Computers and Chemical Engineering 70 (2014) 133–148

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

Nonlinear modeling, estimation and predictive control in APMonitor

John D. Hedengren[a,*], Reza Asgharzadeh Shishavan[a], Kody M. Powell[b], Thomas F. Edgar[b]

[a] Department of Chemical Engineering, Brigham Young University, Provo, UT 84602, United States
[b] The University of Texas at Austin, Austin, TX 78712, United States

ARTICLE INFO

Article history:
Received 10 August 2013
Received in revised form 19 April 2014
Accepted 27 April 2014
Available online 9 May 2014

Keywords:
Advanced process control
Differential algebraic equations
Model predictive control
Dynamic parameter estimation
Data reconciliation
Dynamic optimization

ABSTRACT

This paper describes nonlinear methods in model building, dynamic data reconciliation, and dynamic optimization that are inspired by researchers and motivated by industrial applications. A new formulation of the ℓ_1-norm objective with a dead-band for estimation and control is presented. The dead-band in the objective is desirable for noise rejection, minimizing unnecessary parameter adjustments and movement of manipulated variables. As a motivating example, a small and well-known nonlinear multivariable level control problem is detailed that has a number of common characteristics to larger controllers seen in practice. The methods are also demonstrated on larger problems to reveal algorithmic scaling with sparse methods. The implementation details reveal capabilities of employing nonlinear methods in dynamic applications with example code in both MATLAB and Python programming languages.

1. Introduction

Applications of Model Predictive Control (MPC) are ubiquitous in a number of industries such as refining and petrochemicals (Darby and Nikolaou, 2012). Applications are also somewhat common in chemicals, food manufacture, mining, and other manufacturing industries (Qin and Badgwell, 2003). Contributions by Morari and others have extended the MPC applications to building climate control (Gyalistras et al., 2011; Oldewurtel et al., 2010), stochastic systems (Nolde et al., 2008; Oldewurtel et al., 2008), induction motors (Papafotiou et al., 2007), and other fast processes with explicit MPC (Bemporad et al., 2002; Hedengren and Edgar, 2008; Johansen, 2004; Domahidi et al., 2011; Ferreau et al., 2008; Pannocchia et al., 2007). A majority of the applications employ linear models that are constructed from empirical model identification, however, some 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 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.

A purpose of this article is to give implementation details on using nonlinear models in the typical steps of dynamic optimization including (1) model construction, (2) fitting parameters to data, (3) optimizing over a future predictive horizon, and (4) transforming differential equations into sets of algebraic equations. Recent advancements in numerical techniques have permitted the direct application of nonlinear models in control applications (Findeisen et al., 2007), 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 researchers and practitioners. Up to this point, there remain relatively few actual industrial applications of control based on nonlinear models. An objective of this paper is to reduce the barriers to implementation of nonlinear advanced control applications. This is attempted by giving implementation details on the following topics:

- nonlinear model development
- parameter estimation from dynamic data

---

* Corresponding author. Tel.: +1 801 477 7341.
E-mail address: john.hedengren@byu.edu (J.D. Hedengren).

http://dx.doi.org/10.1016/j.compchemeng.2014.04.013
0098-1354/

- model predictive control with large-scale models
- direct transcription for solution of dynamic models

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 Optimization Suite (Hedengren, 2014, 2012), freely available software for solution of linear programming (LP), quadratic programming (QP), nonlinear programming (NLP), and mixed-integer (MILP and MINLP) problems. Several other software platforms can also solve dynamic optimization problems with a variety of modeling systems, solution strategies, and solvers (Cizniar et al., 2005; Houska et al., 2011; Piela et al., 1991; Tummescheit et al., 2010; Simon et al., 2009; Nagy et al., 2007).

Of particular interest for this overview is the transformation of the differential and algebraic equation (DAE) systems into equivalent NLP or MINLP problems that can be solved by large-scale optimizers such as the active set solver APOPT (Hedengren et al., 2012) and the interior point solver IPOPT (Wächter and Biegler, 2006). Specific examples are included in the appendices with commands to reproduce the examples in this paper. Some other examples include applications of computational biology (Abbott et al., 2012), unmanned aerial systems (Sun et al., 2014), chemical process control (Soderstrom et al., 2010), solid oxide fuel cells (Jacobsen et al., 2013; Spivey et al., 2012), industrial process fouling (Spivey et al., 2010), boiler load following (Jensen and Hedengren, 2012), energy storage (Powell et al., 2957; Powell and Edgar, 2011, 2012), subsea monitoring systems (Hedengren and Brower, 2012; Brower et al., 2012, 2013), and friction stir welding of spent nuclear fuel (Nielsen, 2012).

This paper includes a number of innovative techniques for formulating large-scale control and optimization problems. A dead-band is added to well-known ℓ_1-norm objective forms for estimation and optimization. This form is different than the forms previously proposed (Genceli and Nikolaou, 1993; Garcia et al., 1989) in that it specifies a dead-band for noise rejection and move suppression. The formulation allows for batch or periodic control and avoids a separate steady-state target calculation. Similar characteristics to prior work (Nikolaou, 2001) include tuning for speed of response, ranked utilization of manipulated variables (MVs), treatment of controlled variables (CVs) with equal concern, and prioritization among separate sets of MVs and CVs.

The objective form presented here for estimation and control is compared to squared-error or ℓ_2-norm objectives that are reported in the literature. The appendices include concise source code that can be used to reproduce the results or serve as a framework for further applications. The target audience is the practitioner or researcher interested in applying nonlinear estimation and control to nonlinear dynamic applications.

2. Nonlinear modeling

A critical aspect of any controller is obtaining a sufficiently correct model form. The model form may include adjustable parameters that are not directly measurable but can be tuned to match both steady-state and dynamic data. Model identification involves adjustment of parameters to fit process data. Models may be linear or nonlinear, empirical or based on fundamental forms that results from material and energy balances, reaction kinetic mechanisms, or other pre-defined model structure. The foundation of many of these correlations is on equations of motion, individual reaction expressions, or balance equations around a control volume such as *accumulation* = *inlet* − *outlet* + *generation* − *consumption*. In the case of a mole balance, for example, this includes molar flows, reactions, and an accumulation term $\left(\frac{dn_i}{dt} = (n_i)_{in} - (n_i)_{out} - (n_i)_{rxn}\right)$. Model structure may also include constraints such as fixed gain ratios, constraints on compositions, or other bounds that reflect physical realism. Detailing the full range of potential model structures is outside the scope of this document. Eq. (1) is a statement of a general model form that may include differential, algebraic, continuous, binary, and integer variables.

$$0 = f\left(\frac{dx}{dt}, x, y, p, d, u\right) \tag{1a}$$

$$0 = g(x, y, p, d, u) \tag{1b}$$

$$0 \leq h(x, y, p, d, u) \tag{1c}$$

The solution of Eq. (1) 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 $u = (u_0, u_1, \ldots, u_{n-1})$. Likewise, the variables values may be determined from the equations such as differential x or algebraic equations y. The equations include differential f, algebraic g, and inequality constraints h. The inequality constraints are included to model physical phenomena such as phase changes where complementarity conditions are required. It is important that the differential terms dx/dt be expressed in implicit form as shown in Eq. (1a) because some models cannot be rearranged into semi-explicit form such as $dx/dt = f(x, y, p, d, u)$. With the methods for solving DAEs demonstrated in Section 5, consistent initial conditions are not required and higher index DAEs are solvable without differentiating the high index algebraic expressions (Biegler, 2007). An example of this capability for both inconsistent initial conditions and high index DAEs is given by a pendulum application (Hedengren, 2014). The pendulum equations of motion are written as index-0 (ODE), index-1, index-2, and index-3 DAEs and solvable with this approach. The drawback of this approach is that the problem size is generally large, requiring the use of sparse methods and highly efficient solvers. Also, a suitable initial guess for the state trajectories is often required for solver convergence.

To implement Eq. (1) within the APMonitor Modeling Language, the following sections are defined with example values for each of the constants, parameters, variables, intermediates, and equations as shown in Listing 1. In the above example, values are defined with optional constraints and initial conditions. The sample model describes a simple objective function $\min(x-5)^2$ and a linear, first-order equation $\tau(dx/dt) = -x + y$ that dynamically relates the input y to the output x. The intermediate variable y is defined as $y = Ku$ to simplify the implicit expression below. The above model is of no specific practical importance but is used to demonstrate the modeling format for differential and algebraic equations. The model is compiled at run-time to provide sparse first and second derivatives of the objective function and equations to solvers through well-known automatic differentiation techniques (Barth et al., 2008).

Listing 1. First order linear model in APMonitor

```
Model
  % Values that remain constant
  Constants
    K = 2                   % Model Constant
  End Constants

  % Values specified by the user or optimizer
  Parameters
    tau = 2, >= 1, <= 10    % Model Parameter
    u   = 3, <= 100         % Manipulated Variable
  End Parameters

  % Implicitly solved variables
  Variables
    x                       % Controlled Variable
  End Variables

  % Explicit definition of temporary variables
  Intermediates
    y = K * u               % Define Variable and Equation
  End Intermediates

  % Inequalities, equalities, algebraic or differential  equations
  Equations
    tau * $x = -x + y       % First-order differential equation
  End Equations
End Model
```

3. Nonlinear dynamic estimation

Along with model form, the objective function is important to ensure desirable results. A common objective form is the least squares form: $(y_{model} - y_{measured})^2$ (see Eq. (2)). 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 (Abul-el-zeet and Roberts, 2002; Liebman et al., 1992; McBrayer and Edgar, 1995; Soderstrom et al., 2000; Ramamurthi et al., 1993).

Table 1 details the equations of the typical squared error norm and the ℓ_1-norm objective. The ℓ_1-norm formulation in Eq. (3) is less sensitive to data outliers and adjusts parameter values only when measurements are outside of a noise dead-band. A small penalty on Δp (change in the parameter values) also discourages parameter movement without sufficient improvement in the model predictions. The change can be from an initial guess or the prior estimates from a Moving Horizon Estimation (MHE) approach. The ℓ_1-norm is similar to an absolute value function but is instead formulated 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. Without the dead-band ($db = 0$) in Eq. (3), the equations for c_U, c_L are not required and the form reduces to the commonly known ℓ_1-norm for estimation that has desirable performance for outlier elimination (Albuquerque and Biegler, 1996; Arora and Biegler, 2004; Biegler and Arora, 2001; Gatzke and Doyle, 2002; Mahadevan and Doyle, 2004; Voelker et al., 2013).

Pseudo-random binary signals (PRBS) are a popular technique to generate linear plant response models from data (Landau et al., 2011). The example problem in Section 6.1 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 (Ramlal et al., 2007). 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 ($(\tau(dy/dt) = -y + Ku)$ 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 with the simultaneous solution approach is that the data reconciliation problem can be very large. The data reconciliation problem is large because a discretization point of the DAE 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 in the prediction horizon. On the other hand, the sequential solution approach (solving objective function and model equations successively) reduces the number of variables that must be solved simultaneously (Binder et al., 2001). This approach is better suited to systems that have a small number of decision variables yet large number of model variables or a long time horizon.

Other challenges in aligning the model to measured values 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 (Shaohua et al., 2011). One solution to automatically eliminate parameters with little sensitivity to the objective is to impose a small penalty on parameter movement from a nominal value (Hedengren et al., 2007). This approach automatically prevents unnecessary movement of parameter values that have little effect on the results of the parameter estimation.

4. Nonlinear control and optimization

There are many challenges to the application of DAEs directly in nonlinear control and optimization (Biegler et al., 2012). Recent advances include simultaneous methods (Binder et al., 2001), decomposition methods (Albuquerque and Biegler, 1997; Diehl et al., 2002), efficient nonlinear programming solvers (Wächter

Table 1
Estimation: two forms for dynamic data reconciliation.

Estimation with a squared error objective

$$\begin{aligned}
\min_{x,y,p,d} \Phi &= (y_x - y)^T W_m (y_x - y) + \Delta p^T c_{\Delta p} + \left(y - \hat{y}\right)^T W_p \left(y - \hat{y}\right) \\
\text{s.}t. \quad 0 &= f\left(\frac{dx}{dt}, x, y, p, d, u\right) \\
0 &= g(x, y, p, d, u) \\
0 &\le h(x, y, p, d, u)
\end{aligned} \tag{2}$$

Estimation with an ℓ_1-norm objective with dead-band

$$\begin{aligned}
\min_{x,y,p,d} \Phi &= w_m^T (e_U + e_L) + w_p^T (c_U + c_L) + \Delta p^T c_{\Delta p} \\
\text{s.}t. \quad 0 &= f\left(\frac{dx}{dt}, x, y, p, d, u\right) \\
0 &= g(x, y, p, d, u) \\
0 &\le h(x, y, p, d, u) \\
e_U &\ge y - y_x + \frac{db}{2} \\
e_L &\ge y_x - \frac{db}{2} - y \\
c_U &\ge y - \hat{y} \\
c_L &\ge \hat{y} - y \\
e_U, e_L, c_U, c_L &\ge 0
\end{aligned} \tag{3}$$

Nomenclature for Eqs. (2) and (3)

Φ	objective function
y_x	measurements $(y_{x,0}, \ldots, y_{x,n})^T$
y	model values $(y_0, \ldots, y_n)^T$
$\hat{y}$	prior model values $\left(\hat{y}_0, \ldots, \hat{y}_n\right)^T$
w_m, W_m	measurement deviation penalty
w_p, W_p	penalty from the prior solution
$c_{\Delta p}$	penalty from the prior parameter values
db	dead-band for noise rejection
x, u, p, d	states (x), inputs (u), parameters (p), or unmeasured disturbances (d)
Δp	change in parameters
f, g, h	equation residuals, output function, and inequality constraints
e_U, e_L	slack variable above and below the measurement dead-band
c_U, c_L	slack variable above and below a previous model value

Table 2
Control: two objective forms for nonlinear dynamic optimization.

Control squared error objective

$$\begin{aligned}
\min_{x,y,u} \Phi &= (y - y_t)^T W_t (y - y_t) + y^T c_y + u^T c_u + \Delta u^T c_{\Delta u} \\
\text{s.}t. \quad 0 &= f\left(\frac{dx}{dt}, x, y, p, d, u\right) \\
0 &= g(x, y, p, d, u) \\
0 &\le h(x, y, p, d, u) \\
\tau_c \frac{dy_t}{dt} + y_t &= sp
\end{aligned} \tag{4}$$

Control ℓ_1-norm objective

$$\begin{aligned}
\min_{x,y,u} \Phi &= w_{hi}^T e_{hi} + w_{lo}^T e_{lo} + y^T c_y + u^T c_u + \Delta u^T c_{\Delta u} \\
\text{s.}t. \quad 0 &= f\left(\frac{dx}{dt}, x, y, p, d, u\right) \\
0 &= g(x, y, p, d, u) \\
0 &\le h(x, y, p, d, u) \\
\tau_c \frac{dy_{t,hi}}{dt} + y_{t,hi} &= sp_{hi} \\
\tau_c \frac{dy_{t,lo}}{dt} + y_{t,lo} &= sp_{lo} \\
e_{hi} &\ge y - y_{t,hi} \\
e_{lo} &\ge y_{t,lo} - y
\end{aligned} \tag{5}$$

Nomenclature for Eqs. (4) and (5)

Φ	objective function
y	model values $(y_0, \ldots, y_n)^T$
$y_t, y_{t,hi}, y_{t,lo}$	desired trajectory target or 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
u, x, p, d	inputs (u), states (x), parameters (p), and disturbances (d)
f, g, h	equation residuals (f), output function (g), and inequality constraints (h)
τ_c	time constant of desired controlled variable response
e_{lo}, e_{hi}	slack variable below or above the trajectory dead-band
sp, sp_{lo}, sp_{hi}	target, lower, and upper bounds to final setpoint dead-band

and Biegler, 2006), improved estimation techniques (Haseltine and Rawlings, 2005; Odelson et al., 2006; Hedengren and Edgar, 2006; Spivey et al., 2009), and experience with applications to industrial systems (Hedengren et al., 2007; Darby et al., 2011). In particular, 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 reliability, operator training, and management support are critical factors in the success of an application (Soderstrom et al., 2010).

A common objective function form is the squared error or ℓ_2-norm objective (see Eq. (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 Eq. (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. Lower ranking objectives are met as degrees of freedom remain. However, the best objective function will always be met by minimizing the error associated with high ranking objectives. For problems that have safety, environmental, economic, and other competing priorities, the ℓ_1-norm with a dead-band gives an intuitive form that manages these trade-offs as shown in Fig. 1.

Priorities are assigned by giving the highest weighting (w_{hi}, w_{lo}) to the most important objectives. For the hypothetical pressure control example in Fig. 1 the safety constraint is never violated (highest priority). The economic target (lowest priority) is only satisfied when the other constraints are also satisfied from 0–2 min and drives the response along the upper limit of the environmental constraint from 2–5 min. When the environmental constraint (second highest priority) is violated, the response is driven to the lower limit of the safety constraint to have the least penalty for the environmental violation from 5–10 min. This dead-band also gives flexibility to have non-symmetric objective functions in cases

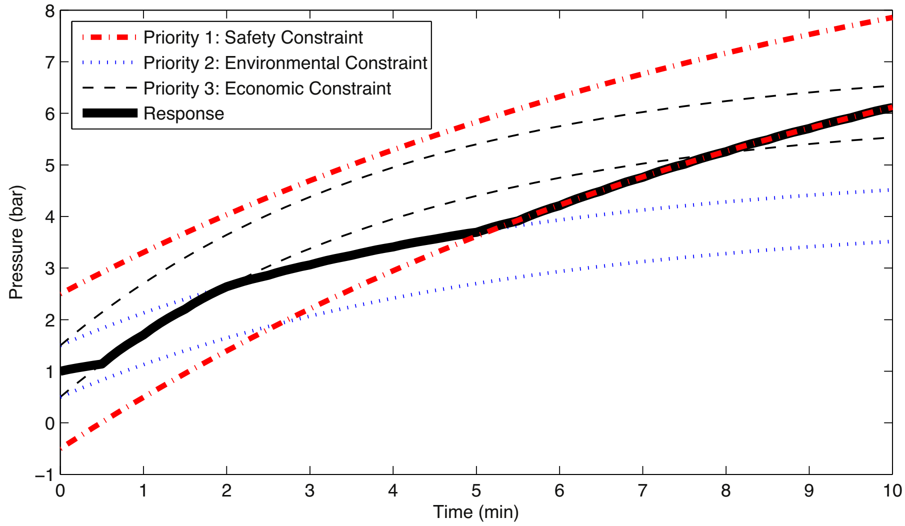


Fig. 1. Competing priorities with safety, environmental, and economic ranges.

where an upper or lower limit is more important. Table 2 details the square error and ℓ_1-norm objective functions.

The reference trajectories in both the squared-error and ℓ_1-norm moderate the speed at which the controller attempts to reach the desired setpoint *sp* or reach the desired range sp_{lo}, sp_{hi} as shown in Fig. 2. Three different ℓ_1-norm trajectories are shown with varying initial conditions and are classified as a reference trajectory (inner-most), a pure dead-band (constant band), and a funnel trajectory (widest at the beginning). The initial conditions for $y_{t,hi}$ and $y_{t,lo}$ adjust the starting positions of the reference trajectory region of no penalty. For dead-band control, the initial conditions are set to the final target values with $y_{t,hi} = sp_{hi}$ and $y_{t,lo} = sp_{lo}$. If restrictions on near-term dynamics are less important than reaching a target steady-state value, the gap between $y_{t,hi}$ and $y_{t,lo}$ can be made large relative to the range of the final dead-band sp_{hi} and sp_{lo} as shown by the funnel trajectory in Fig. 2.

5. Numerical solution of DAE systems

Two types of methods for solving nonlinear MPC and dynamic optimization problems include sequential methods and simultaneous methods (Binder et al., 2001). With the more compact sequential approach, the model equations are repeatedly solved to convergence tolerance to provide an objective function and gradient. The supervisory layer then proposes new decision variables and the simulation process is repeated. Conversely, the simultaneous approach involves solving the model equations and optimizing the objective function in parallel.

Sequential methods are easier to implement, but may fail to converge in a reasonable time for problems with a large number of degrees of freedom, thus delivering sub-optimal solutions. However, because sequential methods solve the model equations by forward integration, the solutions are always feasible with respect to the dynamic model, if not optimal. The simultaneous solution approach may be advantageous for certain problems, especially boundary value problems, terminal time constraints, and systems with unstable modes (Biegler, 2007). Simultaneous optimization approaches generally have a computational advantage for control problems with many decision variables but with a moderate number of state variables. Sequential approaches may have computational advantage for a small number of decision variables coupled with large-scale models. Typical cases of large-scale models are distributed parameter systems. In this case, the computational gain obtained through simultaneous methods from the elimination of repeated integration is overcome by the very large number of space and time discretized states.

A characteristic of the simultaneous problem formulation is that a general DAE model can be posed in open equation format (refer to Eq. (1)). In open equation format, DAE models of index-1 or higher are solved without rearrangement or differentiation. The values of certain parameters, disturbances, or decision variables are discrete values over the time horizon to make the problem tractable for

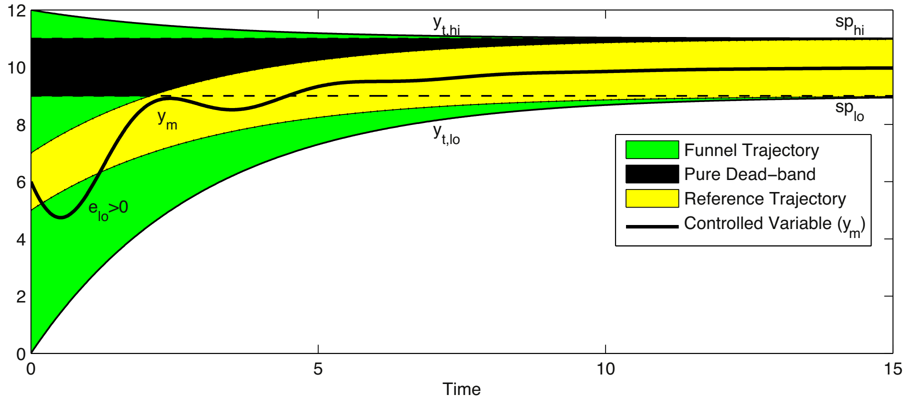


Fig. 2. Three examples of ℓ_1-norm dead-band trajectory regions for model predictive control.

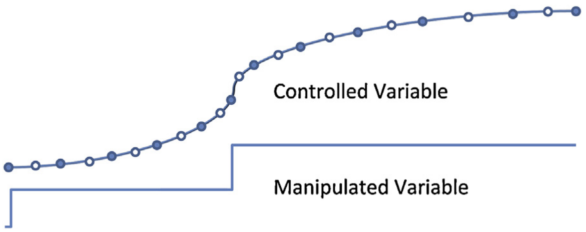


Fig. 3. Dynamic equations are discretized over a time horizon and solved simultaneously. The solid nodes depict starting and ending locations for local polynomial approximations that are pieced together over the time horizon. With one internal node for each segment, this example uses a 2nd order polynomial approximation for each step.

numerical solution (e.g. MVs in Fig. 3). On the other hand, integrated variables are determined from differential and algebraic equations and generally have a continuous profile (e.g. CVs in Fig. 3).

One solution approach to this dynamic system is the conversion of the DAE system to algebraic equations through direct transcription (Findeisen et al., 2007). This technique is also known as orthogonal collocation on finite elements (Carey and Finlayson, 1975). Converting the DAE system to a Nonlinear Programming (NLP) problem permits the solution by large-scale solvers (Liebman et al., 1992; Albuquerque and Biegler, 1995). Additional details of the simultaneous approach are shown in Section 5.1 and an example problem in Section 5.2.

5.1. 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 $1,\ldots,n$ as shown in Eq. (6). In the case of Eq. (6), four points are shown for the derivation. The initial value, x_0, is a fixed initial condition or otherwise equal to the final point from the prior interval.

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix} = M\left(\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix} - \begin{bmatrix}x_0\\ x_0\\ x_0\end{bmatrix}\right) \tag{6}$$

The solution of the differential equations at discrete time points is approximated by a Lagrange interpolating polynomial as shown in Eq. (7).

$$x(t) = A + Bt + Ct^2 + Dt^3 \tag{7}$$

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 $t_1 = 1/2$. An example of internal nodes are displayed in Fig. 3 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, Eq. (7) is differentiated and substituted into Eq. (6) to give Eq. (8). Note that the A coefficient from Eq. (7) is cancelled by x_0 on the right-hand side of Eq. (8).

$$\begin{gathered}\begin{bmatrix}B + 2Ct_1 + 3Dt_1^2\\ B + 2Ct_2 + 3Dt_2^2\\ B + 2Ct_3 + 3Dt_3^2\end{bmatrix} = M\begin{bmatrix}Bt + Ct_1^2 + Dt_1^3\\ Bt + Ct_2^2 + Dt_2^3\\ Bt + Ct_3^2 + Dt_3^3\end{bmatrix}\\ \begin{bmatrix}1 & 2t_1 & 3t_1^2\\ 1 & 2t_2 & 3t_2^2\\ 1 & 2t_3 & 3t_3^2\end{bmatrix}\begin{bmatrix}B\\ C\\ D\end{bmatrix} = M\begin{bmatrix}t_1 & t_1^2 & t_1^3\\ t_2 & t_2^2 & t_2^3\\ t_3 & t_3^2 & t_3^3\end{bmatrix}\begin{bmatrix}B\\ C\\ D\end{bmatrix}\end{gathered} \tag{8}$$

Finally, rearranging and solving for M gives the solution shown in Eq. (9).

$$M = \begin{bmatrix}1 & 2t_1 & 3t_1^2\\ 1 & 2t_2 & 3t_2^2\\ 1 & 2t_3 & 3t_3^2\end{bmatrix}\begin{bmatrix}t_1 & t_1^2 & t_1^3\\ t_2 & t_2^2 & t_2^3\\ t_3 & t_3^2 & t_3^3\end{bmatrix}^{-1} \tag{9}$$

The final form that is implemented in practice is shown in Eq. (10) by inverting M and factoring out the final time t_n ($t_n N = M^{-1}$). This form improves the numerical characteristics of the solution, especially as the time step approaches zero ($t_n \to 0$).

$$t_n N\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix} = \begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix} - \begin{bmatrix}x_0\\ x_0\\ x_0\end{bmatrix} \tag{10}$$

The matrices that relate $\frac{dx}{dt}$ to x are given in Tables A.6 and A.7 in Appendix A for intervals with 3–6 nodes.

5.2. Example solution by orthogonal collocation

A simultaneous solution demonstrates the application of orthogonal collocation. In this case, the first order system $\tau(dx/dt) = -x$ is solved at 6 points from $t_0 = 0$ to $t_n = 10$ using Eq. (A.4). In this case $\tau = 5$ and the initial condition is specified at $x_0 = 1$. For this problem, the time points for (dx/dt) 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 (dx/dt) are generated in Eq. (11).

$$\frac{dx}{dt} = \begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\\ \dot{x}_5\end{bmatrix} = (t_n N_{5x5})^{-1}\left(\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\\ x_5\end{bmatrix} - \begin{bmatrix}x_0\\ x_0\\ x_0\\ x_0\\ x_0\end{bmatrix}\right) \tag{11}$$

Substitution of Eq. (11) into the derivatives of the model equation yields a linear system of equations as shown in Eq. (12).

$$\begin{gathered}\tau\frac{dx}{dt} = -x\\ \tau(t_n N_{5x5})^{-1}\left(\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\\ x_5\end{bmatrix} - \begin{bmatrix}x_0\\ x_0\\ x_0\\ x_0\\ x_0\end{bmatrix}\right) = -\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\\ x_5\end{bmatrix}\end{gathered} \tag{12}$$

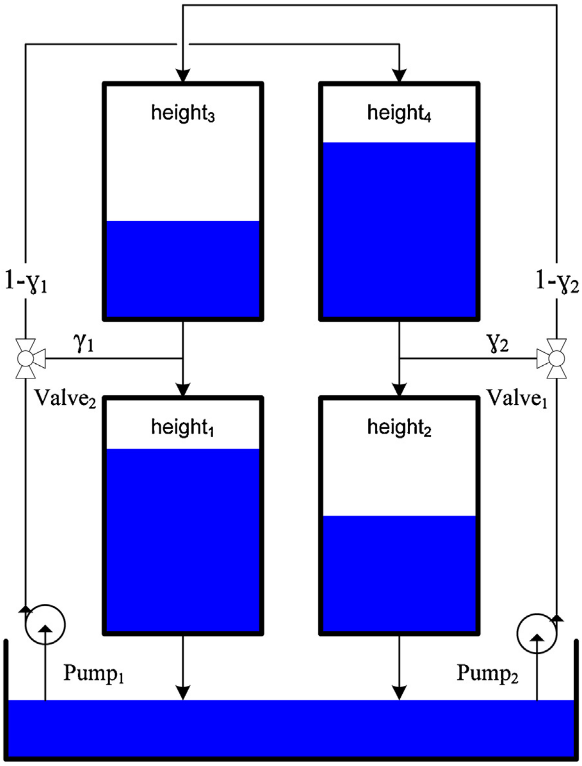


Fig. 4. Diagram of the quadruple tank process. Pump 1 supplies tanks 1 and 4 while pump 2 supplies tanks 2 and 3.

Eq. (12) is rearranged and solved with linear algebra as shown in Eq. (13).

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \left(\tau(t_n N_{5x5})^{-1} + I\right)^{-1} \tau(t_n N_{5x5})^{-1} \begin{bmatrix} x_0 \\ x_0 \\ x_0 \\ x_0 \\ x_0 \end{bmatrix} = \begin{bmatrix} 0.791 \\ 0.489 \\ 0.277 \\ 0.171 \\ 0.135 \end{bmatrix} \quad (13)$$

The numerical solution given in Eq. (13) is within three significant figures of the analytical solution $x(t) = x_0 e^{-t/\tau}$, verifying that the numerical solution approximations are sufficiently accurate in this case. This is not always the case and discretization must sometimes be refined to reduce numerical error.

6. Application: quadruple tank level control

A quadruple tank process shown in Fig. 4 has been the subject of theoretical (Johansson, 2002) and practical demonstrations (Raff et al., 2006; Gatzke et al., 2000; Johansson, 2000; Drca, 2007) of a multivariable and highly coupled system (Gatzke et al., 2000). The four tank process has also been a test application for application of decentralized and coordinated control techniques (Mercangöz and Doyle, 2007; Alvarado et al., 2011). 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 unstable poles in a transfer function representation of the system. Another challenge is the nonlinear tendency of the system. For example, this can be characterized by variable gains of the MVs to the CVs.

The four tank process has two pumps that are adjusted with variable voltage to pump 1 (υ_1) and pump 2 (υ_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 MVs and CVs. 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 (Johansson, 2002). A RHP zero indicates that there may either be overshoot or an inverse response to a step change in the MV.

A combination of material balances and Bernoulli's law yields the process model for the four tank process as shown in Eq. (14). The equations are also displayed in Appendix B in the APMonitor Modeling Language.

$$\begin{aligned} q_a &= k_m \upsilon_1 + k_b \\ q_b &= k_m \upsilon_2 + k_b \end{aligned} \tag{14a}$$

$$\begin{aligned} q_{1,in} &= \gamma_1 q_a + q_{3,out} \\ q_{2,in} &= \gamma_2 q_b + q_{4,out} \\ q_{3,in} &= (1-\gamma_2) q_b \\ q_{4,in} &= (1-\gamma_1) q_a \end{aligned} \tag{14b}$$

$$\begin{aligned} q_{1,out} &= c_1 \sqrt{2gh_1} \\ q_{2,out} &= c_2 \sqrt{2gh_2} \\ q_{3,out} &= c_3 \sqrt{2gh_3} \\ q_{4,out} &= c_4 \sqrt{2gh_4} \end{aligned} \tag{14c}$$

$$\begin{aligned} A_1 \frac{dh_1}{dt} &= q_{1,in} - q_{1_{out}} \\ A_2 \frac{dh_2}{dt} &= q_{2,in} - q_{2_{out}} \\ A_3 \frac{dh_3}{dt} &= q_{3,in} - q_{3_{out}} \\ A_4 \frac{dh_4}{dt} &= q_{4,in} - q_{4_{out}} \end{aligned} \tag{14d}$$

where γ_1 is the split factor for tanks 1 and 4 and γ_2 is the split factor leading to tanks 2 and 3 and the range of allowable values is $0 \le \gamma_i \le 1$. When $\gamma_i = 0$ all of the flow from the pumps enters the top tanks (3 or 4) and when $\gamma_i = 1$ all of the flow enters the lower tanks (1 or 2). The other parameters for this model include c_i as the outflow factor for tank i, k_m as the valve linearization slope, k_b as the valve linearization intercept, and A_i as the cross-sectional area of tank i. The variables include q_a as the flow from pump 1, q_b as the flow from pump 2, $q_{i,in}$ as the inlet flow to tank i, $q_{i,out}$ as the outlet flow from tank i, and h_i as the height of liquid in tank i.

Eq. set (14a) is the relationship between pump voltage and flow while Eq. set (14b) defines the inlet flow to each of the tanks. Eq. set (14c) is the outlet flow from each of the tanks with tanks 3 and 4 draining to tanks 1 and 2, respectively. Finally, equation set (14d) is a material balance around each tank with accumulation, inlet, and outlet terms. In this case, the density is assumed to be constant allowing a volumetric balance to be used instead.

Table 3
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 s	10.3 s
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

Table 5
Changing parameter results with corrupted data.

	γ_1	γ_2	$c_{1,3}$	$c_{2,4}$	k_m	k_b
Parameter value change with ℓ_1-norm						
Case 1 (Outlier)	0%	0%	0%	0%	0%	0%
Case 2 (Drift)	1%	1%	2%	0%	1%	0%
Case 3 (Noise)	5%	2%	8%	4%	2%	21%
Parameter value change with squared error						
Case 1 (Outlier)	11%	6%	3%	4%	6%	42%
Case 2 (Drift)	3%	11%	15%	5%	3%	37%
Case 3 (Noise)	9%	4%	2%	9%	7%	72%

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 28 cm^2 while tanks 2 and 4 have a cross-sectional area of 32 cm^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.

6.1. 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 additional equations for the differential states in the optimization problem from the orthogonal collocation transformation (see Section 5). Direct transcription by orthogonal collocation on finite elements is one of the methods to convert DAE systems into a Nonlinear Programming (NLP) problem (Biegler, 2010). This is accomplished by approximating time derivatives of the DAE system as algebraic relationships as discussed previously. Fig. 5 shows the results of the reconciliation to the PRBS-generated data.

Only levels for tanks 1 and 2 are measured as shown in Fig. 5. 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 solution overview is shown in Table 3 while initial and final values of the parameters are displayed in Table 4. MATLAB and Python scripts for configuring and solving this problem are shown in Listing 3 of Appendix C. The MATLAB or Python scripts use the APMonitor Modeling Language (Hedengren, 2014) model (see Appendix B) to create the differential and algebraic (DAE) model. APMonitor translates the problem into an NLP and solves the equations with one of many large-scale solvers. The particular solver used in this study is IPOPT, an interior point large-scale nonlinear programming solver (Wächter and Biegler, 2006), for solving the resulting optimization problem. A summary of the optimization problem and the solution is shown in Table 3.

Using different objective function forms resulted in similar parameter estimates and comparable model predictions. As seen in Table 4, the optimal values for the parameters were well within the upper and lower constraints. These constraints were set for both ℓ_1-norm and squared-error problems 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 had roughly twice the number of variables and required 3 times the amount of CPU time to find a solution. In this case, the increased computational time is an additional cost associated with ℓ_1-norm estimation.

Improved outlier rejection and parameter estimates are shown by purposefully introducing corrupted data. Three cases are shown in Fig. 6 with the corrupted data being introduced at 1200 s.

The first case of corrupted data is a single outlier that is 10 cm higher than the actual measured value. While this specific outlier could easily be removed by automated outlier detection, it may not be possible to eliminate all outliers from data especially for real-time or large-scale systems. A second case involves measurement drift at a rate of +0.1 cm per second. After 550 s, the measurement drift is corrected and the measurement returns to actual measured values. A third case introduces normally distributed measurement noise with zero mean and standard deviation of one (Table 5).

For all cases, it is desirable to retain original parameters even in the presence of corrupted data. The ℓ_1-norm form outperforms the squared-error form in two of the three cases and slightly better on the case with added noise. In the case of the single outlier, the ℓ_1-norm parameter values do not change, demonstrating the value in rejecting outlier values. In the case of measurement drift, the ℓ_1-norm error parameters change by from 0-2% while the squared-error parameters change between 3-37%. Finally, for the measurement noise case, the ℓ_1-norm and squared-error parameters both change although the squared-error parameters change by roughly twice that of the ℓ_1-norm parameters. This corrupted data example demonstrates the ability of the ℓ_1-norm to better reject outliers, sensor drift, and noise.

6.2. Nonlinear optimization of the quadruple tank system

Continuing with the quadruple tank example, the squared error model parameters from Section 6.1 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 NMPC application to better predict the future response.

Table 4
Results of the dynamic data reconciliation.

Initial and final values of the estimation problem					
Parameter	Initial value	Lower bound	Upper bound	ℓ_1-Norm results	Squared error results
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

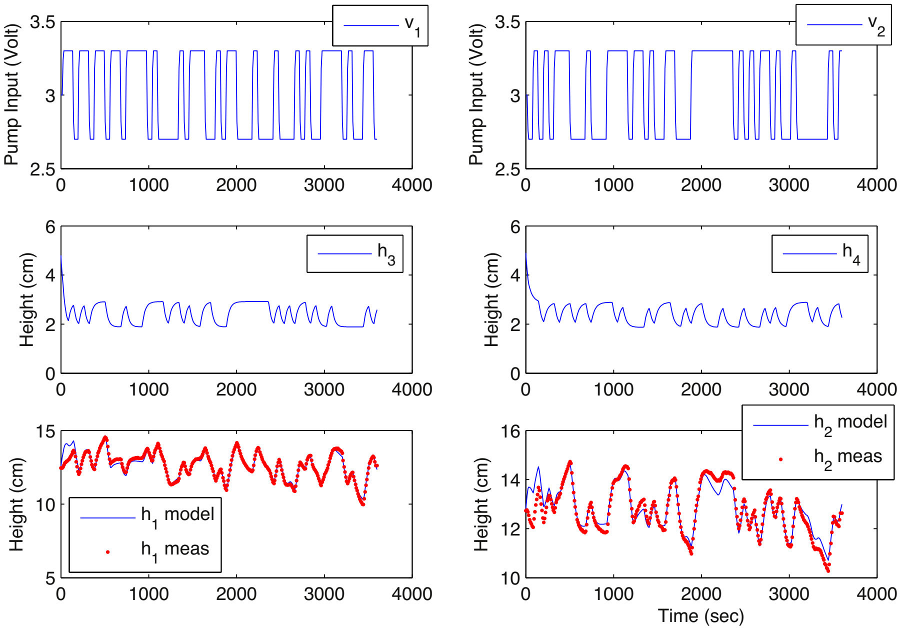


Fig. 5. Results of the dynamic parameter estimation using PRBS generated data.

Once the model is updated, nonlinear optimization calculates the optimal trajectory of the MV. In this case, a future move plan of the voltage to the two pumps is calculated as shown in Fig. 7. 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 $\left|\Delta MV\right| \leq 1$. With a cycle time of 1 second, the rate that the voltage to the pump can change is $\pm 1 V/sec$. 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 s 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 MVs is typical for 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 MVs must over-react to move the process faster. In most cases, steady state values of the MVs 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 MVs. 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. Other MPC ℓ_1-norm formulations have particular drawbacks that either lead to dead-beat or idle control performance (Rao and Rawlings, 2000).

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

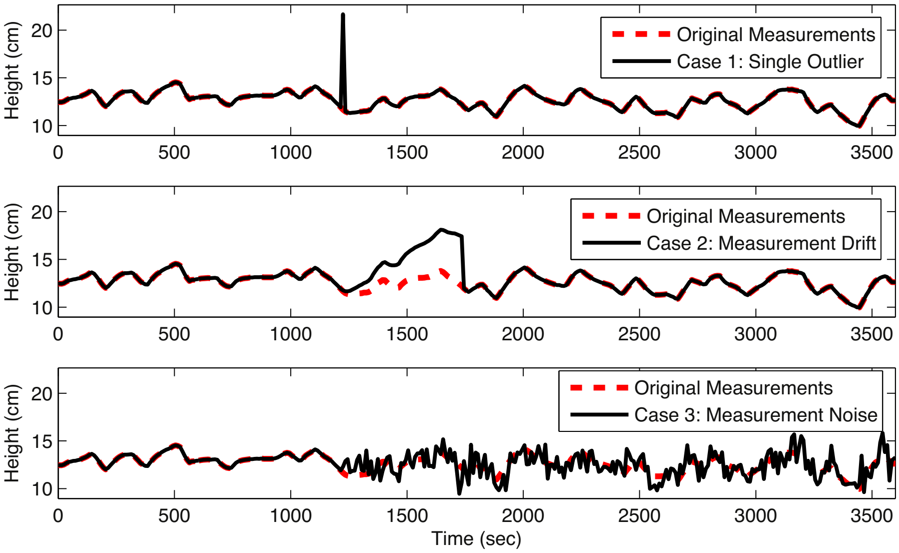


Fig. 6. Three cases of corrupted data with (1) single outlier, (2) measurement drift, and (3) measurement noise.

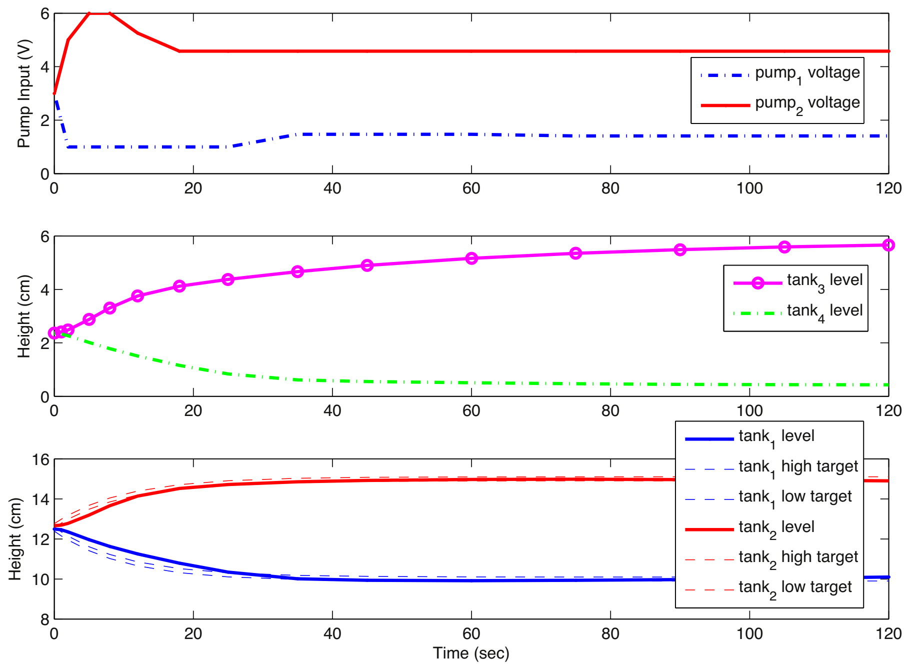


Fig. 7. Model predictive control solution showing voltage input to the pumps 1 and 2.

order reference trajectory with time constant τ_c. Only values that are outside this dead-band are penalized in the objective 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. The bottom subplot of Fig. 7 displays the CV response along with the upper and lower trajectories that define the control objective.

7. Large-scale systems

The quadruple tank system is a small-scale system that has been included here and in many other benchmark studies to demonstrate control techniques for multi-variable systems. An additional example is the computational requirements for large-scale systems. A test of the scale-up of the simultaneous approach for optimization is presented here with varying problem sizes with a state space model. In particular, the number of MVs and CVs is varied to reveal computational time required to determine an optimal solution for a single cycle of the controller. The controller has a quadratic objective function and linear constraints as shown in Eq. (15).

$$\min_{x\in R^n, y\in R^p, u\in R^m} \Phi = (y-y_t)^T W_t (y-y_t) + y^T c_y + u^T c_u + \Delta u^T c_{\Delta u}$$

$$\begin{aligned} \text{s.t.} \quad & \frac{dx}{dt} = Ax + Bu, \quad A = -I_{n\times n}, \quad B = \begin{bmatrix} 1 & \dots & 1 \\ 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 0 \end{bmatrix}_{n\times m} \\ & y = Cx + Du, \quad C = I_{p\times n}, \quad D = 0_{p\times m} \\ & \tau_c \frac{dy_t}{dt} + y_t = sp \\ & 0 \le u \le 10 \end{aligned} \tag{15}$$

The number of MVs (m) and number of CVs (p) are adjusted to vary the problem size. The controller is configured with $W_t = I_{p\times p}$, $c_y = c_u = c_{\Delta u} = 0_{m\times 1}$, $\tau_c = 1_{p\times 1}$, $sp = 1_{p\times 1}$, and initial condition $x_0 = 0_{n\times 1}$. Each of the MVs affects each of the CVs, leading to a dense step response mapping. The cycle time is assumed to be 6 s

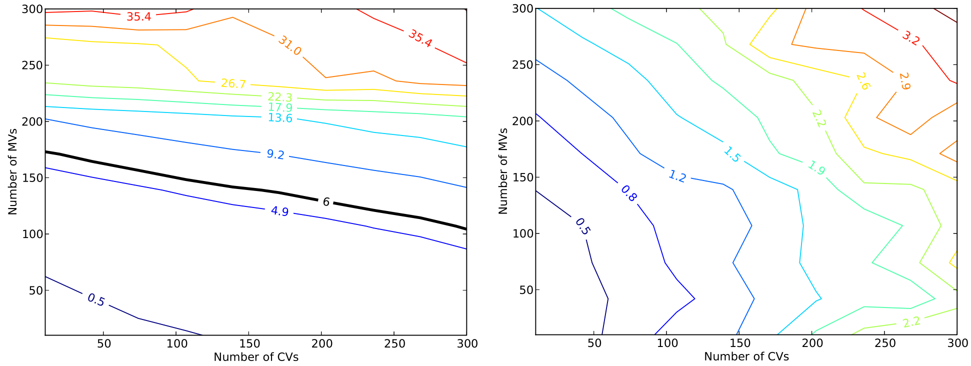


Fig. 8. Contour plot of CPU times for varying numbers of MVs and CVs for APOPT and IPOPT, respectively.

with a prediction horizon of 120 min. The discretization times are chosen as 0, 0.1, 0.2, 0.4, 0.8, 1.5, 3, 6, 12, 25, 50, 60, 80, 100, and 120. The non-uniform time steps allow near-term resolution for control action and long-term predictions for control target calculations. An active set solver (APOPT) and an interior point solver (IPOPT) are tested for the combination of MVs and CVs quantities as shown in Fig. 8.

The APOPT solver has excellent scaling with increased number of CVs but poor computational scaling with increased number of MVs (decision variables). This is expected from an active set solver where the basis selection and active set switching requires intensive matrix operations. Once the correct set of active constraints is determined, the algorithm can rapidly converge to an optimal solution.

The largest case with 300 MVs and 300 CVs translates into an optimization problem with 12,600 variables, 8,400 equations, and 4,200 degrees of freedom (decision variables) because the equations are discretized over the time horizon. Others have also demonstrated large-scale MPC with an ℓ_1-norm objective such a 400 MV/400 CV application to a paper machine cross direction control (Dave et al., 1997, 1999). The present case is solved in 3.8 sec with the IPOPT solver and in 39.5 sec with APOPT solver. A known advantage of interior point solvers is the excellent scaling with additional degrees of freedom. An advantage of active set solvers is the ability to quickly find a solution from a nearby candidate solution. A suggested approach is to use the interior point solver to initialize a problem and switch to an active set method for cycle-to-cycle cases that can be initialized from a prior solution.

8. Conclusions

This paper gives details on the implementation of nonlinear modeling, data reconciliation, and dynamic optimization. The examples relate the common steps typically deployed in linear MPC applications to a comparable procedure for nonlinear applications. As a foundation for using dynamic models, 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. The application in this paper is the quadruple tank process that is a well-known example of multivariate 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. Along with the overview, example MATLAB and Python scripts are given in the Appendix as a guide to implement the problems in the text. While this is not an exhaustive review of all available techniques or software, it provides a platform and case study to advance the use of nonlinear models in control research and practice.

Appendix A. Direct transcription by orthogonal collocation on finite elements

The matrices that relate dx/dt to x are given in Tables A.6 and A.7 for intervals with 3–6 nodes. The formula for 2 nodes reduces to Euler's method for numerical integration differential equations. However, in this case the equations are not solved sequentially in time but simultaneously by an implicit solution method. Additional accuracy can be achieved over one interval with more nodes but more nodes also increases the number of equations and size of the problem. The time dynamic horizon is typically divided over a number of intervals where these equations are applied. An additional set of these equations must be included for every differential variable that appears in the model equations. In this case the differential variables are treated like regular algebraic variables because there is an additional equation for every unknown derivative value at every time point.

Table A.6
Direct transcription to solve differential equations as sets of algebraic equations.

Orthogonal collocation equations	
$t_n N_{2x2} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \begin{bmatrix} x_0 \\ x_0 \end{bmatrix}$	(A.1)
$t_n N_{3x3} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} - \begin{bmatrix} x_0 \\ x_0 \\ x_0 \end{bmatrix}$	(A.2)
$t_n N_{4x4} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} - \begin{bmatrix} x_0 \\ x_0 \\ x_0 \\ x_0 \end{bmatrix}$	(A.3)
$t_n N_{5x5} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} - \begin{bmatrix} x_0 \\ x_0 \\ x_0 \\ x_0 \\ x_0 \end{bmatrix}$	(A.4)

Table A.7
Matrices for direct transcription.

Orthogonal collocation matrices	
$N_{2x2} = \begin{bmatrix} 0.75 & -0.25 \\ 1.00 & 0.00 \end{bmatrix}$	(A.5)
$N_{3x3} = \begin{bmatrix} 0.436 & -0.281 & 0.121 \\ 0.614 & 0.064 & 0.046 \\ 0.603 & 0.230 & 0.167 \end{bmatrix}$	(A.6)
$N_{4x4} = \begin{bmatrix} 0.278 & -0.202 & 0.169 & -0.071 \\ 0.398 & 0.069 & 0.064 & -0.031 \\ 0.387 & 0.234 & 0.278 & -0.071 \\ 0.389 & 0.222 & 0.389 & 0.000 \end{bmatrix}$	(A.7)
$N_{5x5} = \begin{bmatrix} 0.191 & -0.147 & 0.139 & -0.113 & 0.047 \\ 0.276 & 0.059 & 0.051 & -0.050 & 0.022 \\ 0.267 & 0.193 & 0.251 & -0.114 & 0.045 \\ 0.269 & 0.178 & 0.384 & 0.032 & 0.019 \\ 0.269 & 0.181 & 0.374 & 0.110 & 0.067 \end{bmatrix}$	(A.8)

Appendix B. Quadruple tank model

The quadruple tank process is represented by 14 differential and algebraic equations (DAEs). The following model in Listing 2 is expressed in the APMonitor Modeling Language. This file and others included in the paper are available at APMonitor.com as a MATLAB toolbox (Hedengren, 2014a) or as a Python package (Hedengren, 2014b).

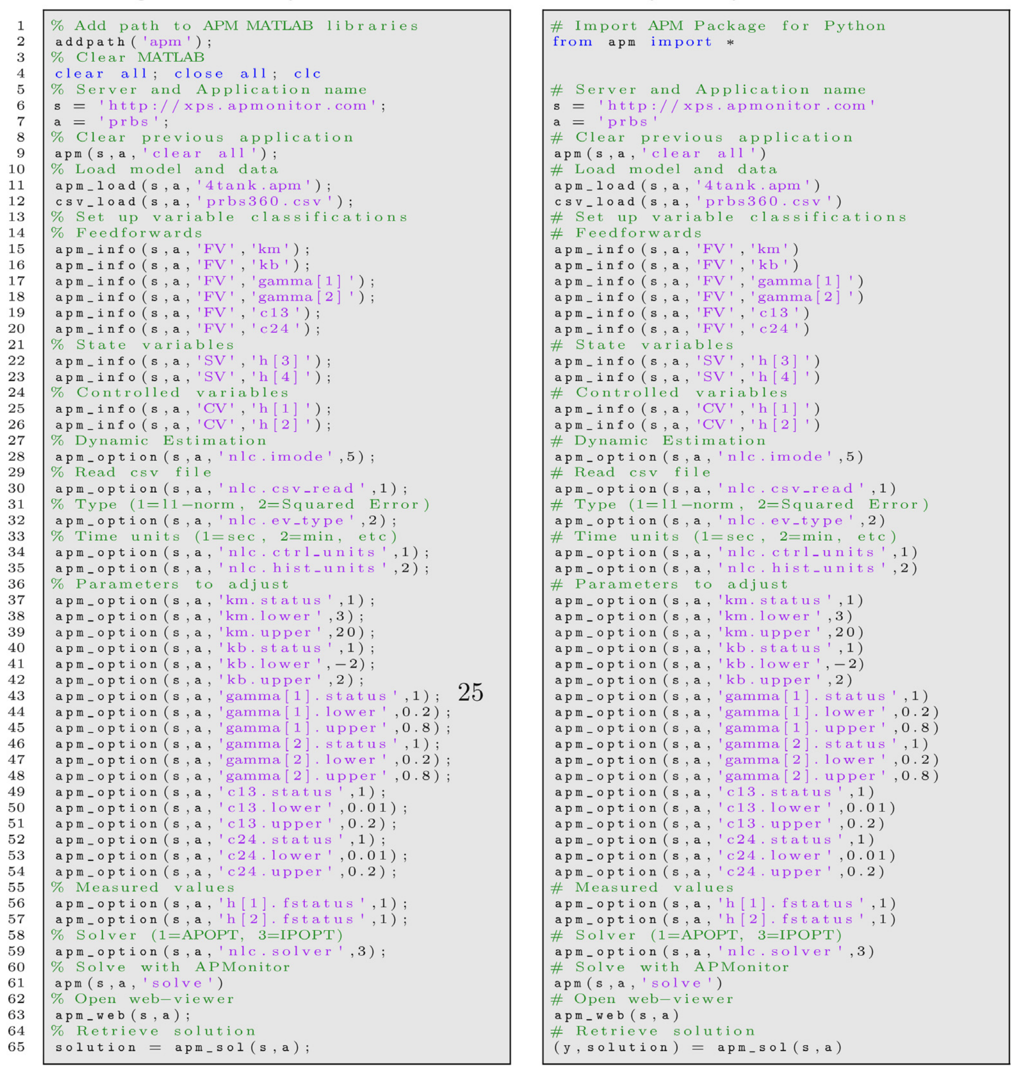
Listing 2. Four tank model in APMonitor.

```
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 C. Parameter estimation with a PRBS-generated signal

The following MATLAB and Python scripts in Listing 3 detail 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 B while the data file is available for download from APMonitor.com under the MATLAB or Python example sections.

Listing 3. MATLAB dynamic estimation.



Listing 3: MATLAB Dynamic Estimation

```
% Add path to APM MATLAB libraries
addpath('apm');
% Clear MATLAB
clear all; close all; clc
% Server and Application name
s = 'http://xps.apmonitor.com';
a = 'prbs';
% Clear previous application
apm(s,a,'clear all');
% Load model and data
apm_load(s,a,'4tank.apm');
csv_load(s,a,'prbs360.csv');
% Set up variable classifications
% Feedforwards
apm_info(s,a,'FV','km');
apm_info(s,a,'FV','kb');
apm_info(s,a,'FV','gamma[1]');
apm_info(s,a,'FV','gamma[2]');
apm_info(s,a,'FV','c13');
apm_info(s,a,'FV','c24');
% State variables
apm_info(s,a,'SV','h[3]');
apm_info(s,a,'SV','h[4]');
% Controlled variables
apm_info(s,a,'CV','h[1]');
apm_info(s,a,'CV','h[2]');
% Dynamic Estimation
apm_option(s,a,'nlc.imode',5);
% Read csv file
apm_option(s,a,'nlc.csv_read',1);
% Type (1=l1-norm, 2=Squared Error)
apm_option(s,a,'nlc.ev_type',2);
% Time units (1=sec, 2=min, etc)
apm_option(s,a,'nlc.ctrl_units',1);
apm_option(s,a,'nlc.hist_units',2);
% Parameters to adjust
apm_option(s,a,'km.status',1);
apm_option(s,a,'km.lower',3);
apm_option(s,a,'km.upper',20);
apm_option(s,a,'kb.status',1);
apm_option(s,a,'kb.lower',-2);
apm_option(s,a,'kb.upper',2);
apm_option(s,a,'gamma[1].status',1);
apm_option(s,a,'gamma[1].lower',0.2);
apm_option(s,a,'gamma[1].upper',0.8);
apm_option(s,a,'gamma[2].status',1);
apm_option(s,a,'gamma[2].lower',0.2);
apm_option(s,a,'gamma[2].upper',0.8);
apm_option(s,a,'c13.status',1);
apm_option(s,a,'c13.lower',0.01);
apm_option(s,a,'c13.upper',0.2);
apm_option(s,a,'c24.status',1);
apm_option(s,a,'c24.lower',0.01);
apm_option(s,a,'c24.upper',0.2);
% Measured values
apm_option(s,a,'h[1].fstatus',1);
apm_option(s,a,'h[2].fstatus',1);
% Solver (1=APOPT, 3=IPOPT)
apm_option(s,a,'nlc.solver',3);
% Solve with APMonitor
apm(s,a,'solve')
% Open web-viewer
apm_web(s,a);
% Retrieve solution
solution = apm_sol(s,a);
```

Python Dynamic Estimation

```
# Import APM Package for Python
from apm import *


# Server and Application name
s = 'http://xps.apmonitor.com'
a = 'prbs'
# Clear previous application
apm(s,a,'clear all')
# Load model and data
apm_load(s,a,'4tank.apm')
csv_load(s,a,'prbs360.csv')
# Set up variable classifications
# Feedforwards
apm_info(s,a,'FV','km')
apm_info(s,a,'FV','kb')
apm_info(s,a,'FV','gamma[1]')
apm_info(s,a,'FV','gamma[2]')
apm_info(s,a,'FV','c13')
apm_info(s,a,'FV','c24')
# State variables
apm_info(s,a,'SV','h[3]')
apm_info(s,a,'SV','h[4]')
# Controlled variables
apm_info(s,a,'CV','h[1]')
apm_info(s,a,'CV','h[2]')
# Dynamic Estimation
apm_option(s,a,'nlc.imode',5)
# Read csv file
apm_option(s,a,'nlc.csv_read',1)
# Type (1=l1-norm, 2=Squared Error)
apm_option(s,a,'nlc.ev_type',2)
# Time units (1=sec, 2=min, etc)
apm_option(s,a,'nlc.ctrl_units',1)
apm_option(s,a,'nlc.hist_units',2)
# Parameters to adjust
apm_option(s,a,'km.status',1)
apm_option(s,a,'km.lower',3)
apm_option(s,a,'km.upper',20)
apm_option(s,a,'kb.status',1)
apm_option(s,a,'kb.lower',-2)
apm_option(s,a,'kb.upper',2)
apm_option(s,a,'gamma[1].status',1)
apm_option(s,a,'gamma[1].lower',0.2)
apm_option(s,a,'gamma[1].upper',0.8)
apm_option(s,a,'gamma[2].status',1)
apm_option(s,a,'gamma[2].lower',0.2)
apm_option(s,a,'gamma[2].upper',0.8)
apm_option(s,a,'c13.status',1)
apm_option(s,a,'c13.lower',0.01)
apm_option(s,a,'c13.upper',0.2)
apm_option(s,a,'c24.status',1)
apm_option(s,a,'c24.lower',0.01)
apm_option(s,a,'c24.upper',0.2)
# Measured values
apm_option(s,a,'h[1].fstatus',1)
apm_option(s,a,'h[2].fstatus',1)
# Solver (1=APOPT, 3=IPOPT)
apm_option(s,a,'nlc.solver',3)
# Solve with APMonitor
apm(s,a,'solve')
# Open web-viewer
apm_web(s,a)
# Retrieve solution
(y,solution) = apm_sol(s,a)
```

Appendix D. Nonlinear control of the quadruple tank process

The MATLAB and Python scripts in Listing 4 detail the commands necessary to reproduce the nonlinear controller presented in this paper. The model file is the same as is shown in Appendix B but updated with new parameters from Table 4.

Listing 4. MATLAB nonlinear control.

Listing 4: MATLAB Nonlinear Control

```
addpath('apm');
% Clear MATLAB
clear all; close all; clc
% Server and Application Name
s = 'http://xps.apmonitor.com';
a = 'nlc';
% Clear previous application
apm(s,a,'clear all');
% Load model
apm_load(s,a,'4tank_nlc.apm');
% Load future time horizon
csv_load(s,a,'control.csv');
% Set up variable classifications
% Feedforwards
apm_info(s,a,'FV','gamma[1]');
apm_info(s,a,'FV','gamma[2]');
% Manipulated variables
apm_info(s,a,'MV','v1');
apm_info(s,a,'MV','v2');
% State variables
apm_info(s,a,'SV','h[3]');
apm_info(s,a,'SV','h[4]');
% Controlled variables
apm_info(s,a,'CV','h[1]');
apm_info(s,a,'CV','h[2]');
% Steady state initialization
apm_option(s,a,'nlc.imode',3);
apm(s,a,'solve');
% Dynamic control
apm_option(s,a,'nlc.imode',6);
% Internal nodes
apm_option(s,a,'nlc.nodes',3);
% Time units (1=sec, 2=min, etc)
apm_option(s,a,'nlc.ctrl_units',1);
apm_option(s,a,'nlc.hist_units',2);
% Read csv file
apm_option(s,a,'nlc.csv_read',1);
% Manipulated variable tuning
apm_option(s,a,'v1.status',1);
apm_option(s,a,'v1.upper',6);
apm_option(s,a,'v1.lower',1);
apm_option(s,a,'v1.dmax',1);
apm_option(s,a,'v1.dcost',1);
apm_option(s,a,'v2.status',1);
apm_option(s,a,'v2.upper',6);
apm_option(s,a,'v2.lower',1);
apm_option(s,a,'v2.dmax',1);
apm_option(s,a,'v2.dcost',1);
% Controlled variable tuning
apm_option(s,a,'h[1].status',1);
apm_option(s,a,'h[1].fstatus',0);
apm_option(s,a,'h[1].sphi',10.1);
apm_option(s,a,'h[1].splo',9.9);
apm_option(s,a,'h[1].tau',10);
apm_option(s,a,'h[2].status',1);
apm_option(s,a,'h[2].fstatus',0);
apm_option(s,a,'h[2].sphi',15.1);
apm_option(s,a,'h[2].splo',14.9);
apm_option(s,a,'h[2].tau',10);
% Set controller mode
apm_option(s,a,'nlc.reqctrlmode',3);
% Run APMonitor
apm(s,a,'solve')
% Open web-viewer
apm_web(s,a);
% Retrieve solution
solution = apm_sol(s,a);
```

Python Nonlinear Control

```
from apm import *


# Server and Application Name
s = 'http://xps.apmonitor.com'
a = 'nlc'
# Clear previous application
apm(s,a,'clear all')
# load model
apm_load(s,a,'4tank_nlc.apm')
# load future time horizon
csv_load(s,a,'control.csv')
# Set up variable classifications
# Feedforwards
apm_info(s,a,'FV','gamma[1]')
apm_info(s,a,'FV','gamma[2]')
# Manipulated variables
apm_info(s,a,'MV','v1')
apm_info(s,a,'MV','v2')
# State variables
apm_info(s,a,'SV','h[3]')
apm_info(s,a,'SV','h[4]')
# Controlled variables
apm_info(s,a,'CV','h[1]')
apm_info(s,a,'CV','h[2]')
# Steady state initialization
apm_option(s,a,'nlc.imode',3)
apm(s,a,'solve')
# Dynamic control
apm_option(s,a,'nlc.imode',6)
# Internal nodes in the collocation
apm_option(s,a,'nlc.nodes',3)
# Time units (1=sec, 2=min, etc)
apm_option(s,a,'nlc.ctrl_units',1)
apm_option(s,a,'nlc.hist_units',2)
# Read csv file
apm_option(s,a,'nlc.csv_read',1)
# Manipulated variable tuning
apm_option(s,a,'v1.status',1)
apm_option(s,a,'v1.upper',6)
apm_option(s,a,'v1.lower',1)
apm_option(s,a,'v1.dmax',1)
apm_option(s,a,'v1.dcost',1)
apm_option(s,a,'v2.status',1)
apm_option(s,a,'v2.upper',6)
apm_option(s,a,'v2.lower',1)
apm_option(s,a,'v2.dmax',1)
apm_option(s,a,'v2.dcost',1)
# Controlled variable tuning
apm_option(s,a,'h[1].status',1)
apm_option(s,a,'h[1].fstatus',0)
apm_option(s,a,'h[1].sphi',10.1)
apm_option(s,a,'h[1].splo',9.9)
apm_option(s,a,'h[1].tau',10)
apm_option(s,a,'h[2].status',1)
apm_option(s,a,'h[2].fstatus',0)
apm_option(s,a,'h[2].sphi',15.1)
apm_option(s,a,'h[2].splo',14.9)
apm_option(s,a,'h[2].tau',10)
# Set controller mode
apm_option(s,a,'nlc.reqctrlmode',3)
# Run APMonitor
apm(s,a,'solve')
# Open web-viewer
apm_web(s,a)
# Retrieve solution
(y,solution) = apm_sol(s,a)
```

References

Darby ML, Nikolaou M. MPC: Current practice and challenges. Control Eng Pract 2012];20(4):328–42.

Qin S, Badgwell T. A survey of industrial model predictive control technology. Control Eng Pract 2003];11:733–64.

Gyalistras D, Pr S, Sagerschnig C, Morari M. Modeling and identification of a large multi-zone office building. In: IEEE international conference on control applications (CCA), 17; 2011]. p. 55–60.

Oldewurtel F, Parisio A, Jones C, Morari M, Gyalistras D, Gwerder M, et al. Energy efficient building climate control using stochastic model predictive control and weather predictions. In: American control conference (ACC); 2010]. p. 5100–5.

Nolde K, Uhr M, Morari M. Medium term scheduling of a hydro-thermal system using stochastic model predictive control. Automatica 2008];44:1585–94.

Oldewurtel F, Jones CN, Morari M. A tractable approximation of chance constrained stochastic MPC based on affine disturbance feedback. In: 2008 47th IEEE conference on decision and control; 2008]. p. 4731–6.

Papafotiou G, Geyer T, Morari M. A hybrid model predictive control approach to the direct torque control problem of induction motors. Int J Robust Nonlinear Control 2007];17:1572–89.

Bemporad A, Morari M, Dua V, Pistokopoulos E. The explicit linear quadratic regulator for constrained systems. Automatica 2002];38:3–20.

Hedengren J, Edgar T. Approximate nonlinear model predictive control with in situ adaptive tabulation. Comput Chem Eng 2008];32:706–14.

Johansen T. Approximate explicit receding horizon control of constrained nonlinear systems. Automatica 2004];40:293–300.

Domahidi A, Zeilinger MN, Morari M, Jones CN. Learning a feasible and stabilizing explicit model predictive control law by robust optimization. In: IEEE Conference on Decision and Control and European Control Conference; 2011]. p. 513–9.

Ferreau HJ, Bock HG, Diehl M. An online active set strategy to overcome the limitations of explicit MPC. Int J Robust Nonlinear Control 2008];18:816–30.

Pannocchia G, Rawlings JB, Wright SJ. Fast, large-scale model predictive control by partial enumeration. Automatica 2007];43(5):852–60.

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

Hedengren J. APMonitor modeling language; 2014] http://APMonitor.com

Hedengren J. APMonitor modeling language for mixed-integer differential algebraic systems. In: Computing society session on optimization modeling software: design and applications; 2012].

Cizniar M, Salhi D, Fikar M, Latifi M. A MATLAB package for orthogonal collocations on finite elements in dynamic optimisation. In: Proceedings of the 15th international conference process control '05; 2005].

Houska B, Ferreau HJ, Diehl M. ACADO toolkit: an open-source framework for automatic control and dynamic optimization. Optimal Control Appl Methods 2011];32:298–312.

Piela P, Westerberg A, Westerberg K, Epperly T. ASCEND: an object-oriented computer environment for modeling and analysis: the modeling language. Comput Chem Eng 1991];15:53–72.

Tummescheit H, Gäfvert M, Bergdahl T, Årzén K-E, AAkesson J. Modeling and optimization with Optimica and JModelica.org Languages and tools for solving large-scale dynamic optimization problems. Comput Chem Eng 2010];34:1737–49.

Simon LL, Nagy ZK, Hungerbuehler K. Swelling constrained control of an industrial batch reactor using a dedicated NMPC environment: OptCon. In: Magni L, Raimond DM, Allgower F, editors. Nonlinear model predictive control. Lecture notes in control and information sciences, vol.384. Berlin/Heidelberg: Springer-Verlag; 2009]. p. 531–9.

Nagy Z, Mahn B, Franke R, Allgöwer F. Evaluation study of an efficient output feedback nonlinear model predictive control for temperature tracking in an industrial batch reactor. Control Eng Pract 2007];15(7):839–50.

Hedengren J, Mojica J, Cole W, Edgar T. APOPT: MINLP solver for differential and algebraic systems with benchmark testing. In: INFORMS national meeting; 2012].

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

Abbott CS, Haseltine EL, Martin RA, Hedengren JD. New capabilities for large-scale models in computational biology. In: AIChE national meeting; 2012].

Sun L, Hedengren J, Beard R. Optimal trajectory generation using model predictive control for aerially towed cable systems. J Guid Control Dyn 2014];37(2):525–39.

Soderstrom T, Zhang Y, Hedengren J. Advanced process control in Exxonmobil Chemical Company: Successes and challenges. In: AIChE national meeting; 2010].

Jacobsen L, Spivey B, Hedengren J. Model predictive control with a rigorous model of a solid oxide fuel cell. In: Proceedings of the American control conference (ACC); 2013]. p. 3747–52.

Spivey B, Hedengren J, Edgar T. Constrained control and optimization of tubular solid oxide fuel cells for extending cell lifetime. In: Proceedings of the American control conference (ACC); 2012]. p. 1356–61.

Spivey B, Hedengren J, Edgar T. Constrained nonlinear estimation for industrial process fouling. Ind Eng Chem Res 2010];49(17):7824–31.

Jensen K, Hedengren J. Improved load following of a boiler with advanced process control. In: AIChE Spring Meeting; 2012].

Powell K, Hedengren J, Edgar T. Dynamic optimization of a solar thermal energy storage system over a 24 hour period using weather forecasts. In: Proceedings of the American Control Conference (ACC); 2013. p. 2952–7.

Powell KM, Edgar TF. Control of a large scale solar thermal energy storage system. In: Proceedings of the 2011 American control conference; 2011]. p. 1530–5.

Powell KM, Edgar TF. Modeling and control of a solar thermal power plant with thermal energy storage. Chem Eng Sci 2012];71:138–45.

Hedengren J, Brower D. Advanced process monitoring of flow assurance with fiber optics. In: AIChE spring meeting; 2012].

Brower D, Hedengren J, Loegering C, Brower A, Witherow K, Winter K. Fiber optic monitoring of subsea equipment. In: Ocean, offshore & arctic engineering OMAE, no. 84143; 2012].

Brower D, Hedengren J, Shishivan RA, Brower A. Advanced deepwater monitoring system. In: Ocean, offshore & arctic engineering OMAE, no. 10920; 2013].

Nielsen I. Modeling and control of friction stir welding in 5 cm thick copper canisters. Linköping University; 2012], Master's thesis.

Genceli H, Nikolaou M. Robust stability analysis of constrained ℓ_1-norm model predictive control. AIChE J 1993];39:1954–65.

Garcia CE, Prett DM, Morari M. Model predictive control: theory and practice a survey. Automatica 1989];25:335–48.

Nikolaou M. Model predictive controllers: a critical synthesis of theory and industrial needs. In: Vol. 26 of Advances in Chemical Engineering. Academic Press; 2001]. p. 131–204.

Biegler LT. An overview of simultaneous strategies for dynamic optimization. Chem Eng Process 2007];46(11):1043–53.

Hedengren J. Pendulum motion in the APMonitor modeling language; 2014] http://APMonitor.com/wiki/index.php/Apps/PendulumMotion

Barth TJ, Keyes DE, Roose D. Advances in automatic differentiation. Lecture Notes in Computational Science and Engineering, 64. Berlin: Springer; 2008].

Abul-el-zeet Z, Roberts P. Enhancing model predictive control using dynamic data reconciliation. AIChE J 2002];48(2):324–33.

Liebman M, Edgar T, Lasdon L. Efficient data reconciliation and estimation for dynamic processes using nonlinear programming techniques. Comput Chem Eng 1992];16:963–86.

McBrayer K, Edgar T. Bias detection and estimation in dynamic data reconciliation. J Process Control 1995];5(4):285–9.

Soderstrom T, Edgar T, Russo L, Young R. Industrial application of a large-scale dynamic data reconciliation strategy. Ind Eng Chem Res 2000];39:1683–93.

Ramamurthi Y, Sistu P, Bequette B. Control-relevant dynamic data reconciliation and parameter estimation. Comput Chem Eng 1993];17(1):41–59.

Albuquerque JS, Biegler LT. Data reconciliation and gross-error detection for dynamic systems. AIChE J 1996];42:2841–56.

Arora N, Biegler LT. A trust region SQP algorithm for equality constrained parameter estimation with simple parameter bounds. Comput Opt Appl 2004];28:51–86.

Biegler LT, Arora N. Redescending estimators for data reconciliation and parameter estimation. Comput Chem Eng 2001];25:1585–99.

Gatzke EP, Doyle FJ III. Use of multiple models and qualitative knowledge for on-line moving horizon disturbance estimation and fault diagnosis. J Process Control 2002];12(2):339–52.

Mahadevan R, Doyle F III. A partial flatness approach to nonlinear moving horizon estimation. In: Proceedings of the 2004 American Control Conference; 2004]. p. 211–5.

Voelker A, Kouramas K, Pistikopoulos EN. Moving horizon estimation: error dynamics and bounding error sets for robust control. Automatica 2013];49:943–8.

Landau I, Lozano R, M'Saad M, Karimi A. Adaptive control: algorithms, analysis and applications. Communications and control engineering, 2nd Edition London: Springer-Verlag; 2011].

Ramlal J, Naidoo V, Allsford K, Hedengren J. Moving horizon estimation for an industrial gas phase polymerization reactor. In: Proc. IFAC symposium on nonlinear control systems design (NOLCOS); 2007].

Binder T, Blank L, Bock H, Burlisch R, Dahmen W, Diehl M, et al. Introduction to model based optimization of chemical processes on moving horizons. In: Grotschel M, Krumke SO, Rambau J, editors. Online optimization of large scale systems. Berlin/Heidelberg: Springer-Verlag; 2001]. p. 295–339.

Shaohua W, Kevin A, Harris T, McAuley K. Selection of optimal parameter set using estimability analysis and MSE-based model-selection criterion. Int J Adv Mechatr Syst 2011];3(3):188–97.

Hedengren J, Allsford K, Ramlal J. Moving horizon estimation and control for an industrial gas phase polymerization reactor. In: Proceedings of the American control conference (ACC); 2007]. p. 1353–8.

Biegler L, Campbell S, Mehrmann V. Control and optimization with differential-algebraic constraints. Philadelphia: SIAM – Society for Industrial and Applied Mathematics; 2012].

Albuquerque J, Biegler L. Decomposition algorithms for on-line estimation with nonlinear DAE models. Comput Chem Eng 1997];21(3):283–99.

Diehl M, Bock HG, Schlöder JP, Findeisen R, Nagy Z, Allgöwer F. Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. J Process Control 2002];12:577–85.

Haseltine E, Rawlings J. Critical evaluation of extended kalman filtering and moving-horizon estimation. Ind Eng Chem Res 2005];44(8):2451–60.

Odelson B, Rajamani M, Rawlings J. A new autocovariance least-squares method for estimating noise covariances. Automatica 2006];42(2):303–8.

Hedengren J, Edgar T. Moving horizon estimation: the explicit solution. In: Proceedings of chemical process control (CPC) VII conference; 2006].

Spivey B, Hedengren J, Edgar T. Monitoring of process fouling using first-principles modeling and moving horizon estimation. In: Proceedings of Texas, Wisconsin, California Control Consortium (TWCCC); 2009].

Darby M, Nikolaou M, Jones J, Nicholson D. RTO: an overview and assessment of current practice. J Process Control 2011];21:874–84.

Carey G, Finlayson B. Othogonal collocation on finite elements. Chem Eng Sci 1975];30:587–96.

Albuquerque JS, Biegler LT. Decomposition algorithms for on-line estimation with nonlinear models. Comput Chem Eng 1995];19:1031–9.

Johansson K. Interaction bounds in multivariable control systems. Automatica 2002];38(6):1045–51.

Raff T, Huber S, Nagy ZK, Allgöwer F. Nonlinear model predictive control of a four tank system: An experimental stability study. In: Proceedings of international conference control applications; 2006]. p. 237–42.

Gatzke E, Meadows E, Wang C, Doyle F III. Model based control of a four-tank system. Comput Chem Eng 2000];24:1503–9.

Johansson K. The quadruple-tank process – a multivariable laboratory process with an adjustable zero. IEEE Trans Control Syst Technol 2000];8(3):456–65.

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

Mercangöz M, Doyle F III. Distributed model predictive control of an experimental four-tank system. J Process Control 2007];17(3):297–308.

Alvarado I, Limon D, noz de la Pe na DM, Maestre J, Ridao M, Scheu H, et al. A comparative analysis of distributed MPC techniques applied to the HD-MPC four-tank benchmark. J Process Control 2011];21(5):800–15.

Biegler L. Nonlinear programming: concepts, algorithms, and applications to chemical processes. Society for Industrial and Applied Mathematics and the Mathematical Optimization Society; 2010].

Rao C, Rawlings J. Linear programming and model predictive control. J Process Control 2000];10:283–9.

Dave P, Willig D, Kudva G, Pekny J, Doyle F III. LP methods in MPC of large-scale systems: application to paper-machine CD control. AIChE J 1997];43:1016–31.

Dave P, Doyle F III, Pekny J. Customization strategies for the solution of linear programming problems arising from large scale model predictive control of a paper machine. J Process Control 1999];9:385–96.

Hedengren J. MATLAB toolbox for the APMonitor modeling language; 2014a] http://APMonitor.com/wiki/index.php/Main/MATLAB

Hedengren J. Python toolbox for the APMonitor modeling language; 2014b] http://APMonitor.com/wiki/index.php/Main/PythonApp