf{D}_0 \) is reduced to \( a g \), which implies that sign changes in the steady state gain could result in the instability of the closed loop system. This is the same result that is given in the nominal stability theorem [13].

**SIMULATION RESULT: POWER UNIT NONLINEAR MODEL**

A power unit is simulated, along with a nonlinear dynamic model of an 160 MW oil fired drum boiler-turbine-generator unit, intended for overall wide range simulation [34-36]. This model represents a three-input, three-output, third-order nonlinear system. The inputs are the position of the valve actuators that control fuel mass flow rate \( (u_1 \text{ in pu}) \), steam flow rate \( (u_2 \text{ in pu}) \) and water flow rate \( (u_3 \text{ in pu}) \). The outputs are the electric power \( (P_e \text{ in MW}) \), drum steam pressure \( (P_r \text{ in kg/cm}^2) \) and drum water level \( (L \text{ in m}) \). The state variables are electric power, drum steam pressure and fluid (steam-water) density \( (\rho f) \).

The model is given by the following state equations:
\[
\begin{align*}
d P_e \frac{dt}{dt} &= 0.9 u_1 - 0.0018 u_2 P_r^{0.8} - 0.15 u_3, \\
d P_r \frac{dt}{dt} &= (0.073 u_2 - 0.016) P_r^{0.8} - 0.1 P_e, \\
d \rho f \frac{dt}{dt} &= (141 u_3 - (1.1 u_2 - 0.19) P_r)/85.
\end{align*}
\] (76)

Drum water level is calculated using the following algebraic equation:
\[
q_e = (0.85 u_2 - 0.14) P_r + 45.59 u_1 - 2.51 u_3 - 2.09,
\]
\[
\alpha_{cs} = \frac{(1 - 0.0015 \rho f)(0.8 P_r - 25.6)}{\rho f (1.0394 - 0.00123 P_r)},
\]
\[
L = 0.05(0.13 \rho f + 100 \alpha_{cs} + 0.11 q_e) - 67.97,
\] (77)
where \( \alpha_{cs} \) is the steam quality and \( q_e \) is the evaporation rate \( (\text{kg/s}) \). The positions of the valve actuators are constrained to lie in the interval [0,1], while their rate of change \( (\text{pu/s}) \) is limited as follows:
\[
-0.007 \leq \frac{du_1}{dt} \leq 0.007, \\
-2.0 \leq \frac{du_2}{dt} \leq 0.02, \\
-0.05 \leq \frac{du_3}{dt} \leq 0.05.
\] (78)

At the load level of 66.65 MW, pressure of 108 kg/cm² and fluid density of 428 kg/m³, the nominal inputs are found to be \( u_n = [0.34 0.69 0.43] \). These values are selected as initial points for inputs and state variables. The responses of the Extended DMC are shown in Figure 2. The graph consists of three outputs and inputs versus time. As indicated, good tracking is obtained for both pressure set point changes at \( t = 100 \) and power demand in \( t = 200 \). Since no change in drum level set point is required, the controller tries to compensate level deviation. Results justify the usage of an algorithm for computing \( d\Pi \) in a MIMO nonlinear system with \( M > 1 \).
In this paper, some extensions on EDMC are introduced. These extensions include application of the existing method on SISO, as well as MIMO systems, with longer control (M) and prediction (P) horizons. Analysis of convergence and closed loop stability, both in finite and infinite sampling time, are presented in detail and summarized in two theorems (Theorems 1 and 2). Simple conditions are obtained for some special cases, both for SISO and MIMO systems. Results given here confirm those in [13], which were obtained for a special case (M = P = 1 and T → ∞). Computer simulation results for a power unit plant indicate that application of the method has good performances in set point tracking.

REFERENCES