Tutorial Overview of Dynamic Optimization and Advanced Control for Industrial Systems

John D. Hedengren, Reza Asgharzadeh Shishavan

Department of Chemical Engineering, Brigham Young University, Provo, UT 84602

Abstract

Linear models are applied in many applications of industrial Advanced Process Control (APC) and a large user base of control engineers and consultants has grown to deploy and sustain these applications. Although project efficiency is increasing and implementation time is decreasing, there remains an art of applying linear Model Predictive Control (MPC) applications. This undocumented art is especially employed when the linear technology is extended to systems that have nonlinear characteristics. This paper describes methods to apply nonlinear models in adaptive model identification and control. This tutorial overview guides the user through a small multivariable level control problem that has a number of common characteristics to larger controllers seen in practice. The overview reveals additional fundamental capabilities by employing nonlinear models in applications of industrial relevance.

Keywords: advanced process control, differential algebraic equations, model predictive control, dynamic parameter estimation

1. Introduction

Applications of Model Predictive Control (MPC) are ubiquitous in a number of industries such as refining and petrochemicals. Applications are also somewhat common in chemicals, food manufacture, mining, and other manufacturing industries [1]. A majority of the applications employ linear models that are constructed from empirical model identification, however, many of these processes have either semi-batch characteristics or nonlinear behavior. To ensure that the linear models are applicable over a wider range of operating conditions and disturbances, the linear models are retrofitted with elements that approximate nonlinear control characteristics. Some of the nonlinear process is captured by including gain scheduling, switching between multiple models depending on operating conditions, and other logical programming when certain events or conditions are present. The art of using linear models to perform nonlinear

Email address: john.hedengren@byu.edu (John D. Hedengren)

control has been refined by a number of control experts to extend linear MPC to a wider range of applications. While this approach is beneficial in deploying applications, maintenance costs are increased and sustainability is decreased due to the complexity of the heuristic rules and configuration.

The purpose of this article is not to detail the best practices with linear models, but instead to give practical guidance on using nonlinear models in the typical steps of an MPC project. Recent advancements in numerical techniques have permitted the direct application of nonlinear models in control applications [2], however, many nonlinear MPC applications require advanced training to build and sustain an application. Perhaps the one remaining obstacle to further utilization of nonlinear technology is the ease of deploying and sustaining applications by practitioners. Up to this point, there remain few actual industrial applications of control based on nonlinear models. The objective of this paper is to reduce the barriers to implementation of nonlinear advanced control applications. This is attempted by giving a practical guide and providing details on the following topics:

- nonlinear model development combining empirical and first principles elements
- parameter estimation from dynamic data
- advanced process control with large-scale nonlinear models
- conversion of differential equations into algebraic equations for efficient solution by nonlinear programming solvers

In addition to the theoretical underpinnings of the techniques, a practical application with process data is used to demonstrate model identification and control. The application used in this paper is a simple level control system that was selected to illustrate the concepts without burdening the reader with model complexity. In practice, much larger and more complex systems can be solved using these techniques. An illustration of scale-up to larger problems gives an indication of the size that can be solved with current computational resources. The example problems are demonstrated with the APMonitor software, a freely available package for solution of differential and algebraic equation (DAE) systems. Specific examples are included in the appendices with commands to reproduce the examples in this paper.

This paper includes a number of innovative techniques for solving large-scale control and optimization problems. One novel contribution of this paper is modified L_1-norm objective forms for estimation and control. These objective forms have a number of advantages over traditional squared-error or L_2-norm objectives that are often reported in the literature. Another contribution is the tutorial nature of the article with concise source code that can be used to reproduce the results or develop further applications. The target audience is the practitioner or researcher interested in applying nonlinear estimation and control to challenging applications of industrial relevance.

2. Model Identification with Nonlinear Equations

A critical aspect of any control project is reliable model identification. Linear model identification used in practice is typically with Finite Impulse Response (FIR) or State Space (SS) forms. Nonlinear model identification also involves adjustment of parameters to fit process data. Linear model identification is often an empirical approach while nonlinear models have some structure that results from material and energy balances, reaction kinetic mechanisms, or other pre-defined model structure. As a result of the model structure, there are fewer parameters that need to be adjusted to fit the process response. Model structure may also include constraints such as fixed gain ratios, zero lower limits on compositions, or other bounds that reflect physical realism.

Along with model form, the form of the identification objective function is important to ensure desirable results. The most common objective form is the least squares form: $(y_{model} - y_{meas})^2$ (see Equation 1). Although intuitive and simple to implement, the squared error form has a number of challenges such as sensitivity to bad data or outliers. The sensitivity to outliers is exacerbated by the squared error objective, commonly proposed for dynamic data reconciliation [3] [4] [5] [6] [7].

Table 1 details the equations of the typical squared error norm and the novel L_1-norm objective. A new form of the identification objective improves some of the problems with the L_2 form. The L_1-norm formulation in Equation 2 is less sensitive to data outliers and better rejects measurement noise. The form is similar to an absolute value function but is instead solved with inequality constraints and slack variables. The absolute value operator is not continuously differentiable which can cause convergence problems for Nonlinear Programming (NLP) solvers. On the other hand, the L_1-norm slack variables and inequalities create an objective function that is smooth and continuously differentiable.

Other challenges in aligning the model to measured values (data reconciliation) include lack of data diversity to obtain certain constants or co-linearity of parameters. The sensitivity of parameters to the objective function can help guide which parameters have a significant effect on the outcome [8]. One solution to automatically eliminate parameters with little sensitivity to objective is to impose a small penalty on parameter movement from a nominal value [9]. This approach automatically prevents unnecessary movement of parameter values that have little effect on the results of the parameter estimation. Historically, most of the techniques for parameter estimability rely on steady-state or statistical analysis of linearized systems. The approach in this article extends the linear or steady-state analysis to dynamic and nonlinear systems.

3. Application: Quadruple Tank Level Control

A quadruple tank process shown in Figure 1 has been the subject of theoretical [10] and practical demonstrations [11] [12] [13] [14] of a multivariable and highly coupled system [12]. The four tank process has also been a test application for application of decentralized and coordinated control techniques

Table 1: Estimation: Two Forms for Dynamic Data Reconciliation

Estimation with a Squared Error Objective (Most Common)

$$
\begin{aligned}
\min_{d} \Phi = & \left(y_x - y_m\right)^T W_m \left(y_x - y_m\right) \ldots + \left(\Delta d\right)^T c_{\Delta d} \\
& \ldots + \left(y_m - \hat{y}_m\right)^T W_p \left(y_m - \hat{y}_m\right) \\
\text{s.t.} \quad & 0 = f(\dot{x}, x, u, p, d) \\
& 0 = g(y_x, x, u, d) \\
& a \geq h(x, u, d) \geq b
\end{aligned} \tag{1}
$$

Estimation with an L_1-norm Objective (Better Outlier Rejection)

$$
\begin{aligned}
\min_{d} \Phi = & \, w_m^T \left(e_U - e_L\right) + w_p^T \left(c_U + c_L\right) + \left(\Delta d\right)^T c_{\Delta d} \\
\text{s.t.} \quad & 0 = f(\dot{x}, x, u, p, d) \\
& 0 = g(y_x, x, u, d) \\
& a \geq h(x, u, d) \geq b \\
& \\
& e_U \geq y_x - y_U \\
& e_L \geq y_L - y_x \\
& c_U \geq y_x - \hat{y}_m \\
& c_L \geq \hat{y}_m - y_x \\
& e_U, e_L, c_U, c_L \geq 0
\end{aligned} \tag{2}
$$

Nomenclature for Equations 1 and 2	
Φ	objective function
y_x	measurements $\left(y_{s,0}, \ldots, y_{s,n}\right)^T$
y_m	model values $\left(y_{m,0}, \ldots, y_{m,n}\right)^T$
$\hat{y}_m$	prior model values $\left(\hat{y}_{m,0}, \ldots, \hat{y}_{m,n}\right)^T$
w_m, W_m	penalty outside measurement dead-band
w_p, W_p	penalty from the prior solution
$c_{\Delta d}$	penalty from the prior disturbance values
f	equation residuals
x	states
u	inputs
d	parameters or unmeasured disturbances
Δd	change in parameters
g	output function
h	inequality constraints
a	lower limits
b	upper limits
e_U	slack variable above the measurement dead-band
e_L	slack variable below the measurement dead-band
c_U	slack variable above the previous value
c_L	slack variable below the previous value

[15] [16]. A number of other interesting characteristics of this process include configurations that cause the system to go unstable. This can be observed by showing that there are right-hand plane (RHP) zeros. Another challenge is the nonlinear tendency of the system. For example, this can be characterized by variable gains of the manipulated variables (MVs) to the controlled variables (CVs).

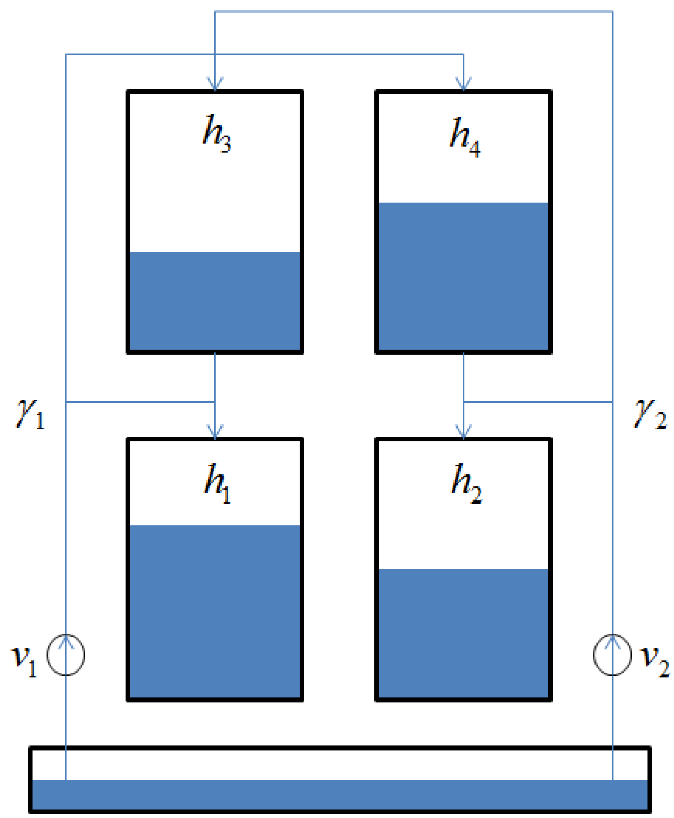

Figure 1: Diagram of the quadruple tank process. Pump 1 supplies tanks 1 and 4 while pump 2 supplies tanks 2 and 3.

The four tank process has two pumps that are adjusted with variable voltage to pump 1 (v_1) and pump 2 (v_2)). A fraction of water from pump 1 is diverted to tank 1 proportional to γ_1 and to tank 4 proportional to $(1 - \gamma_1)$. Similarly, a fraction of water from pump 2 is diverted to tank 2 proportional to γ_2 and to tank 3 proportional to $(1 - \gamma_2)$. The valves that determine γ_1 and γ_2 are manually adjusted previous to the experiment and are held constant throughout a particular period of data collection. All tanks are gravity drained and tank 3 outlet enters tank 1. Tank 4 outlet enters tank 2, creating a coupled system of manipulated and controlled variables. For $(\gamma_1 + \gamma_2) \in (0, 1)$, the linearized system has no RHP zeros with for $(\gamma_1 + \gamma_2) \in (1, 2)$, the linearized system has one RHP zero [10]. A RHP zero indicates that there may either be overshoot or an inverse response to a step change in the manipulated variable.

A combination of material balances and Bournoulli's law yields the process model for the four tank process as shown in Equation 3. The equations are also displayed in Appendix A in the APMonitor Modeling Language.

$$\begin{aligned}
&q_a = k_m v_1 + k_b \\
&q_b = k_m v_2 + k_b \\
&q_{1,in} = \gamma_1 q_a + q_{3,out} \\
&q_{2,in} = \gamma_2 q_b + q_{4,out} \\
&q_{3,in} = (1 - \gamma_2)\, q_b \\
&q_{4,in} = (1 - \gamma_1)\, q_a \\
&q_{1,out} = c_1 \sqrt{2gh_1} \\
&q_{2,out} = c_2 \sqrt{2gh_2} \\
&q_{3,out} = c_3 \sqrt{2gh_3} \\
&q_{4,out} = c_4 \sqrt{2gh_4} \\
&A_1 \frac{\partial h_1}{\partial t} = q_{1,in} - q_{1_out} \\
&A_2 \frac{\partial h_2}{\partial t} = q_{2,in} - q_{2_out} \\
&A_3 \frac{\partial h_3}{\partial t} = q_{3,in} - q_{3_out} \\
&A_4 \frac{\partial h_4}{\partial t} = q_{4,in} - q_{4_out}
\end{aligned} \tag{3}$$

where

γ_1	split factor for tanks 1 and 4
γ_2	split factor for tanks 2 and 3
c_i	outflow factor for tank i
k_m	valve linearization slope
k_b	valve linearization intercept
q_a	flow from pump 1
q_b	flow from pump 2
$q_{i,in}$	inlet flow to tank i
$q_{i,out}$	outlet flow from tank i
A_i	cross-sectional area of tank i
h_i	height of liquid in tank i

The process model is nonlinear because the outlet flow is proportional to the square root of the liquid level. In this experiment, tanks 1 and 3 and tanks 2 and 4 have the same outlet diameter making $c_1 = c_3$ and $c_2 = c_4$. Additionally, tanks 1 and 3 have a cross-sectional area of $28cm^2$ while tanks 2 and 4 have a cross-sectional area of $32cm^2$. Unknown parameters include γ_1, γ_2, $c_{1,3}$, $c_{2,4}$, k_m, and k_b. The unknown parameters are determined from dynamic data.

3.1. Parameter Estimation from Dynamic Data

Pseudo-random binary signals (PRBS) are a popular technique to generate linear plant response models from data [17]. This section demonstrates that PRBS-generated data can be used to determine optimal parameters for nonlinear dynamic models as well. Another technique for fitting model parameters to process data is the use of multiple steady-state data sets [18]. Control engineers identify steady-state periods that cover the major process operating regions of interest. One of the drawbacks to fitting a model with steady-state data is that dynamic parameters cannot be fit from the data. Dynamic parameters are those values that are multiplied by the derivatives with respect to time in the equations. In the case of a linear first order system $\left(\tau \frac{\partial y}{\partial t} = -y + Ku\right)$ the dynamic parameter is τ. However, process time constants can typically be estimated from process fundamentals such as vessel holdups and flow rates. In many cases, the time constants can be approximated reasonably well. However, using only steady-state data for fitting parameters can limit the observability of certain parameters that can only be determined with dynamic data. If nonlinear MPC is to be used to the full potential, dynamic data must be used to fit the models.

Using dynamic data to fit nonlinear dynamic models has a number of challenges. One of the challenges is that the data reconciliation problem can be very large. The data reconciliation problem is large because an instance of the model must be calculated at every time instant where a measurement is available. Using the simultaneous optimization of model and objective function, the number of model states at a particular time is multiplied by the number of time steps.

3.2. Quadruple Tank Parameter Estimation

For the quadruple tank process the model has only 14 differential or algebraic states. When calculated over the PRBS data horizon the resulting optimization problem has 5766 to 11,526 variables, depending on the objective function form. There are no differential states in the optimization problem due to the orthogonal collocation transformation (see Section 5). Orthogonal collocation on finite elements is one of the methods to convert DAE systems into a Nonlinear Programming (NLP) problem [19]. This is accomplished by approximating time derivatives of the DAE system as algebraic relationships. Figure 2 shows the results of the reconciliation to the PRBS-generated data.

Only levels for tanks 1 and 2 are measured as shown in Figure 2. For the quadruple tank process 6 parameters were estimated, namely γ_1, γ_2, $c_{1,3}$, $c_{2,4}$, k_m, and k_b. The optimization problem overview is shown in Table 2 while initial and final values of the parameters are displayed in Table 3. An APM MATLAB script for configuring and solving this problem is shown in Appendix B. The MATLAB script uses the APMonitor Modeling Language [20] model (see Appendix A) to create the differential and algebraic (DAE) model. APMonitor translates the problem into an NLP and IPOPT, an interior point large-scale nonlinear programming solver [21], is used to solve the resulting optimization problem. A summary of the optimization problem and the solution is shown in Table 2.

Using different objective function forms resulted in similar parameter estimates and comparable model predictions. As seen in Table 3, the optimal values for the parameters were well within the upper and lower constraints.

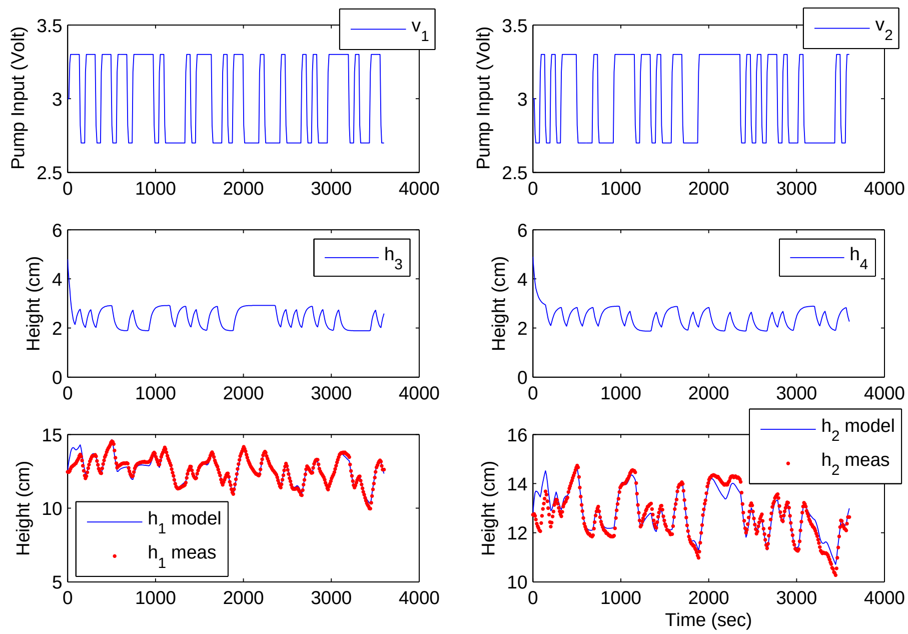

Figure 2: The results of the dynamic parameter estimation using PRBS generated data.

Table 2: Summary of the Dynamic Data Reconciliation

Optimization Problem Overview		
Description	L_1-Norm	Squared Error
Iterations	33	10
CPU Time (2.5 GHz Intel i7 Processor)	32.5 sec	10.3 sec
Number of Variables	11,526	5,766
Number of Equations	11,520	5,760
Degrees of Freedom	6	6
Number of Jacobian Non-zeros	40,312	28,792

These constraints were set based on knowledge of the process; a violation of these constraints would indicate unreasonable parameter values. In this case, the L_1-Norm optimization problem was roughly twice the size and required 3 times the amount of CPU time to find a solution. In this case, the increased computational time is additional cost associated with improved outlier rejection and parameter estimates.

Table 3: Results of the Dynamic Data Reconciliation

Initial and Final Values of the Estimation Problem					
Parameter	Initial Value	Lower Bound	Upper Bound	L_1-Norm	Squared Error
γ_1	0.43	0.20	0.80	0.627	0.585
γ_2	0.34	0.20	0.80	0.591	0.548
$c_{1,3}$	0.071	0.010	0.200	0.0592	0.0630
$c_{2,4}$	0.057	0.010	0.200	0.0548	0.0582
k_m	10.0	3.0	20.0	3.543	3.444
k_b	0.00	-2.00	2.00	-1.675	-0.810

4. Nonlinear Control and Optimization

There are many challenges to application of Differential and Algebraic Equations (DAEs) directly in nonlinear control and optimization. Enabling advances include simultaneous methods [22], decomposition methods [23] [24], efficient nonlinear programming solvers [21], improved estimation techniques [25] [26] [27] [28], and experience with applications to industrial systems [9] [29]. In particular, industrial applications require high service availability, reasonable extrapolation to operating conditions outside the original training set, and explanatory tools that reveal the rationale of the optimization results. Other motivating factors include consideration of lost opportunity during application development, sustainability of the solution, and ease of development and maintenance by engineers without an advanced training. In many instances non-technical challenges such as equipment and base-control reiability, operator training, and management support are critical factors in the success of an application [30].

The most popular objective function form in academia and for industrial applications is the squared error or L_2-norm objective (see Equation 4). In this form, there is a squared penalty for deviation from a setpoint or desired trajectory. The squared error objective is simple to implement, has a relatively intuitive solution, and is well suited for Quadratic Programming (QP) or Nonlinear Programming (NLP) solvers.

An alternative form of the objective function is the L_1-norm objective (see Equation 5) that has a number of advantages over the squared error form similar to those discussed for the estimation case. For control problems, the advantage is not in rejection of outliers but in the explicit prioritization of control objectives.

The L_1-norm simultaneously optimizes multiple objectives in one optimization problem as the solver manipulates the degrees of freedom selectively for the objective function contributions that have the highest sensitivity. For problems that have safety, environmental, economic, and other competing priorities, the L_1-norm gives the most intuitive form that manages these trade-offs. Table 4 details the square error and L_1-norm objective functions.

4.1. Nonlinear Control of the Quadruple Tank System

Returning to the quadruple tank system, the squared error model parameters from Section 3.2 are used to update the model. Either the squared-error or the L_1-norm objective estimation values can be used because of nearly equivalent results. Data reconciliation can either be performed once or repeatedly as new measurements arrive in a receding horizon approach. As new measurements arrive, the model is readjusted to fit the data and continually refine the model predictions. These updated parameters can then be used in the MPC application to better predict the future response.

Once the model is updated, nonlinear control calculates the optimal trajectory of Manipulated Variables (MV). In this case, a future move plan of the voltage to the two pumps is calculated and displayed Figure 3. MV moves are constrained by change, upper, and lower limits. The change constraints are set to limit the amount that the MV can move for each control action step and in this case the move limit is set to $|\Delta MV| \leq 1$. With a cycle time of 1 second, the rate that the voltage to the pump can change is $\pm 1 \frac{V}{sec}$. The control action is also constrained by absolute minimum ($MV_L = 1$) and maximum ($MV_U = 6$) limits. The lower limit is reached for the first pump ($v1$) and remains at the lower limit for 30 seconds before settling at the steady state value at $1.41V$. The upper limit is reached for second pump ($v2$) within two steps into the horizon and afterwards settles to a steady state value of $4.58V$. This over-shoot or under-shoot of manipulated variables is typical for controlled variable (CV) tuning that is faster than the natural process time constant. The natural process time constant is the speed of response due to a step change in a process input. When requesting a response that is faster than this nominal step change, the manipulated variables must over-react to move the process faster. In most cases, steady state values of the manipulated variables are independent of the controller tuning.

CV tuning is a critical element to achieving desirable control performance. Aggressive CV tuning is shown in this example, giving over- or under-shoot of the manipulated variables. For CV tuning that is equal to the natural process time constant, there will typically be a step to the new solution. For slower CV tuning, the MV ramps to the steady state value.

There are many types of CV tuning options that are typical in linear or nonlinear control applications. In this case, an L_1-norm with dead-band is demonstrated for the simulated controller. The speed of the CV response is dictated by an upper and lower first order reference trajectory with time constant τ_c. Only values that are outside this dead-band are penalized in the objective

Table 4: Control: Two Objective Forms for Nonlinear Model Predictive Control

Control Squared Error Objective

$$\begin{aligned}
\min_u \Phi &= (y_m - y_t)^T W_t (y_m - y_t) \ldots \\
&\ldots + (y_m)^T c_y + (u)^T c_u + (\Delta u)^T c_{\Delta u} \\
\text{s.t.} \quad & 0 = f(\dot{x}, x, u, p, d) \\
& 0 = g(y_x, x, u, d) \\
& a \geq h(x, u, d) \geq b
\end{aligned} \tag{4}$$

Control L_1-norm Objective

$$\begin{aligned}
\min_d \Phi &= w_{hi}^T (e_{hi}) + w_{lo}^T (e_{lo}) \ldots \\
&\ldots + (y_m)^T c_y + (u)^T c_u + (\Delta u)^T c_{\Delta u} \\
\text{s.t.} \quad & 0 = f(\dot{x}, x, u, p, d) \\
& 0 = g(y_x, x, u, d) \\
& a \geq h(x, u, d) \geq b \\
& \\
& \tau_c \frac{\partial y_{t,hi}}{\partial t} + y_{t,hi} = sp_{hi} \\
& \tau_c \frac{\partial y_{t,lo}}{\partial t} + y_{t,lo} = sp_{lo} \\
& e_{hi} \geq (y_m - y_{t,hi}) \\
& e_{lo} \geq (y_{t,lo} - y_m)
\end{aligned} \tag{5}$$

Nomenclature for Equations 4 and 5	
Φ	objective function
y_m	model values $(y_{m,0}, \ldots, y_{m,n})^T$
$y_{t,hi}$, $y_{t,lo}$	desired trajectory dead-band
w_{hi}, w_{lo}	penalty outside trajectory dead-band
c_y, c_u, $c_{\Delta u}$	cost of y, u and Δu, respectively
f	equation residuals
x	states
u	inputs
d	parameters or unmeasured disturbances
g	output function
h	inequality constraints
a	lower limits
b	upper limits
τ	time constant of desired controlled variable response
e_{lo}	slack variable below the trajectory dead-band
e_{hi}	slack variable above the trajectory dead-band

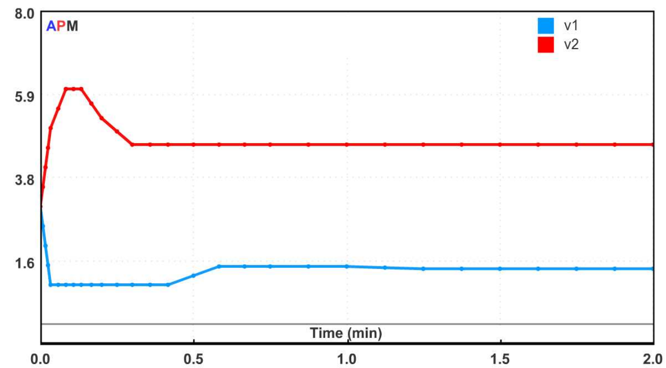

Figure 3: Optimal solution of the voltage input to the pumps 1 and 2.

function. The form of this controller objective is desirable for minimizing unnecessary MV movement to achieve a controller objective. In this form, MV movement only occurs if the projected CV response is forecast to deviate from a pre-described range. Figures 4 and 5 display the CV response along with the upper and lower trajectories that define the control objective.

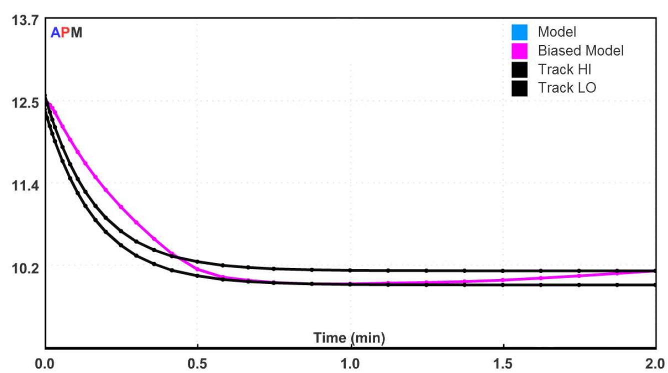

Figure 4: Height response in tank 1 and the associated controller objective.

5. Numerical Solution of DAE Systems

The simultaneous solution of the model equations and objective function has a number of advantages over other techniques. Other methods include the direct shooting approaches [31] where the objective function and model equations are solved separately and iteratively towards an optimal solution. With a simultaneous solution of the objective and model equations, there is improved

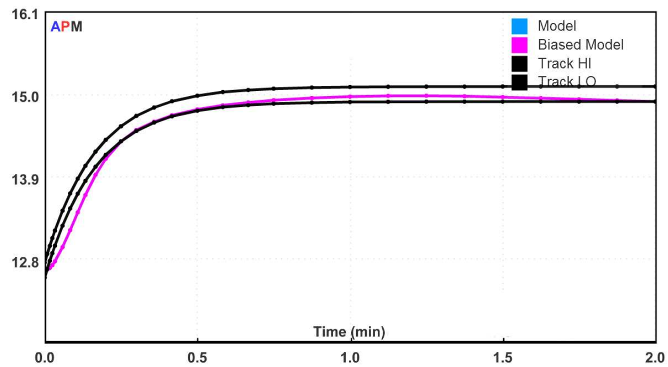

Figure 5: Height response in tank 2 and the associated controller objective.

computational performance with additional degrees of freedom. Another advantage with a simultaneous solution is that a general DAE model form can be posed in open equation format (see Equation 6).

$$\begin{aligned} 0 &= f\left(\frac{\partial x}{\partial t}, x, y, p, d, u\right) \\ 0 &= g(x, y, p, d, u) \\ 0 &\leq h(x, y, p, d, u) \end{aligned} \tag{6}$$

In open equation format, DAE models of index-1 or higher are solved without rearrangement. The solution of Equation 6 is determined by the initial state x_0, a set of parameters p, a trajectory of disturbance values $d = (d_0,d_1,\ldots,d_{n-1})$, and a sequence of control moves $u = (u_0,u_1,\ldots,u_{n-1})$. The values of the parameters, disturbances, or decision variables (p, d, and u) are discrete values over the time horizon to make the problem tractable for numerical solution (e.g. Manipulated Variables in Figure 6). On the other hand, integrated variables are determined from differential and algebraic equations and generally have a continuous profile (e.g. Controlled Variables in Figure 6). One solution approach to this dynamic system is the conversion of the DAE system to algebraic equations through direct transcription [2]. This technique is also known as orthogonal collocation on finite elements [32]. Converting the DAE system to a Nonlinear Programming (NLP) problem permits the solution by large-scale solvers [4] [33]. Additional details of the simultaneous approach are shown in Section 5.1 and an example problem in Section 5.2.

5.1. Derivation of Weighting Matrices for Orthogonal Collocation

The objective is to determine a matrix M that relates the derivatives to the non-derivative values over a horizon at points $1,\ldots,n$. In this case, four points are shown for the derivation. The initial value, x_0, is a fixed initial condition.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = M \left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} - \begin{bmatrix} x_0 \\ x_0 \\ x_0 \end{bmatrix} \right) \tag{7}$$

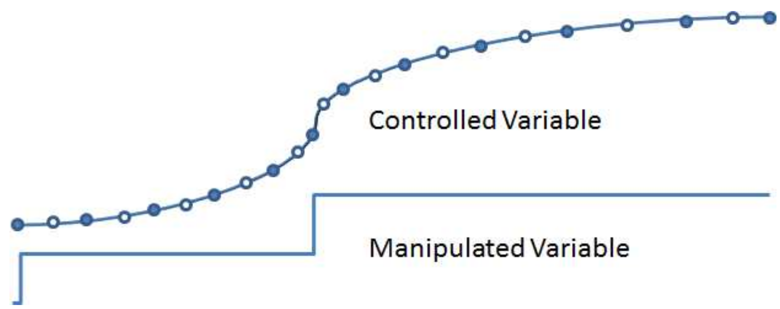

Figure 6: Dynamic equations are discretized over a time horizon and solved simultaneously. The solid nodes depict starting and ending locations for local polynomial approximations that are pieced together over the time horizon. With one internal node for each segment, this example uses a 2nd order polynomial approximation for each step.

The solution of the differential equations at discrete time points is approximated by a Lagrange interpolating polynomial as shown in Equation 8.

$$x(t) = A + Bt + Ct^2 + Dt^3 \tag{8}$$

Time points for each interval are chosen according to Lobatto quadrature. All time points are shifted to a reference time of zero ($t_0 = 0$) and a final time of $t_n = 1$. For 3 nodes per horizon step, the one internal node is chosen at $t_1 = \frac{1}{2}$. An example of internal nodes are displayed in Figure 6 where the horizon is broken into multiple intervals of Lobatto quadrature with 3 nodes per horizon step (one internal node). In the case of 4 nodes per horizon step, the internal values are chosen at $t_1 = \frac{1}{2} - \frac{\sqrt{5}}{10}$ and $t_2 = \frac{1}{2} + \frac{\sqrt{5}}{10}$. With 5 nodes, time values are $\frac{1}{2} - \frac{\sqrt{21}}{14}$, $\frac{1}{2}$, and $\frac{1}{2} + \frac{\sqrt{21}}{14}$. At 6 nodes, time values are $\frac{1}{2} - \frac{\sqrt{7+2\sqrt{7}}}{42}$, $\frac{1}{2} - \frac{\sqrt{7-2\sqrt{7}}}{42}$, $\frac{1}{2} + \frac{\sqrt{7-2\sqrt{7}}}{42}$, and $\frac{1}{2} + \frac{\sqrt{7+2\sqrt{7}}}{42}$. In this derivation, a third-order polynomial approximates the solution at the four points in the horizon. Increasing the number of collocation points increases the corresponding polynomial order. For initial value problems, the coefficient A is equal to x_0, when the initial time is arbitrarily defined as zero. To determine the coefficients B, C, and D, Equation 8 is differentiated and substituted into Equation 7 to give Equation 9. Note that the A coefficient from Equation 8 is cancelled by x_0 on the right-hand side of Equation 9.

$$\begin{aligned}
\begin{bmatrix} B + 2Ct_1 + 3Dt_1^2 \\ B + 2Ct_2 + 3Dt_2^2 \\ B + 2Ct_3 + 3Dt_3^2 \end{bmatrix} &= M \begin{bmatrix} Bt + Ct_1^2 + Dt_1^3 \\ Bt + Ct_2^2 + Dt_2^3 \\ Bt + Ct_3^2 + Dt_3^3 \end{bmatrix} \\
\begin{bmatrix} 1 & 2t_1 & 3t_1^2 \\ 1 & 2t_2 & 3t_2^2 \\ 1 & 2t_3 & 3t_3^2 \end{bmatrix} \begin{bmatrix} B \\ C \\ D \end{bmatrix} &= M \begin{bmatrix} t_1 & t_1^2 & t_1^3 \\ t_2 & t_2^2 & t_2^3 \\ t_3 & t_3^2 & t_3^3 \end{bmatrix} \begin{bmatrix} B \\ C \\ D \end{bmatrix}
\end{aligned} \tag{9}$$

Finally, rearranging and solving for M gives the solution shown in Equation 10.

$$M = \begin{bmatrix} 1 & 2t_1 & 3t_1^2 \\ 1 & 2t_2 & 3t_2^2 \\ 1 & 2t_3 & 3t_3^2 \end{bmatrix} \begin{bmatrix} t_1 & t_1^2 & t_1^3 \\ t_2 & t_2^2 & t_2^3 \\ t_3 & t_3^2 & t_3^3 \end{bmatrix}^{-1} \tag{10}$$

The final form that is implemented in practice is shown in Equation 11 by inverting M and factoring out the final time t_n $\left(t_n N = M^{-1}\right)$. This form improves the numerical characteristics of the solution, especially as the time step approaches zero ($t_n \to 0$).

$$t_n N \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} - \begin{bmatrix} x_0 \\ x_0 \\ x_0 \end{bmatrix} \right) \tag{11}$$

The matrices that relate $\frac{\partial x}{\partial t}$ to x are given in Table 5 for intervals with 3 to 6 nodes.

5.2. Example Solution by Orthogonal Collocation

A simultaneous solution demonstrates the application of orthogonal collocation. In this case, the first order system $\tau \frac{\partial x}{\partial t} = -x$ is solved at 6 points from $t_0 = 0$ to $t_n = 10$ using Equation 18. In this case $\tau = 5$ and the initial condition is specified at $x_0 = 1$. For this problem, the time points for $\frac{\partial x}{\partial t}$ and x are selected as 0, 1.175, 3.574, 6.426, 8.825, and 10. The value of x is specified at $t_0 = 0$ due to the initial condition. As a first step, equations for $\frac{\partial x}{\partial t}$ are generated in Equation 20.

$$\frac{\partial x}{\partial t} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \end{bmatrix} = \left(t_n N_{5x5}\right)^{-1} \left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} - \begin{bmatrix} x_0 \\ x_0 \\ x_0 \\ x_0 \\ x_0 \end{bmatrix} \right) \tag{20}$$

Substitution of Equation 20 into the derivatives of the model equation yields a linear system of equations as shown in Equation 21.

$$\tau \frac{\partial x}{\partial t} = - x$$

$$\tau \left(t_n N_{5x5}\right)^{-1} \left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} - \begin{bmatrix} x_0 \\ x_0 \\ x_0 \\ x_0 \\ x_0 \end{bmatrix} \right) = - \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \tag{21}$$

Equation 21 is rearranged and solved with a linear algebra as shown in Equation 22.

Table 5: Orthogonal Collocation on Finite Elements

Orthogonal Collocation Matrices

$$t_n N_{2x2} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \begin{bmatrix} x_0 \\ x_0 \end{bmatrix} \right) \quad (12)$$

$$N_{2x2} = \begin{bmatrix} 0.75 & -0.25 \\ 1.00 & 0.00 \end{bmatrix} \quad (13)$$

$$t_n N_{3x3} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} - \begin{bmatrix} x_0 \\ x_0 \\ x_0 \end{bmatrix} \right) \quad (14)$$

$$N_{3x3} = \begin{bmatrix} 0.436 & -0.281 & 0.121 \\ 0.614 & 0.064 & 0.046 \\ 0.603 & 0.230 & 0.167 \end{bmatrix} \quad (15)$$

$$t_n N_{4x4} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} - \begin{bmatrix} x_0 \\ x_0 \\ x_0 \\ x_0 \end{bmatrix} \right) \quad (16)$$

$$N_{4x4} = \begin{bmatrix} 0.278 & -0.202 & 0.169 & -0.071 \\ 0.398 & 0.069 & 0.064 & -0.031 \\ 0.387 & 0.234 & 0.278 & -0.071 \\ 0.389 & 0.222 & 0.389 & 0.000 \end{bmatrix} \quad (17)$$

$$t_n N_{5x5} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \end{bmatrix} = \left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} - \begin{bmatrix} x_0 \\ x_0 \\ x_0 \\ x_0 \\ x_0 \end{bmatrix} \right) \quad (18)$$

$$N_{5x5} = \begin{bmatrix} 0.191 & -0.147 & 0.139 & -0.113 & 0.047 \\ 0.276 & 0.059 & 0.051 & -0.050 & 0.022 \\ 0.267 & 0.193 & 0.251 & -0.114 & 0.045 \\ 0.269 & 0.178 & 0.384 & 0.032 & 0.019 \\ 0.269 & 0.181 & 0.374 & 0.110 & 0.067 \end{bmatrix} \quad (19)$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \left(\tau \left(t_n N_{5x5}\right)^{-1} + I\right)^{-1} \tau \left(t_n N_{5x5}\right)^{-1} \begin{bmatrix} x_0 \\ x_0 \\ x_0 \\ x_0 \\ x_0 \end{bmatrix} = \begin{bmatrix} 0.791 \\ 0.489 \\ 0.277 \\ 0.171 \\ 0.135 \end{bmatrix} \tag{22}$$

The explicit solution $x(t) = x_0 e^{-\frac{t}{\tau}}$ verifies that the numerical solution results are correct.

6. Conclusions

This tutorial paper serves as a guide to practitioners in relating the common steps typically deployed in linear MPC applications to a comparable procedure for nonlinear applications. The application in this paper is the quadruple tank process that is a basic example of multivariate modeling and control. As a first step, certain parameters of the model are adjusted to fit to PRBS data through dynamic data reconciliation. In a next step, the controller is tuned to provide desirable control responses for set point tracking and disturbance rejection. For both estimation and control cases, alternate squared error and L_1-norm error forms are compared. While the L_1-norm error uses additional variables and equations, it adds only linear equality and inequality constraints. As a final section of the tutorial, the process of converting differential equations into a set of algebraic equations is reviewed. This conversion step is necessary to solve the model and objective function simultaneously with NLP solvers. Along with the tutorial overview, example MATLAB scripts are given in the Appendix as a guide to implement the problems in the text. While this tutorial overview is not an exhaustive review of all available techniques or software, it provides a case study to advance the use of nonlinear models in control practice.

Appendix A. Quadruple Tank Model

The quadruple tank process is represented by 14 differential and algebraic equations (DAEs). The following model is expressed in the APMonitor Modeling Language. This file and others included in the paper are available at APMonitor.com.

```
Model
  Constants
    % gravitational constant (cm/s^2)
    g = 981
    % tank cross-sectional area (cm^2)
    Area[1] = 28
    Area[2] = 32
    Area[3] = 28
    Area[4] = 32
    % relation of level to voltage measurement (V/cm)
    kc = 0.50
  End Constants
```

```

  Parameters
    % relation of input voltage to pump flow rate (cm^3/sec / V)
    km = 10.0, >=3.0, <=20.0  % slope
    kb = 0.0, >=-20.0, <=20.0   % intercept
    % correction factors to fit model to real data
    c13 = 0.071, >0.01, <=0.2  % outlet flow corrections
    c24 = 0.057, >0.01, <=0.2  % outlet flow corrections
    % fractional split to tank 1 vs. tank 4
    gamma[1] = 0.43, >=0, <=1
    % fractional split to tank 2 vs. tank 3
    gamma[2] = 0.34, >=0, <=1
    % voltage to pump A
    v1 = 3, >=0, <=10      % Volt
    % voltage to pump B
    v2 = 3, >=0, <=10      % Volt
  End Parameters

  Variables
    % tank height - diameter = 6 cm,  max height = 20 cm
    h[1] = 12.6, >=1e-5
    h[2] = 13.0, >=1e-5
    h[3] = 4.8 , >=1e-5
    h[4] = 4.9 , >=1e-5
  End Variables

  Intermediates
    % correction factors
    c[1] = c13
    c[2] = c24
    c[3] = c13
    c[4] = c24
    % pump flows
    qa = v1 * km + kb
    qb = v2 * km + kb
    % inlet flows from pumps
    q[1] = gamma[1] * qa
    q[2] = gamma[2] * qb
    q[3] = (1-gamma[2]) * qb
    q[4] = (1-gamma[1]) * qa
    % outlet flows
    out[1:4] = c[1:4] * sqrt(2*g*h[1:4])
    % total inlet flows
    in[1] = q[1] + out[3]
    in[2] = q[2] + out[4]
    in[3] = q[3]
    in[4] = q[4]
  End Intermediates

  Equations
    Area[1:4] * $h[1:4] = in[1:4] - out[1:4]  % $ = differential
  End Equations
End Model
```

Appendix B. Parameter Estimation with a PRBS-Generated Signal

The following MATLAB script details the commands necessary to reproduce the parameter estimation case presented in this paper. The parameter estimation uses two elements including the model file (4tank.apm) and a data file (prbs360.csv). The model file is shown in Appendix A while the data file is available for download from APMonitor.com under the APM MATLAB examples section.

```
% Add path to APM MATLAB libraries (available at APMonitor.com)
addpath('../apm');
% Clear MATLAB
clear all; close all;
% Server and Application name
server = 'http://xps.apmonitor.com'; app = 'prbs';
% Clear previous application
apm(server,app,'clear all');
% load model and data
disp('Loading model'); apm_load(server,app,'4tank.apm');
disp('Loading data'); csv_load(server,app,'prbs360.csv');
% Set up variable classifications for data flow
% Feedforwards - measured process disturbances
apm_info(server,app,'FV','km');
apm_info(server,app,'FV','kb');
apm_info(server,app,'FV','gamma[1]');
apm_info(server,app,'FV','gamma[2]');
apm_info(server,app,'FV','c13');
apm_info(server,app,'FV','c24');
% State variables (for display only)
apm_info(server,app,'SV','h[3]');
apm_info(server,app,'SV','h[4]');
% Controlled variables (for controller design)
apm_info(server,app,'CV','h[1]');
apm_info(server,app,'CV','h[2]');
% imode (1=ss, 2=mpu, 3=rto, 4=sim, 5=est, 6=ctl)
apm_option(server,app,'nlc.imode',5);
% read csv file
apm_option(server,app,'nlc.csv_read',1);
% estimated variable error type (1=L1-norm, 2=Squared Error)
apm_option(server,app,'nlc.ev_type',2);
% time units (1=sec, 2=min, 3=hrs, 4=days, etc)
apm_option(server,app,'nlc.ctrl_units',1);
apm_option(server,app,'nlc.hist_units',2);
% parameters to adjust
apm_option(server,app,'km.status',1);
apm_option(server,app,'km.lower',3);
apm_option(server,app,'km.upper',20);
apm_option(server,app,'kb.status',1);
apm_option(server,app,'kb.lower',-2);
apm_option(server,app,'kb.upper',2);
apm_option(server,app,'gamma[1].status',1);
apm_option(server,app,'gamma[1].lower',0.2);
apm_option(server,app,'gamma[1].upper',0.8);
apm_option(server,app,'gamma[2].status',1);
apm_option(server,app,'gamma[2].lower',0.2);
apm_option(server,app,'gamma[2].upper',0.8);
apm_option(server,app,'c13.status',1);
apm_option(server,app,'c13.lower',0.01);
apm_option(server,app,'c13.upper',0.2);
apm_option(server,app,'c24.status',1);
apm_option(server,app,'c24.lower',0.01);
apm_option(server,app,'c24.upper',0.2);
% measured values for model fitting
apm_option(server,app,'h[1].fstatus',1);
apm_option(server,app,'h[2].fstatus',1);
% solver (1=APOPT, 3=IPOPT)
apm_option(server,app,'nlc.solver',3);
% Solve with APMonitor
apm(server,app,'solve')
% Open web-viewer
apm_web(server,app);
% Retrieve solution (creates solution.csv locally)
solution = apm_sol(server,app);
```

Appendix C. Nonlinear Control of the Quadruple Tank Process

The following MATLAB script details the commands necessary to reproduce the nonlinear controller presented in this paper. The model file is the same as is shown in Appendix A but updated with updated parameters from Table 3. The APM MATLAB libraries are available for download from APMonitor.com.

```
% Add path to APM libraries
addpath('../apm');
% Clear MATLAB
clear all; close all;
% Server and Application Name
server = 'http://xps.apmonitor.com';  app = 'nlc';
% Clear previous application
apm(server,app,'clear all');
% load model with updated parameters
apm_load(server,app,'4tank.apm');
% load future time horizon
csv_load(server,app,'control.csv');
% Set up variable classifications for data flow
% Feedforwards - measured process parameters
apm_info(server,app,'FV','gamma[1]');
apm_info(server,app,'FV','gamma[2]');
% Manipulated variables (for controller design)
apm_info(server,app,'MV','v1');
apm_info(server,app,'MV','v2');
% State variables (for display only)
apm_info(server,app,'SV','h[3]');
apm_info(server,app,'SV','h[4]');
% Controlled variables (for controller design)
apm_info(server,app,'CV','h[1]');
apm_info(server,app,'CV','h[2]');
% steady state initialization
% imode = 3, steady state mode
apm_option(server,app,'nlc.imode',3);
apm(server,app,'solve');
% imode = 6, switch to dynamic control
apm_option(server,app,'nlc.imode',6);
% nodes = 3, internal nodes in the collocation structure (2-6)
apm_option(server,app,'nlc.nodes',3);
% time units (1=sec, 2=min, etc)
apm_option(server,app,'nlc.ctrl_units',1); % controller time units
apm_option(server,app,'nlc.hist_units',2); % units for trending
% read csv file
apm_option(server,app,'nlc.csv_read',1);
% Manipulated variable tuning
apm_option(server,app,'v1.status',1); % turn on v1
apm_option(server,app,'v1.upper',6);  % upper bound
apm_option(server,app,'v1.lower',1);  % lower bound
apm_option(server,app,'v1.dmax',1);   % max move per cycle
apm_option(server,app,'v1.dcost',1);  % movement penalty
apm_option(server,app,'v2.status',1); % turn on v2
apm_option(server,app,'v2.upper',6);  % upper bound
apm_option(server,app,'v2.lower',1);  % lower bound
apm_option(server,app,'v2.dmax',1);   % max move per cycle
apm_option(server,app,'v2.dcost',1);  % movement penalty
% Controlled variable tuning
apm_option(server,app,'h[1].status',1);  % turn on h[1]
apm_option(server,app,'h[1].fstatus',0); % turn off feedback status
apm_option(server,app,'h[1].sphi',10.1); % setpoint high
apm_option(server,app,'h[1].splo',9.9);  % setpoint low
apm_option(server,app,'h[1].tau',10);    % speed of response
apm_option(server,app,'h[2].status',1);  % turn on h[2]
apm_option(server,app,'h[2].fstatus',0); % turn off feedback status
apm_option(server,app,'h[2].sphi',15.1); % setpoint high
```

```
apm_option(server,app,'h[2].splo',14.9); % setpoint low
apm_option(server,app,'h[2].tau',10);    % speed of response
% Set controller mode
apm_option(server,app,'nlc.reqctrlmode',3);
% Run APMonitor
apm(server,app,'solve')
% Open web-viewer
apm_web(server,app);
% Retrieve solution (creates solution.csv locally)
solution = apm_sol(server,app);
```

References

[1] S. Qin, T. Badgwell, A survey of industrial model predictive control technology, Control Engineering Practice 11 (2003) 733–764.

[2] R. Findeisen, F. Allgöwer, L. Biegler, Assessment and future directions of nonlinear model predictive control, Springer-Verlag, Berlin, 2007.

[3] Z. Abul-el-zeet, P. Roberts, Enhancing model predictive control using dynamic data reconciliation, AIChE Journal 48 (2) (2002) 324–333.

[4] M. Liebman, T. Edgar, L. Lasdon, Efficient data reconciliation and estimation for dynamic processes using nonlinear programming techniques, Computers and Chemical Engineering 16 (1992) 963–986.

[5] K. McBrayer, T. Edgar, Bias detection and estimation in dynamic data reconciliation, Journal of Process Control 5 (4) (1995) 285–289.

[6] T. Soderstrom, T. Edgar, L. Russo, R. Young, Industrial application of a large-scale dynamic data reconciliation strategy, Industrial and Engineering Chemistry Research 39 (2000) 1683–1693.

[7] Y. Ramamurthi, P. Sistu, B. Bequette, Control-relevant dynamic data reconciliation and parameter estimation, Computers and Chemical Engineering 17 (1) (1993) 41–59.

[8] W. Shaohua, A. Kevin, T. Harris, K. McAuley, Selection of optimal parameter set using estimability analysis and MSE-based model-selection criterion, International Journal of Advanced Mechatronic Systems 3 (3) (2011) 188–197.

[9] J. Hedengren, K. Allsford, J. Ramlal, Moving horizon estimation and control for an industrial gas phase polymerization reactor, in: Proceedings of the American Control Conference (ACC), New York, NY, 2007, pp. 1353–1358.

[10] K. Johansson, Interaction bounds in multivariable control systems, Automatica 38 (6) (2002) 1045–1051.

[11] T. Raff, S. Huber, Z. K. Nagy, F. Allgöwer, Nonlinear model predictive control of a four tank system: An experimental stability study, in: Proceedings of International Conference Control Applications, Munich, Germany, 2006, pp. 237–242.

[12] E. Gatzke, E. Meadows, C. Wang, F. D. III, Model based control of a four-tank system, Computers and Chemical Engineering 24 (2000) 1503–1509.

[13] K. Johansson, The quadruple-tank process - a multivariable laboratory process with an adjustable zero, IEEE Transcations on Control Systems Technology 8 (3) (2000) 456–465.

[14] I. Drca, Nonlinear model predictive control of the four tank process, Master's thesis, Universidad de Sevilla (2007).

[15] M. Mercangöz, F. D. III, Distributed model predictive control of an experimental four-tank system, Journal of Process Control 17 (3) (2007) 297–308, ¡ce:title¿Special Issue ADCHEM 2006 Symposium¡/ce:title¿. doi:10.1016/j.jprocont.2006.11.003.

[16] I. Alvarado, D. Limon, D. M. noz de la Peña, J. Maestre, M. Ridao, H. Scheu, W. Marquardt, R. Negenborn, B. D. Schutter, F. Valencia, J. Espinosa, A comparative analysis of distributed mpc techniques applied to the hd-mpc four-tank benchmark, Journal of Process Control 21 (5) (2011) 800–815, ¡ce:title¿Special Issue on Hierarchical and Distributed Model Predictive Control¡/ce:title¿. doi:10.1016/j.jprocont.2011.03.003.

[17] I. Landau, R. Lozano, M. M'Saad, A. Karimi, Adaptive Control: Algorithms, Analysis and Applications, 2nd Edition, Communications and Control Engineering, Springer-Verlag, London, 2011.

[18] J. Ramlal, V. Naidoo, K. Allsford, J. Hedengren, Moving horizon estimation for an industrial gas phase polymerization reactor, in: Proc. IFAC Symposium on Nonlinear Control Systems Design (NOLCOS), Pretoria, South Africa, 2007.

[19] L. Biegler, Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes, Society for Industrial and Applied Mathematics and the Mathematical Optimization Society, 2010.

[20] J. Hedengren, Optimization and Analytics in the Oil and Gas Industry, Springer-Verlag Berlin Heidelberg, 2012, Ch. Advanced Process Monitoring, p. submitted.

[21] A. Wächter, L. Biegler, On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming, Mathematical Programming 106 (1) (2006) 25–57.

[22] T. Binder, L. Blank, H. Bock, R. Burlisch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J. Schlöder, O. Stryk, Online Optimization of Large Scale Systems, Springer-Verlag Berlin Heidelberg, 2001, Ch. Introduction to model based optimization of chemical processes on moving horizons, pp. 295–339.

[23] J. Albuquerque, L. Biegler, Decomposition algorithms for on-line estimation with nonlinear DAE models, Computers and Chemical Engineering 21 (3) (1997) 283–299.

[24] M. Diehl, H. G. Bock, J. P. Schlöder, R. Findeisen, Z. Nagy, , F. Allgöwer, Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations, Journal of Process Control 12 (2002) 577–585.

[25] E. Haseltine, J. Rawlings, Critical evaluation of extended kalman filtering and moving-horizon estimation, Ind. Eng. Chem. Res. 44 (8) (2005) 2451–2460.

[26] B. Odelson, M. Rajamani, J. Rawlings, A new autocovariance least-squares method for estimating noise covariances, Automatica 42 (2) (2006) 303–308.

[27] J. Hedengren, T. Edgar, Moving horizon estimation - the explicit solution, in: Proceedings of Chemical Process Control (CPC) VII Conference, Lake Louise, Alberta, Canada, 2006.

[28] B. Spivey, J. Hedengren, T. Edgar, Monitoring of process fouling using first-principles modeling and moving horizon estimation, in: Proc. Texas, Wisconsin, California Control Consortium (TWCCC), Austin, TX, 2009.

[29] M. Darby, M. Nikolaou, J. Jones, D. Nicholson, RTO: An overview and assessment of current practice, Journal of Process Control 21 (2011) 874–884.

[30] T. Soderstrom, Y. Zhang, J. Hedengren, Advanced process control in exxonmobil chemical company: Successes and challenges, in: AIChE National Meeting, Salt Lake City, UT, 2010.

[31] S. Jang, B. Joseph, H. Mukai, Comparison of two approaches to on-line parameter and state estimation of nonlinear systems, Ind. Eng. Chem. Process Des. Dev. 25 (1986) 809–814.

[32] G. Carey, B. Finlayson, Othogonal collocation on finite elements, Chemical Engineering Science 30 (1975) 587–596.

[33] J. Albuquerque, L. Biegler, Decomposition algorithms for on-line estimation with nonlinear models, Computers and Chemical Engineering 19 (10) (1995) 1031–1039.