ISR
INSTITUTE FOR SYSTEMS RESEARCH

TECHNICAL RESEARCH REPORT

Optimization-based Tuning of Nonlinear Model Predictive Control with State Estimation

by E. Ali and E. Zafiriou

T.R. 93-51

The Institute for Systems Research is supported by the
National Science Foundation Engineering Research Center Program (NSFD CD 8803012),
the University of Maryland, Harvard University, and Industry

Optimization-based Tuning of Nonlinear Model Predictive Control with State Estimation

Emad Ali and Evanghelos Zafiriou*

Department of Chemical Engineering and Institute for Systems Research

A.V. Williams Bldg., University of Maryland,

College Park, MD 20742, U.S.A.

Abstract

Nonlinear Model Predictive Controllers determine appropriate control actions by solving an on-line optimization problem. A nonlinear process model is utilized for on-line prediction, making such algorithms particularly appropriate for the control of chemical reactors. The algorithm presented in this paper incorporates an Extended Kalman Filter, which allows operations around unstable steady-state points. The paper proposes a formalization of the procedure for tuning the several parameters of the control algorithm. This is accomplished by specifying time-domain performance criteria and using an interactive multi-objective optimization package off-line to determine parameter values that satisfy these criteria. Three reactor examples are used to demonstrate the effectiveness of the proposed on-line algorithm and off-line tuning procedure.

*Author to whom correspondence should be addressed. E-mail:zafiriou@src.umd.edu

1 Overview

In order to meet the increasing needs of designing control systems that take into account the nonlinear process characteristics, a number of Model Predictive Control (MPC) algorithms have emerged in the last decade, which directly utilize nonlinear models for on-line prediction. A description of various MPC algorithms is given in a review paper by Bequette[1]. Nonlinear MPC (NLMPC) has received a lot of interest due to its success in simulated tests and some industrial applications[2]. However, performance may deteriorate in the presence of model-plant mismatch especially for open-loop unstable processes with slow dynamics. Moreover, theoretical analysis for robustness in these cases is extremely difficult. This situation led researchers to incorporate process estimation algorithms for improving the accuracy of the model prediction.

Wright *et al.*[3], Li and Biegler[4], and Eaton and Rawlings[5] coupled their NLMPC algorithms with an optimization-based parameter estimation method to reduce the adverse effects due to parametric error as well as unmeasured disturbances. Both the unknown model parameters and the load disturbances are estimated on-line by minimizing a discrepancy between the past measurements and predictions of the process output using available optimization software.

Another attempt is the combined parameter and state estimation via nonlinear programming. This algorithm is proposed by Jang *et al.*[6] to continuously provide the controller with updates of the model parameters and estimates of the initial conditions of the state variables. Sistu and Bequette[7] satisfactory implemented the same estimation method to NLPC algorithm to control an open-loop unstable system corrupted by disturbances and imprecise parameter values.

A simple alternative way to compensate for the impact of model uncertainty is by augmenting the controller with a state estimator by Kalman Filtering as proposed by Ricker[8] for linear MPC. This approach was successfully extended by Gattu and Zafiriou[9] to Nonlinear Quadratic Dynamic Matrix Control (NLQDMC) for disturbance rejection of open-loop unstable processes.

In this study, we couple our NLMPC algorithm with on-line Extended Kalman Filter (EKF). The purpose is to improve the closed-loop performance of processes with slow dynamics and stablize open-loop unstable processes in the presence of unmeasured disturbances. This approach uses, every sampling time, linearized model dynamics along with preassumed characteristics of the unmeasured disturbances and measurement noise to evaluate the steady state filter gain. Knowing the filter gain and only the output measurement at the current sampling time, an additive correction term is computed and used in conjunction with the nonlinear process model to correct the current states and the predicted states over the future horizon.

As far as robust stability and performance conditions are concerned, little theoretical progress has been made for the case of nonlinear process. For unconstrained systems, Li *et al.*[10], and Gattu and Zafiriou[11] used the concept of contraction mapping to establish sufficient closed-loop stability conditions for their algorithms for the case of open-loop stable plants. For nonlinear models, however, these conditions are not useful from a practical point of view, since they are usually conservative[11]. The situation is much more complicated when constraints are included in the on-line optimization. For linear process dynamics, Zafiriou and Marchal[12] proved that the presence of hard constraints in the MPC algorithm can lead to instability, even though the unconstrained algorithm may be stable. The contraction mapping technique was successfully utilized in this case due to the linear process models. Muske and Rawlings[13] have also obtained stability results by using an infinite prediction horizon.

The great difficulties in developing theoretical conditions that guarantee stability and good performance of NLMPC, especially in the presence of disturbances, model uncertainty, and output constraints, lead designers to trial-and-error tuning of the NLMPC parameters. The prediction horizon and the control horizon can be used to adjust the speed of the response. Weights on the change of manipulated variables reduce the aggressiveness of the control action. For multi-output processes, the weights on the error for each output trade the performance of one output to another. The trade-off problem, however, for several competing objectives can be quite complex, especially, when one has to also include possible errors and disturbances.

This paper attempts a formalization of what is currently a trial-and-error procedure for NLMPC parameter tuning. The real-valued parameters of the algorithm are determined by an off-line optimization. While the integer parameters such as the prediction horizon and the control horizon are found by grid search. By modifying the NLMPC algorithm, the prediction horizon can also be treated as a continuous real variable which simplifies the grid search to one variable search. The objective of the off-line optimization is to ensure that certain performance specifications, e.g., required speed of response and limited overshoot, are satisfied in the presence of modeling error and disturbances that lie within certain maximum bounds. The off-line problem is solved with an interactive multi-objective optimization tool called CONSOLE, developed by Tits *et al.*[14]. Such a multi-objective approach has been used by Seaman *et al.*[15] to tune a PID controller.

Section two discusses the NLMPC algorithm and the state estimation by EKF algorithm. Section three describes the off-line tuning of the parameters. Section four uses three examples to illustrate the effectiveness of the proposed algorithm.

2 NLMPC Algorithm

2.1 On-line Optimization

In this paper we use a Model Predictive Control algorithm that utilizes a nonlinear dynamic model of the process for predicting future outputs and states. The algorithm finds a sequence of M future manipulated variables by minimizing on-line an objective function based on the desired output trajectories over a prediction horizon P. After the optimization, the first element of this future sequence is implemented. Then at the next sampling time, after a new measurement has been obtained, a new optimization is carried out. The objective function is as follows:

$$\min_{u(t_k),\ldots,u(t_{k+M-1})} \sum_{l=1}^{P} \parallel \Gamma e(t_{k+l}) \parallel^2 + \sum_{l=1}^{M} \parallel \Lambda \Delta u(t_{k+l-1}) \parallel^2 \tag{1}$$

Where k denotes the current sampling point, $t_i = iT$ with T the sampling time, and

$\| \cdot \|$ the Euclidean vector norm. The predicted error is defined as

$$e(t_{k+l}) = y(t_{k+l}) - r(t_{k+l}) + d(t_{k+l}) \tag{2}$$

where r is the setpoint, y is the model output, and d is the deviation of the process measurement from the model output. Since future measurements are not known, the disturbance $d(t_{k+l})$, for $l = 1, ..., P$, is considered constant in the future and equal to $d(t_k)$. The inputs u are constant between sampling points. Δu indicates the change in manipulated inputs ($\Delta u(t_{k+l}) = u(t_{k+l}) - u(t_{k+l-1})$). The inputs are assumed constant after $k + M - 1$, i.e., $\Delta u(t_{k+i}) = 0, i \geq M$. Γ and Λ are diagonal matrices of weights on the outputs and the change of manipulated variables respectively. Constraints on both u and Δu are also included in the optimization problem. The optimization is carried out with the NPSOL software, written by Gill *et al.*[16], which uses a Successive Quadratic Programming (SQP) algorithm.

The output prediction is obtained via numerical integration of the model differential equations for specified inputs, using the software package DASSL[17]. Incorporation of state estimation in the output prediction to compensate for model-plant mismatch and disturbances is discussed in the following section.

In section three it will be shown that tuning the integer parameters M and P is achieved by grid search because the off-line optimization algorithm accepts only continuous real variables. However, in our NLMPC formulation, P represents the integration horizon for output prediction, therefore we are not limited to using integer P. Hence the grid search can be simplified to one variable search by treating P as a real variable. To allow such a non-integer P, the objective function in equation (1) is modified as follows:

$$\min_{u(t_k),\ldots,u(t_{k+M-1})} \sum_{i=1}^{n_y} \Gamma_i^2 \bar{e}_i(t_{k+P})/T + \sum_{l=1}^{M} \| \Lambda \Delta u(t_{k+l-1}) \|^2 \tag{3}$$

where n_y denotes the number of outputs, and Γ_i denotes the i_{th} element of Γ. $\bar{e}_i(t_{k+P})$ is obtained by the simultaneous solution of the original model equations (with a state estimation correction term as described in the next section) and of

$$\dot{\bar{e}}_i(t) = (y_i(t) - r_i(t) + d_i(t))^2, i = 1, \ldots, n_y \tag{4}$$

The numerical integration start at t_k and ends at ends at t_{k+P}. The solution is obtained by repeated integration for one sampling interval using specific constant u for each sampling interval and constant d over the whole prediction horizon. When P is non-integer, the last integration interval is not equal to the sampling interval. Once again the inputs, u, are assumed constant after $k+M-1$. The computed error is then divided by the sampling time in order to get the same order of magnitude as the discrete error used in equation (1). Henceforth we shall refer to the NLMPC formulation with integer P as the standard NLMPC and to the one with non-integer P as the modified NLMPC.

2.2 State Estimation

In practice, the performance of NLMPC may become poor in the presence of model-plant mismatch, especially for open-loop unstable systems, where the model states may diverge away from the actual states. A state estimator can be combined with the on-line optimization to reset the model states.

Our approach uses an Extended Kalman Filter (EKF) which allows an *additive* state correction formulation. State estimation by EKF is easy to implement and requires less computational effort than approaches using nonlinear programming for solving a coupled parameter and estimation problem. The construction of the EKF for nonlinear models is discussed in Lewis[18]. Here we extend the calculational procedure of the continuous-discrete filter gain[18] to evaluate its steady state value which will be used for the standard NLMPC. The correction term is then equal to the product of the calculated steady state filter gain with the deviation of the current predicted output from its actual measured value.

To design the filter gain, assume that the model equations can be represented by:

$$\dot{x} = f(x, u, t) + w \tag{5}$$

$$y = h(x) + v \tag{6}$$

where $w \sim (0, Q)$ and $v \sim (0, R)$ are white Gaussian noise processes assumed to be independent of each other, and to characterize the unmeasured disturbances and the measurement noise respectively. Q and R are the respective covariances of w and v and they are assumed to be diagonal matrices of the form $Q = q^2 I$ and $R = r^2 I$. In the absence of accurate knowledge of the disturbance and noise characteristics, we further simplify the filter tuning method. Defining $\sigma^2 = q^2/r^2$ and letting $r^2 = 1.0$ will uniquely determine the Kalman Filter gain and simplify its tuning to determining only one parameter[19]. Although the trivial simplification of the covariance matrices is not optimal, it provides simple treatment against disturbance injection in the plant. This parameter can then provide a closed-loop observer, without which an MPC algorithm applied to an open-loop unstable process would be unstable. Therefore σ is used to provide stability and robustness in the presence of modeling error, disturbances, and measurement noise. One should note, however, that the assumption that w and v are white noise will result in biased state and/or output estimates in the presence of persistent disturbances, e.g., step-like input or output disturbances. This can be corrected with appropriate augmentation of the system of (5) and (6), but then a different "type" of model would have to be used for different cases. To avoid this, and since significant simplification is already accepted for Q and R, we use a simple additional correction of the output with the current disturbance estimate. This is accomplished by including the d term in (2) and (4). This concept was used by Ricker[8] for linear MPC and also by Gattu and Zafiriou[9] for NLQDMC.

The model output prediction for a set of M future values of u, which is determined by the online optimization loop, is described by the following steps:

step 1: Initialization. *Known at the current sampling point, k, are the plant measurement $\tilde{y}(t_k)$, the model state vector $x(t_k)$, the manipulated variable vector $u(t_{k-1})$. Set $P_k = I$.*

step 2: Linearization. *Obtain the following jacobians:*

$$A_k = \nabla_x f(x,u,t)\mid_{x=x(t_k),u=u(t_{k-1}),t=t_k} \tag{7}$$

$$H_k = \nabla_x h(x)\mid_{x=x(t_k)} \tag{8}$$

step 3: Time update. *Integrate numerically the error covariance equation:*

$$\dot{P}(t) = A_k P(t) + P(t) A_k^T + Q \tag{9}$$

for one sampling interval using DASSL, with initial condition $P = P_k$. Define the solution as $\bar{P}_k$.

step 4: Measurement update. *Compute the Kalman Gain:*

$$K_k = \bar{P}_k H_k^T [H_k \bar{P}_k H_k^T + R]^{-1} \tag{10}$$

and update the error covariance:

$$P_k = [I - K_k H_k]\bar{P}_k \tag{11}$$

Repeat steps 3 and 4 till steady state value of P_k is reached.

step 5: Correction factor. *Compute the Kalman Filter correction factor :*

$$F_k = K_k[\tilde{y}(t_k) - h(x(t_k))] \tag{12}$$

step 6: Output prediction. *Set $\hat{x}(t_k) = x(t_k) + F_k$ and integrate the state equation over one sampling time to get $x(t_{k+1})$, then reset the value of the new state to $\hat{x}(t_{k+1})$ by adding F_k, and evaluate the output $y(t_{k+1}) = h(\hat{x}(t_{k+1}))$. Repeat the last step over the prediction horizon P. During the integration the manipulated variables are allowed to vary at M discrete times, specifically at $t_k, t_{k+1}, \ldots, t_{k+M-1}$.*

Addition of F_k to the model states is used to compensate for modeling error. In addition, a constant prediction of the disturbance, d, is essentially added to the predicted output to eliminate steady state offset as indicated by the d term in (2) and (4). Several techniques for obtaining the limiting solution of (9) are discussed in the literature[20,21,22]. The method of direct numerical integration is used here for simplicity. The choice of an initial value of $P_k = I$ (step 1) does not affect the steady state value of P_k and K_k, obtained by the repetition of steps 3 and 4. The steady state value of P_k is the unique solution of the corresponding Ricatti equation[18].

For the case of modified NLMPC, the steady state Kalman gain calculation remains the same. However, the objective function does not explicitly utilize output prediction at each sampling point. Instead error prediction at $k+P$ (i.e., the value of $\bar{e}(t_{k+P})$) is needed. Therefore, step 6 is replaced by:

step 6: Error prediction. *Integrate simultaneously the differential equations:*

$$\dot{x} = f(x, u, t) + F_k/T \tag{13}$$

$$\dot{\bar{e}} = (y(t) - r(t) + d(t))^2 \tag{14}$$

from the sampling point k up to $k+P$ with initial conditions $x = x(t_k)$ and $\bar{e} = \bar{e}(t_k)$. During the integration the manipulated variables are allowed to vary at M discrete times, specifically at $t_k, t_{k+1}, \ldots, t_{k+M-1}$. Denote the solution as $\bar{e}(t_{k+P})$.

For small T, K_k/T approximates the continuous Kalman gain that should be used in (11). This approximation is used for simplicity.

3 Tuning Procedure

This section formulates the tuning parameter selection problem as an off-line optimization carried out with the interactive CONSOLE software[23]. This is a flexible formulation that allows one to use several types of performance criteria.

CONSOLE has been used for controller design in the past. For example, it was used to obtain the parameters of PID controllers[24,25]. Here we extend the same idea to tuning the NLMPC parameters. The off-line optimization can, e.g., determine the parameter values that make the closed-loop response stay within a preset constraint envelope over defined time domain as shown in Fig. 1. The envelope is used to represent various performance objectives. For example, constraints can be used to limit overshoot or undershoot, and/or maintain desired response speed. The values of constraints can be specified as functions of the setpoint change values, e.g., proportional to them. Alternatively one could choose to directly minimize the maximum possible overshoot [24,25].

The procedure can be described by the loop shown in Fig. 2 The desired performance specification, e.g., the envelope of Fig. 1 are given to CONSOLE. Also, possible setpoint changes, disturbances, model parameter errors etc, can be specified. By discretizing the range of possible values of these quantities one can define multiple objectives or constraints for CONSOLE, each of which corresponds to a particular set of values, and require that the specified performance criteria be satisfied or optimized. CONSOLE will then find suitable tuning parameter values that force every NLMPC response obtained for different discretized value of modeling error, setpoints, or disturbances to lie within the maximum bounds.

During the optimization carried out by CONSOLE, simulations of the closed-loop control system under NLMPC have to be run repeatedly for different tuning parameters, setpoints, disturbances, as well as model and plant. CONSOLE determines the next set of values to be tried, based on rigorous optimization theory, instead of trial-and-error. An added feature of CONSOLE is its interactive nature, which allows the designer to specify "good" and "bad" values for each performance constraint specification and interactively change them if CONSOLE can not find tuning parameter values to satisfy them. The good and bad values are used to scale the performance constraint specifications by dividing the deviation from the good value by the difference between the good and the bad values. If the difference between good and bad values is selected equal to unity, the corresponding constraint is not scaled. This concept is discussed in detail by Nye and Tits[26]. Note that we treat performance constraints as soft constraints in this paper.

It is also important to mention the concept of CONSOLE's set of parameters called *nominal variations*. The user should provide a nominal variation for each design parameter, besides its initial guess, to indicate his confidence in the initial guess. In fact it is a way to scale different design parameters so that when they are changed by amount equal to their nominal variation, they are expected to have an equal effect on the most active constraints and objectives. Therefore proper choice of the nominal variation may help accelerate the progress of the optimization process[27].

It should be emphasized that CONSOLE treats all design variables as continuous real variables. The control horizon M, and for standard NLMPC, the prediction horizon P are integer variables. Such variables can be usually approximated by continuous variables if their values are large. However, we wish to keep M and P as small as possible, while satisfying the specifications, in order to avoid unnecessary increasing the computations of the on-line optimization that NLMPC requires.

Therefore, when standard NLMPC algorithm is used, M and P are determined by performing grid search. For each grid point (i.e., each fixed values of M and P) CONSOLE is used to determine the real-valued parameters. On the other hand, with modified NLMPC, which allows for a non-integer P, only M has to be fixed, while the remaining parameters including P are determined by CONSOLE.

Note, that in addition to the usual MPC tuning parameters, the sampling time (T), and the covariance ratio (σ) can also be used as such parameters.

4 Illustrations

In this section three chemical reactor examples are used. The first two examples are open-loop unstable processes, and illustrate the capability of EKF to stablize the system when disturbances and parametric error exist. They also demonstrate the effectiveness of the tuning parameter selection procedure. The last example uses a pH control problem to demonstrate how the off-line tuning procedure can be used to maximize the sampling time while maintaining the best attainable performance. In all simulations, the effect of the upper and lower performance constraints are balanced by equating the difference between their "good" and "bad" values. In this case CONSOLE will try to satisfy both bounds equally. In the first two examples the optimization problem is formulated as a functional constraint problem. For the pH example we have an objective function for maximizing the sampling time. The results reported in this paper correspond to the case where the "bad" values are selected so that the absolute value of their difference from the "good" value is unity, thus using no special scaling of the constraints, as explained in section 3. We did examine the use of appropriate scaling for both examples in section 4.1 and 4.2, but the results were essentially identical to those where no scaling of the constraints was used.

For each off-line optimization solved with CONSOLE we report the total number of NLMPC simulations. Several NLMPC simulations (corresponding to different values of disturbances and modeling error) are required every time that a new point in the variable space (NLMPC tuning parameter space) has to be tried. This includes simulations made for numerical derivative computations. CONSOLE uses a forward difference formula for each parameter. The number of total NLMPC simulations is provided as an alternative to CPU time. The type of software that one uses for solving the on-line NLP has a significant effect on CPU time for each simulation. Several authors have tried to reduce the on-line computations by taking into account the specific nature of NLMPC optimization[28,29]. We have not attempted to do so and we are simply using the NPSOL package[16]. By looking at the number of of NLMPC simulations, this factor is neutralized and we can study the convergence speed of CONSOLE.

4.1 Catalytic CSTR Example

This example is taken from the paper by Brengel and Seider[29]. An exothermic catalytic reaction in the form of $A + B \rightarrow P$ is taking place. The reactor model is:

$$\dot{x}_1 = u_1 + u_2 - 0.2x_1^{0.5} \tag{15}$$

$$\dot{x}_2 = \frac{(C_{b1} - x_2)u_1}{x_1} + \frac{(C_{b2} - x_2)u_2}{x_1} - \frac{k_1 x_2}{(1 + k_2 x_2)^2} \tag{16}$$

The process outputs are $y_1 = x_1$ (tank level), and $y_2 = x_2$ (concentration of B in the reactor). C_{b1}, C_{b2} are the concentrations in the inlet feeds of concentrated and dilute B respectively. The manipulated variables are the corresponding flow rates u_1 and u_2. The model parameter values are $k_1 = k_2 = 1$, $C_{b1} = 24.9$, and $C_{b2} = 0.1$. The control goal is to move the process from initial stable steady state conditions of $u_1 = u_2 = 1$, $y_1 = 40$, and $y_2 = 0.4$, to a new unstable steady state at $y_1 = 100$ and $y_2 = 2.787$ by manipulating u_1 and u_2. This step change test is carried out in the presence of a disturbance at the inlet concentration C_{b1}, and physical constraints on the manipulated variables between 0 and 10. For this situation the concentration response suffers from excessive overshoot and slow disturbance rejection as reported by Brengel and Seider.

Tuning of the NLMPC parameters is necessary in order to reduce the overshoot, reject disturbances, and maintain proper speed of the response. In order to impose these desired properties with CONSOLE, they are translated into transient upper and lower bounds on the response of the process outputs y_1 and y_2. For y_2 tight upper bounds of 2.86 and 2.8 for the intervals 0-10 and 10-30 *min* respectively are employed to prevent overshoot. Lower bounds of 2.6 and 2.73 for the intervals 5-10 and 10-30 *min* respectively, are also imposed to avoid sluggishness and ensure convergence of the response to its final steady state. For y_1 a constant upper bound of 101 and a lower bound of 99 from time 10 to 30 *min* were used to avoid performance degradation due to penalty weight variation. The performance bounds are represented by the solid lines in Figs. 3 to 5.

Grid search along with CONSOLE is used to determine the optimal values of M, P, σ, Λ, and the weight on the second output (Γ_2) that simultaneously force the responses of the setpoint change under three different values of disturbances to fulfill the performance constraints. Note that Γ_1 was not tuned because the others are simply relative to it. In particular step disturbances of magnitude 5.0, 0, and -5.0 on C_{b1} were used. The results of this search, which uses a sampling time of $1min$, are summarized in Table 1. At each fixed value of M and P, the parameters σ, Λ, and Γ_2 in Table 1 are the final values found by runing CONSOLE with initial values of (0,diag[0,0],1) and nominal variation values of (1,diag[1,1],1) respectively. The table indicates whether the three responses at the obtained parameter values satisfy the bounds or not. Since for $M = 1$, values were found for the other tuning parameters that satisfy the specifications, there is no need to try larger values for M. Note that small M is desirable for computational reasons. The table also shows the total required NLMPC simulations which have been used for each grid point.

For small values of P the response of y_2 is so aggressive that CONSOLE was not able to move from the initial values of the tuning parameters. For large values of P the response of y_1 is slower and further decrease of the overshoot was not possible because it forces the response of y_1 to violate the lower bounds. Only at $M = 1$ and $P = 3$ optimal tuning parameters that successfully satisfy the desired objectives were found. Figure 3 shows the setpoint responses of y_1 and y_2 at $M = 1$, $P = 3$, and the initial values of the tuning parameters. Figure 4 shows the responses for the same values of M and P but at the final values of the tuning parameters. The figures also show the responses at additional disturbances values of +2.0 and -2.0. Since this system has slow dynamics it takes a longer time to settle down to its final steady state value which would be clear if longer simulation time was used in the figures.

The use of modified NLMPC that utilizes a non-integer P was also considered. In this case the value of M has to be fixed and CONSOLE is used to determine the optimal values of P, Λ, σ, and Γ_2. The results are given in Table 2 for fixed value of $M = 1$. It should be noted that the unequal variation values shown in the table helped in improving the progress of the off-line optimization process. Unfortunately the obtained parameters were not optimal but they are probably the best ones. The corresponding simulations at these values are shown in Fig. 5. The bound violations are negligible. Even though the output bounds were too tight, we got an acceptable solution because the bounds are soft. This example illustrates that, even if the bounds were chosen somewhat arbitrarily, a good solution, not necessarily satisfying the bounds, is not eliminated.

Different initial guesses of P and σ in addition to equal variation values were also tried but no successful results were obtained.

Finally, one should note that the NLMPC simulations required by CONSOLE would have been reduced by about one third, if the case of zero disturbance on C_{b1} was not used in CONSOLE.

4.2 CSTR Temperature Control Example

This control problem is adapted from Sistu and Bequette[7]. The objective is to control the temperature of an exothermic irreversible reaction taking place in a CSTR by manipulating the cooling jacket temperature. The dimensionless model equations are given by:

$$\dot{x}_1 = -px_1K(x_2) + q(x_{1f} - x_1) \tag{17}$$

$$\dot{x}_2 = Bpx_1K(x_2) - (q + s)x_2 + su + qx_{2f} \tag{18}$$

$$y = x_2 \tag{19}$$

where $K(x_2) = exp(x_2/(1 + x_2/20))$, x_1 is the dimensionless concentration, x_2 is the dimensionless temperature, and u is the dimensionless cooling jacket temperature. Initial conditions are $u = 0$, $x_1 = 0.856$, and $x_2 = 0.8859$, which correspond to a stable steady state point. The nominal values of the model physical parameters are

$B = 8.0, s = 0.3, q = 1.0, p = 0.072$, $x_{1f} = 1.0$, and $x_{2f} = 0.0$.

Sistu and Bequette[7] used coupled parameter and state estimations via a nonlinear programming technique in order to address performance deterioration due to modeling error or disturbances. This example was used to test their parameter estimation algorithm by two simulations. The first one is a setpoint change to the unstable steady state (y=2.7517) for the case where the values of the dimensionless heat transfer coefficient, s, are 0.2 for the model and 0.3 for the plant, and u is bounded between -1 and 2. While the second simulates the response to a +20% feed composition (x_{1f}) disturbance for the same initial conditions and values of s. In both cases $M = 1$, $P = 5$, and a sampling time of 0.25 were used.

In this paper we try to determine NLMPC parameters for which the desired performance characteristics are maintained in the presence of modeling error in s, without the need for changing on-line the model parameter values. For the same simulation conditions and sampling time, we use CONSOLE to obtain NLMPC parameters that make the responses fit inside desired performance constraints. These constraints are chosen to represent similar performance to that obtained by Bequette and Sistu. The bounds used for the two tests are as follows; for the setpoint change a constant upper bound of 2.81 and a lower bound of 2.7 between 5 and 20 were implemented. For the disturbance case upper bounds of 0.9 and 0.887 for the time intervals 0-5 and 5-20 and a constant lower bound of 0.885 were imposed.

For the purpose of comparison with the results of Bequette and Sistu, the heat transfer coefficient s of the plant is fixed at 0.3, while for the model values of 0.2 and 0.3 were examined. A summary of the results in the case of integer prediction horizon P for initial guess of (0,0) and nominal variation of (1,1) is given in Table 3 At M=1, our search indicated that higher values of P are not favorable. Large values of P make the setpoint response sluggish and no further improvement can be made by tuning σ and Λ. At M=2 and P=5 bound satisfaction was obtained.

Figure 6 gives the responses for the two cases at $M = 2$ and $P = 5$ for initial value of $\sigma = 0$ and $\Lambda = 0$. Figure 7 shows the improved responses at the final value of $\sigma = 3.53$ and $\Lambda = 0.015$. It is clear that the final values of σ and Λ result in excellent responses, comparable to those of Bequette and Sistu.

The use of non-integer P eliminates the need for a large grid search. In this case we fix M and use CONSOLE to search for P, σ, and Λ for the modified NLMPC algorithm. Our results for the case of $M = 1$ indicated that no solution can be found. This result agrees with the grid search results for the same value of M. The results for the case of $M = 2$ are given in Table 4. For the first initial guess we reported two different results each with different set of variations values. When a large variation on P is used, its value changes faster than the other parameters. This made the optimization end at a local optimum. In the equal variation case the optimization moved towards a solution that satisfied the bounds. The simulations for these values are shown in Fig. 8. Although all responses successfully satisfy the constraints, a slight oscillation is observed in the disturbance-rejection response of the imperfect model. This oscillation was ignored by CONSOLE since it satisfies the bounds within the specific simulation time used in CONSOLE. By investigating the same response with longer simulation time we found that the response is in fact unstable. One way to eliminate this oscillation is to try using tighter constraints on the outputs of the imperfect model. Therefore we reran CONSOLE starting from the previously obtained parameters using new tighter constraints. The new constraints are the same as before except that in the cases of imperfect model we imposed additional upper and lower bounds of $\pm 5 \times 10^{-7}$ from the setpoints from time 15 to 20. New parameter values and improved stable responses were obtained as given in Table 4 and shown in Fig. 9.

4.3 pH control Example

This example considers a highly nonlinear process studied in Li *et al.*[10]. The process consists of two CSTRs in series and a pH measurement champer. In this example perfect modeling is assumed. The control objective is to bring the pH value to a desired setpoint by manipulating the acid flow rate. The process model can be written as:

$$\dot{x}_1 = -5 \times 10^{-6} + 0.1u - (0.05 + u)x_1 \tag{20}$$

$$\dot{x}_2 = (0.05 + u)(x_1 - x_2) \tag{21}$$

$$\dot{x}_3 = 10(x_2 - x_3) \tag{22}$$

$$y = -log_{10}(0.5(x_3 + (x_3^2 + 4 \times 10^{-14})^{0.5})) \tag{23}$$

where u is the manipulated variable (acid flow rate; pH=1), x_1, x_2, and x_3 are the difference in concentrations $(H^+ - OH^-)$ in the first tank, second tank, and pH measuring champer respectively. The steady state values are $u = 0$, $x_1 = x_2 = x_3 = -10^{-4}$, and the new setpoint is $y = 7$. This control objective is examined by Li *et al.*[10] using a single-step Newton-type controller with a sampling time $(T) = 40sec$, where oscillatory response was observed. Oscillation is eliminated by adding an integral control term to the Newton-type controller, however a sluggish response is obtained. Here we deal with this problem by tunning M and P as well as the sampling time.

By adjusting M and P, for a sampling time of $40sec$, we found that at $M = 1$ and $P = 3$ a non-oscillatory pH response is obtained as shown by the dashed line in Fig. 10. Next we seek to maximize the sampling time, for the case where $M = 1$ and $P = 3$, in order to reduce the overall online computational effort. Maximization is carried out by CONSOLE subject to some pH performance constraints as shown by the solid lines in Fig. 10. Specifically, upper bounds were chosen to be 10, and 7.01 for the intervals 0-400, and 400-500 *sec* respectively. While a lower bound of 6.95 were set for the whole interval. These bound correspond to best performance obtained in Li *et. al.*[10].

By runing CONSOLE the maximum sampling time to preserve our performance requirements is found to be $263.9sec$ when T is bounded above by $500sec$ and to be $180sec$ when bounded by $180sec$. The responses at the maximum sampling time are given by the dot-and-dash and the solid lines in Fig. 10.

Since P can be treated as non-integer variable in the modified NLMPC algorithm, the above objective, i.e., maximizing the sampling time while satisfying the performance constraints, is reconsidered by tuning both T and P at fixed M for the same upper bounds on T as above and an upper bound at 10 for P. The results are reported in

Table 5 and simulations shown in Fig. 10. The maximum T obtained in this case is larger than what was found in the standard NLMPC for the case when T is bounded above by 500sec. The reason is that in the modified NLMPC P is allowed to vary. One should note of course that unmeasured disturbances and modeling errors would probably result in smaller values for T. The goal of this illustration is to show how additional designer-defined parameters like the sampling time can be tuned simultaneously with the NLMPC parameters via CONSOLE.

5 Concluding Remarks

This paper considered the question of tuning a nonlinear Model Predictive Control algorithm. A differential equation model is assumed for the process and an Extended Kalman Filter is incorporated to allow stable operation around open-loop unstable steady state points and better disturbance rejection. This particular type of algorithm was selected because there are no design techniques for selecting the tuning parameters for robust stability and performance. In the absence of robustness conditions, we have cast the tuning problem as an off-line optimization, with the goal of satisfying time domain specifications. These specifications may include mixed setpoint tracking and disturbance rejection requirements, in the presence of modeling error.

The use of the software package CONSOLE proved very effective in obtaining solutions to the off-line optimization problem. One can expect the technique to work even more efficiently when applied to problems in which the simulations of MPC algorithms are less computationally demanding, as, for example, the case of discrete nonlinear models, like neural networks, or linear models. Although in the case of linear models theoretical design techniques are available, the time-domain orientation of this approach may be preferable to a designer.

The illustrating examples have also pointed out certain weaknesses that future work will have to address. As the temperature control example demonstrated, no stability guarantees are provided. Caution and careful examination of the solution can help avoid a problem, but it is no substitute for rigorous theoretical conditions, assuming such conditions are available. Another problem is the existence of several local optima for the off-line optimization which may generate the need for repeated runs. One should note though, that after a set of tuning parameter values is found that satisfies the time-domain specifications, it is sufficient, regardless of being only a local optimum. Still, a better understanding of the properties of the off-line optimization is needed.

Acknowledgements

Support for this project is provided by the National Science Foundation (PYI grant CTS-9057292), the Institute for Systems Research, and a grant from Texaco. The authors are grateful to Dr. Jian Zhou and Prof. André Tits for making the CONSOLE software available, and for their help in using it.

References

1. Bequette, B. W. *Ind. & Eng. Chem. Res.*, 1991, **30**, 1391.

2. Garcia, C. E. "Quadratic Dynamic Matrix Control of Nonlinear Processes: An Application to a Batch Reaction Process", *AICHE Ann. mtg.*, San Francisco, CA, 1984.

3. Wright, G. T., Breedijk, T. B., and Edgar, T. F. "On-line Parameter Estimation and Adaptation in Nonlinear Model-Based Control", Amer. Control Conf., Boston, MA, 2782-2787, 1991.

4. Li, W. C., and Biegler, L. T. *Ind. & Eng. Chem. Res.*, 1990, **29**, 1647.

5. Eaton, J. W., and Rawlings, J. B. *Comp. and Chem. Eng.*, 1990, **14**, 469.

6. Jang, S. S., Joseph, B., and Mukai, H. *Ind. & Eng. Chem. Res.*, 1987, **26**, 2106.

7. Sistu, P. B., and Bequette, B. W., *AICHE J.*, 1991, **37**, 1711.

8. Ricker, N. L. *Ind. & Eng. Chem. Res.*, 1990, **29**, 374.

9. Gattu, G., and Zafiriou, E. *Ind. & Eng. Chem. Res.*, 1992, **31**, 1096.

10. Li, W. C., Biegler, L. T., Economou, C. G., and Morari, M., *Comp. and Chem. Eng.*, 1990, **14**, 451.

11. Gattu, G., and Zafiriou, E., *IFAC/DYCORD+ '92 preprints*, College Park, MD, 1992, 303.

12. Zafiriou, E., and Marchal, A. L. *AICHE J.*, 1991, **37**, 1550.

13. Muske, K. R., and Rawlings J. B. *AICHE J.*, 1992, **39**, 262.

14. Fan, M. K., Tits, A. L., Zhou, J., Wang, L., and Koninckx, J., "Console User's Manual (version 1.1)", System Research Center, University of Maryland, Technical Report SRC-TR 87-212r2, College Park,MD, 1990.

15. Seaman, C. M., Desrochers, A. A., and List, G. F. *Identification and Multi-Objective Controller Tuning of a Plastic Injection Molding Machine*, Proc. Amer. Control Conf., 262-267, Boston, MA, 1991.

16. Gill, P. E., Murray, W., Sauders, M. A., and Wright, M. H., "User's Guide for NPSOL (version 4.0): A Fortran Program for Nonlinear Programming", Technical Report SOL 86-2, Department of Operations Research, Stanford University 1986.

17. Petzold, L. R., "A Description of DASSL: A Differential-Algebraic System Solver", in *Scientific Computing*, R. S. Stepleman, Ed., North-Holand 1983.

18. Lewis, F. L., "Optimal Estimation with Introduction to Stochastic Control Theory", Wiley and Sons, New York, 1986, 260.

19. Hamilton, J. C., Seborg, D. E., and Fisher, D. G. *AICHE J.*, 1973, **19**, 901.

20. Kwakernaak, H., Sivan, R., "Linear Optimal Control Systems", Wiley and Sons, New York, 1972, 248.

21. Hagander, P. *Information Sciences*, 1972, **4**, 35.

22. Rothschild, D., and Jameson, A. *Int. J. Control*, 1970, **11**, 181.

23. Fan, M. K., Wang, L. S., Koninckx, J., and Tits, A. L. "CONSOLE : A CAD Tandem for Optimization-Based Design Interacting with User-Supplied Simulators", Proc. of the American Control Conference, Atlanta, GA, 1988.

24. Fan, M. K., "Optimization-Based Design of Nonlinear Systems Using CONSOLE and SIMNON", System Research Center, University of Maryland, Technical Report SRC-TR87-204, College Park, MD, 1987.

25. Zhou, J., *Ph.D Dissertation*, Electrical Engineering Department,University of Maryland, College Park, MD, USA, 1992.

26. Nye, W. T., and Tits, A. L. *Int. J. Cont.,* 1986, **43**, 1693.

27. Fan, M. K., Wang, L. S., Koninckx, J., and Tits, A. L., *IEEE Control System Magazine*, 1989, **9**, 1.

28. Li, W. C., Biegler, L. T. *Ind. and Eng. Chem. Res.*, 1988, **27**, 562.

29. Brengel, D. D., and Seider, W. D. *Ind. & Eng. Chem. Res.*, 1989, **28**, 1812.

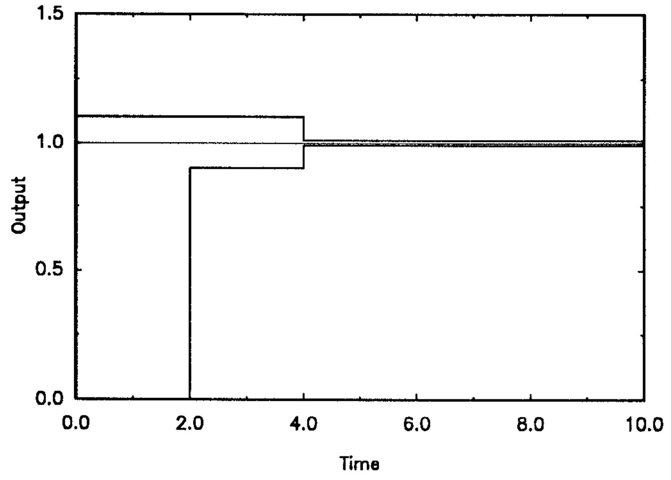

Figure 1: Example of desired response profile for setpoint tracking

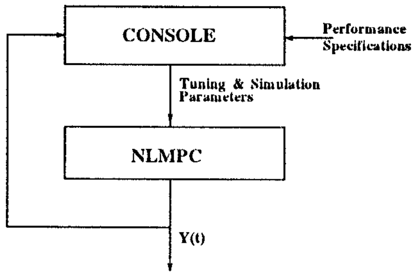

Figure 2: Tuning procedure

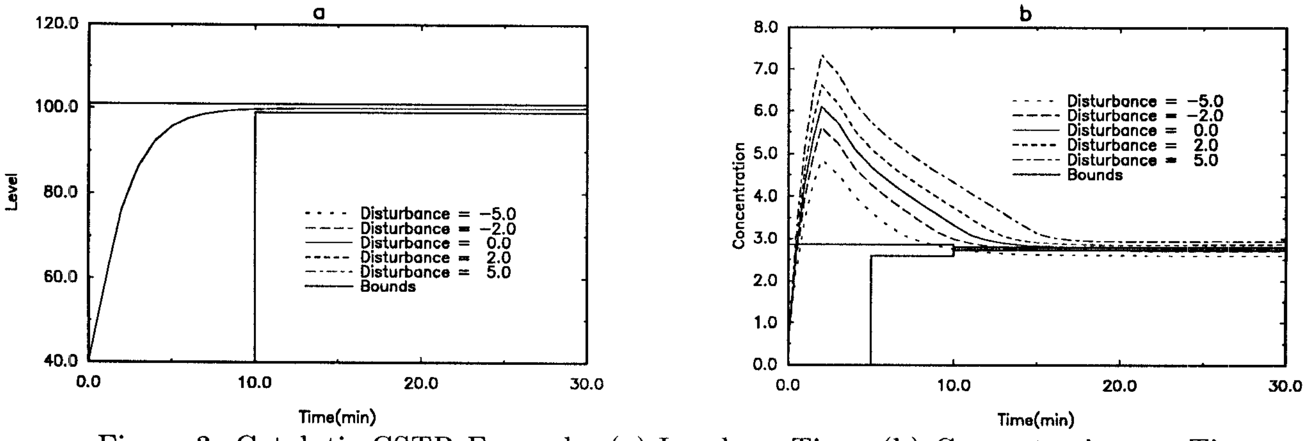

Figure 3: Catalytic CSTR Example. (a) Level vs. Time, (b) Concentration vs. Time. $0 \leq u \leq 10$, $M = 1$, $P = 3$, $\sigma = 0$, Λ =diag[0,0], Γ =diag[1,1].

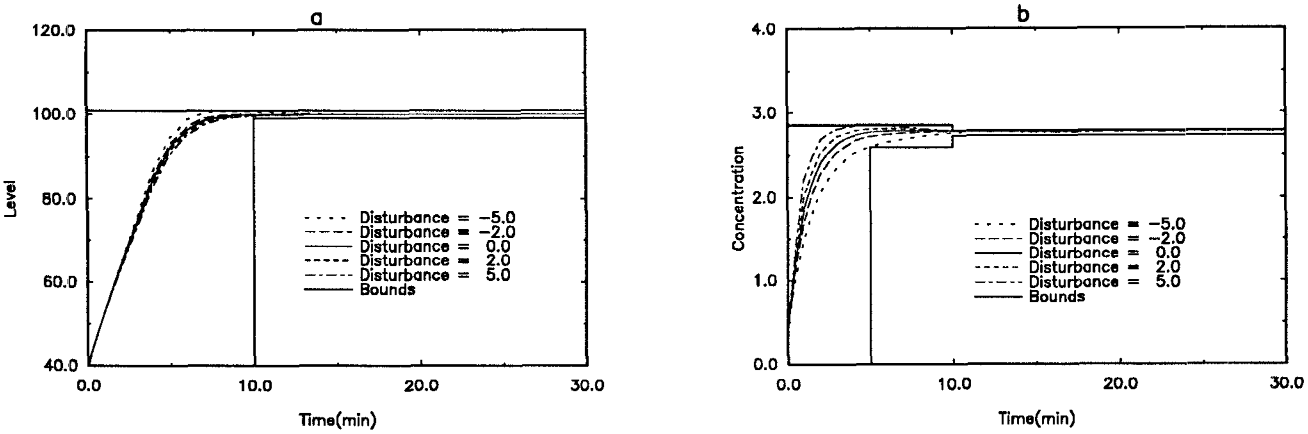

Figure 4: Catalytic CSTR Example. (a) Level vs. Time, (b) Concentration vs. Time. $0 \leq u \leq 10$, $M = 1$, $P = 3$, $\sigma = 0.087$, Λ =diag[15.4,0], Γ =diag[1,108.6].

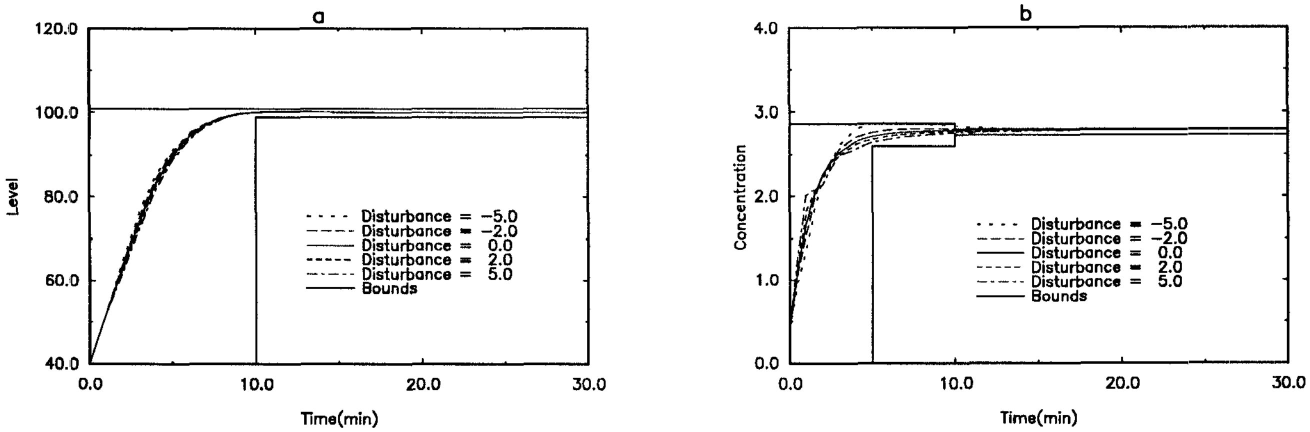

Figure 5: Catalytic CSTR Example. (a) Level vs. Time, (b) Concentration vs. Time. $0 \leq u \leq 10$, $M = 1$, $P = 4.94$, $\sigma = 6.346$, Λ =diag[5.4,5.76], Γ =diag[1,37.25].

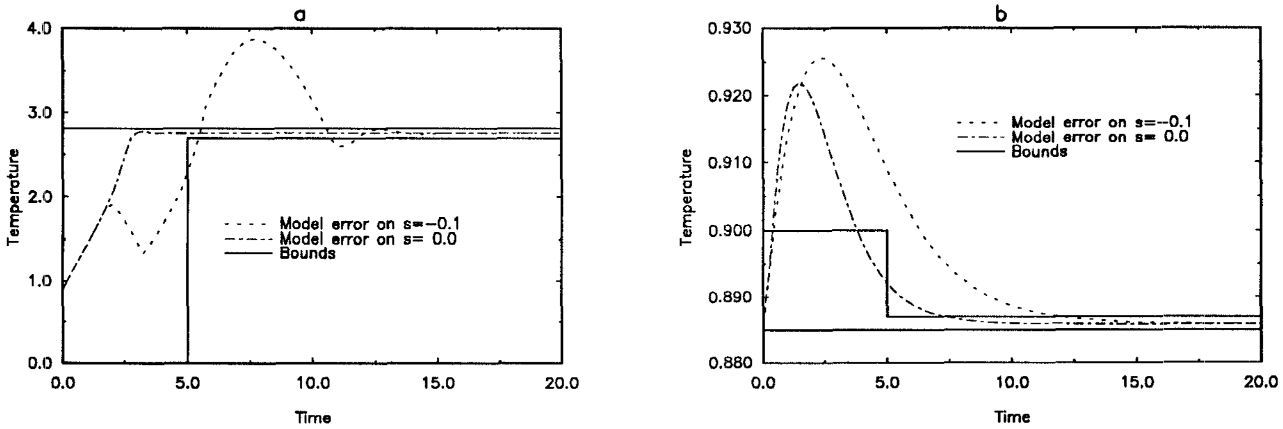

Figure 6: Temperature Control Example. (a) setpoint tracking. (b) disturbance rejection. $M = 2$, $P = 5$, $\sigma = 0$, $\Lambda = 0$.

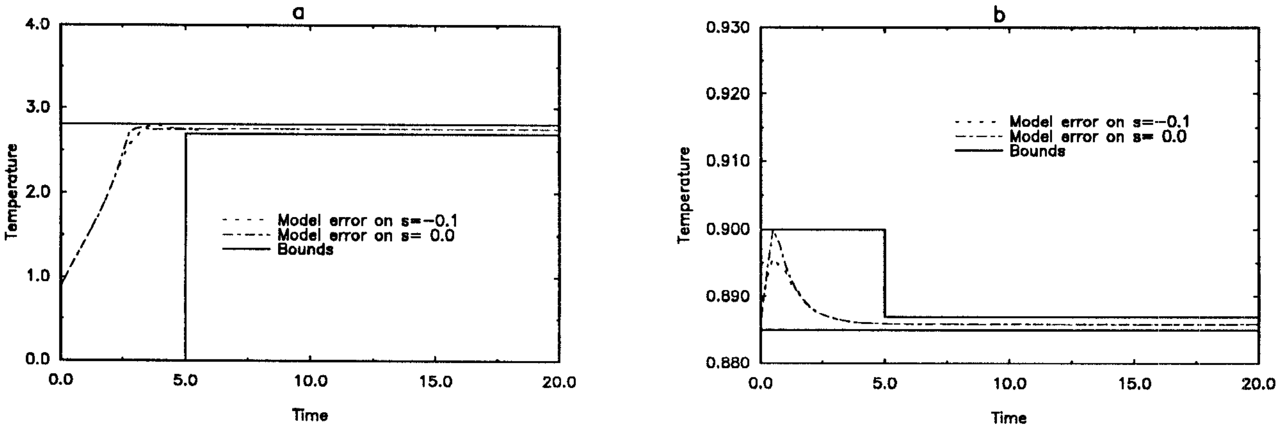

Figure 7: Temperature Control Example. (a) setpoint tracking. (b) disturbance rejection. $M = 2$, $P = 5$, $\sigma = 3.53$, $\Lambda = 0.015$.

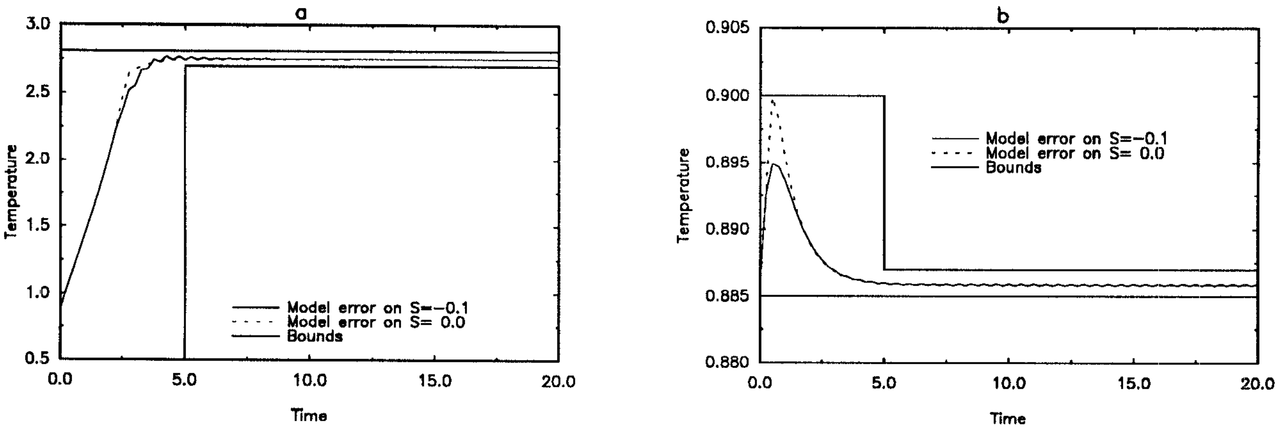

Figure 8: Temperature Control Example. (a) setpoint tracking. (b) disturbance rejection. $M = 2$, $P = 9.506$, $\sigma = 2.52$, $\Lambda = 6.2 \times 10^{-3}$.

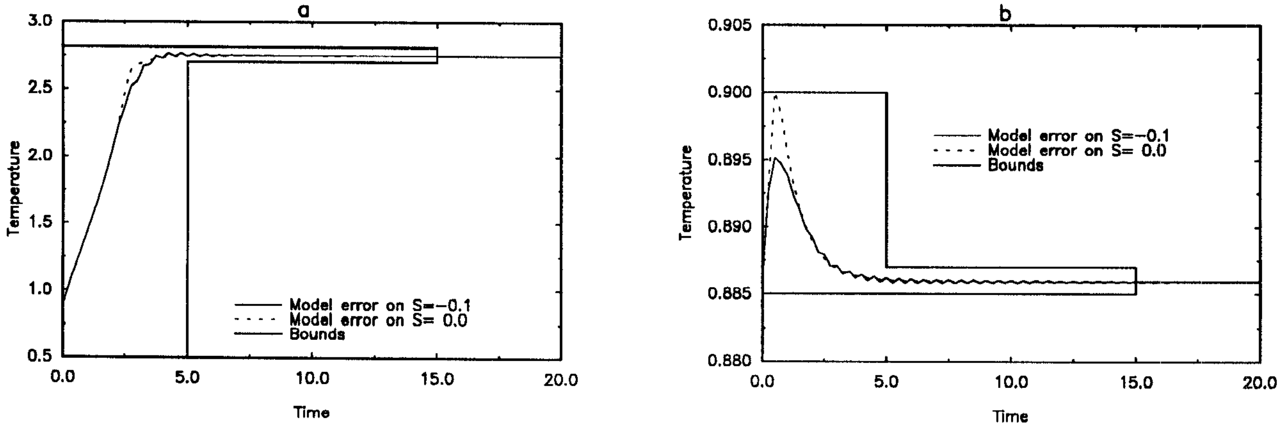

Figure 9: Temperature Control Example. (a) setpoint tracking. (b) disturbance rejection. $M = 2$, $P = 9.51$, $\sigma = 2.52$, $\Lambda = 1.04 \times 10^{-2}$.

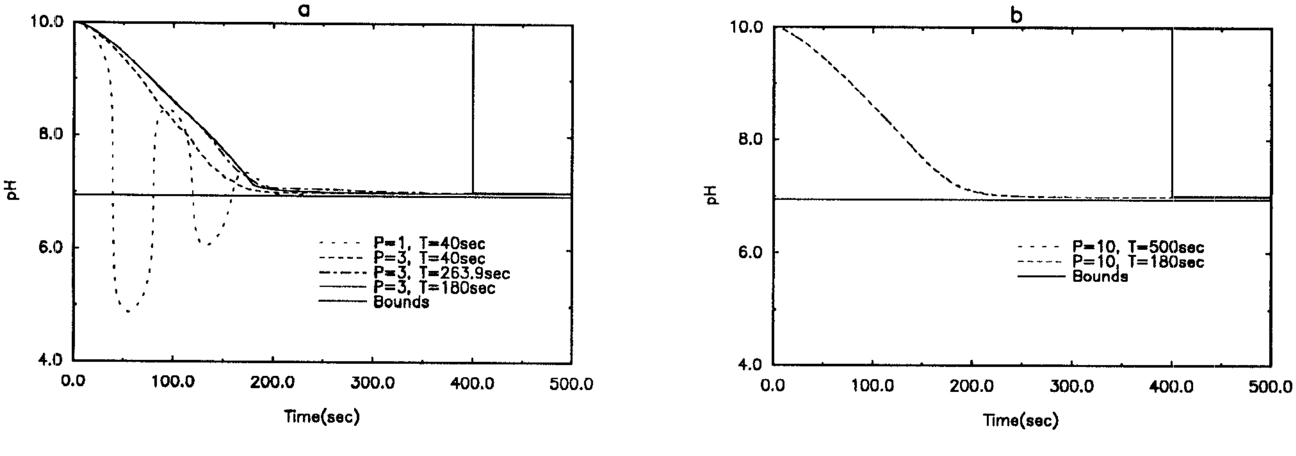

Figure 10: pH Control Example. $M = 1$. (a) standard NLMPC (b) modified NLMPC.

Table 1: Results of the tuning parameter search for example 1 using standard NLMPC.

M	P	Γ_2	σ	Λ [diag]	B	N
1	1	1.0	0.0	[0,0]	No	12
	2	1.9	4.0e-9	[0,0]	No	12
	3	108.6	0.087	[15.46,0]	Yes	147
	4	18.44	0.097	[0,0]	No	129
	5	168.6	0.109	[1.8,5.3]	No	363
	6	18.83	0.134	[1.85,13.2]	No	255
	7	9.88	0.157	[21.6,13.3]	No	328

B=Bounds satisfied, N=NLMPC simulations

Table 2: Results of the tuning parameter search for example 1 using modified NLMPC.

Parameters	M=1		
	Initial	Variation	Final
P	10.0	10	4.943
σ	1.0	1	2.524
Λ[diag]	[0,0]	[1,1]	[5.4,5.76]
Γ_2	1.0	10*	37.24
B	—	—	No
N	—	—	776

B=Bounds satisfied, N=NLMPC simulations

* Γ_2^2 was used as variable with variation 100.

Table 3: Results of the tuning parameter search for example 2 using standard NLMPC.

M	P	σ	Λ	B	N
1	1	1.26	0.0	No	72
	2	56.2	0.0	No	84
	3	15.9	0.0	No	84
	4	11.5	0.0	No	84
	5	5.95	0.0	No	88
	6	15.1	0.0	No	84
2	2	1.04	0.0	No	72
	3	0.922	0.0	No	72
	4	0.99	0.0	No	72
	5	3.53	0.015	Yes	184
	6	1.04	0.0	No	70

B=Bounds satisfied, N=NLMPC simulations

Table 4: Results of the tuning parameter search for example 2 using modified NLMPC.

Parameters	M=2					
	Initial	Variation	Final	Initial	Variation	Final
P	10.0	10(1)	2.67(9.506)	9.506	1	9.51
σ	1.0	1(1)	1.56(2.52)	2.52	1	2.52
Λ	0.0	1(1)	$9.6\times10^{-3}(6.1\times10^{-3})$	6.1×10^{-3}	1	1.04×10^{-2}
B	—	—	No(Yes)	—	—	Yes
N	—	—	168(184)	—	—	265

B=Bounds satisfied, N=NLMPC simulations

Table 5: Results for the tuning parameter search for example 3 using the modified NLMPC

Parameters	M=1		
	Initial	Tmax=180	Tmax=500
P	1.0	10.0	10.0
T	40.0	180.0	500.0
B	—	Yes	Yes
N	—	15	23

B=Bounds satisfied, N=NLMPC simulations