JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 21, No. 4, July–August 1998

Recursive Deadbeat Controller Design

Jer-Nan Juang*
NASA Langley Research Center, Hampton, Virginia 23681
and
Minh Q. Phan†
Princeton University, Princeton, New Jersey 08544

A recursive algorithm is presented for the deadbeat predictive control that brings the output response to rest after a few time steps. The main idea is to put together the system identification process and the deadbeat control design into a unified step. It starts with reformulating the conventional multistep output-prediction equation to explicitly include the coefficient matrices to weight past input and output time histories for computation of feedback control force. The formulation thus derived satisfies simultaneously the system identification and the deadbeat control requirements. As soon as the coefficient matrices are identified satisfying the output prediction equation, no further work is required to design a deadbeat controller. The method can be implemented recursively just as any typical recursive system identification technique.

Introduction

THERE are many interesting technical problems in the area of controlled aerospace structures that NASA researchers are trying to solve. These problems, for example, include acoustic noise reduction, flow control, ride quality control, flexible spacecraft attitude control, and vibration suppression. Active or passive control for a dynamic system is not a new subject. Many control techniques are available today and ready to be used for application to these interesting problems. Some of the techniques are the quadratic optimization technique,[1,2] the pole placement technique,[3] the virtual passive technique,[4] the energy dissipation technique,[5] and the adaptive control technique.[6–8] Some researchers prefer to work in the frequency domain using the frequency response functions (FRF) whereas others use the state-space model (SSM) in the time domain to design controllers. The model-based techniques need a mathematical model (FRF or SSM) within a certain level of accuracy to design a controller. Except for a few simple cases, system identification[9–12] must be involved in the design process to verify the open-loop model and the closed-loop design as well. As a result, it may take considerable time to iterate the design process until performance requirements are met. For the systems with minimum uncertainties, the iteration procedure would not bother the control engineers, as long as a satisfactory control design can be found.

For systems with unknown disturbances and considerable uncertainties, the controller must be able to adapt the unknown changes in real time. Adaptive control techniques are developed for this purpose. The approach is to adjust the control gains to reflect the system changes so as to continuously check and meet the performance requirements. Most adaptive control techniques require the controlled system to be minimum phase[13–19] in the sense that all of the system transmission zeros are stable. The minimum-phase system in the continuous-time domain does not guarantee its minimum phase in the discrete-time domain. In practice, only a few structural systems in the discrete-time domain are minimum phase.

Recently, an innovative neural network method was developed[20] for online system identification and adaptively optimized control. For practical applications, this method was modified and enhanced[21] by using a multiprocessor architecture, which may have a variety of configurations but is particularly suited for a neural network. The neural network may be built up of neurons that are either purely one way (forward signal path) or two way. Each neuron may be provided with its own synaptic weight, adjusted using only the local and backward signals.

To understand better the neural networks in the identification and control of dynamic systems, a study[22] has been conducted focusing on how the neural networks handle linear systems and how it is related to conventional system identification and control methods. The study explained the fundamental concepts of neural networks in their simplest terms. Among the topics discussed were feedforward and recurrent networks in relation to the standard state-space and observer models, linear and nonlinear autoregressive models, linear predictors, one-step ahead control, and model reference adaptive control for linear and nonlinear systems. It is concluded that the output prediction determined from input and output data is a key to the success of an adaptive controller that leads to the study of predictive controllers.

Predictive controller designs[23–32] were developed to particularly address the nonminimum-phase problems with the hope that they can be implemented in real time. Two indirect techniques and one direct technique were derived[32] using the concept of deadbeat predictive control law that brought the output response to rest after a few finite time steps. The indirect techniques require identification of coefficient matrices of a finite difference model representing the controlled system. The deadbeat predictive controllers are then computed using the identified coefficient matrices. Note that the identified matrices minimize the output error between the estimated and real outputs. The direct technique computes the deadbeat predictive controller directly from input and output data without explicitly identifying the system parameters. However, it requires minimization of the output error first and then performance of the inversion of a Hankel-like[32] matrix to calculate the control gains for the past input and past output signal. Because it takes time to invert a matrix, both direct and indirect algorithms have a drawback for real-time application. Nevertheless, the direct algorithm did provide the fundamental framework for further development of a recursive technique for real-time implementation.

A new recursive technique is presented in this paper for the design of a deadbeat predictive controller. It uses the approach derived for the direct algorithm.[32] The technique computes the gain matrices recursively and directly from input and output data in every sampling period. In addition, the recursive formula satisfies both system identification and deadbeat predictive control equations simultaneously. As a result, the design process is completed in such a way that there is no time delay between the identification step and the control gain computation step.

This paper begins with a brief introduction of multistep output prediction.[32] The basic formulation for a deadbeat controller design

Received Dec. 18, 1996; revision received Feb. 12, 1998; accepted for publication Feb. 19, 1998.

*Principal Scientist, Structural Dynamics Branch. Fellow AIAA.

†Assistant Professor, Department of Aerospace and Mechanical Engineering.

is then derived giving the mathematical foundation of the recursive method. A recursive formula with computational steps is also included for real-time implementation. With a slight modification, the formula is extended to compute the feedforward gain for a measurable or predictable disturbance input. Finally, several numerical examples are given for illustration of the method.

Multistep Output Prediction

For the multistep output prediction,[32] the input/output relationship of a linear system is commonly described by a finite difference model. Given a system with r inputs and m outputs, the finite difference equation for the $r \times 1$ input $\boldsymbol{u}(k)$ and the $m \times 1$ output $\boldsymbol{y}(k)$ at time k is

$$\begin{aligned} \boldsymbol{y}(k) &= \alpha_1 \boldsymbol{y}(k-1) + \alpha_2 \boldsymbol{y}(k-2) + \cdots + \alpha_p \boldsymbol{y}(k-p) \\ &\quad + \beta_0 \boldsymbol{u}(k) + \beta_1 \boldsymbol{u}(k-1) + \beta_2 \boldsymbol{u}(k-2) + \cdots + \beta_p \boldsymbol{u}(k-p) \end{aligned} \tag{1}$$

It simply means that the current output can be predicted by the past input and output time histories. The finite difference model is also often referred to as the ARX model where AR refers to the autoregressive part and X refers to the exogeneous part. The coefficient matrices, α_i $(i = 1, 2, \ldots, p)$ of $m \times m$ and β_i $(i = 1, 2, \ldots, p)$ of $m \times r$ are commonly referred to as the observer Markov parameters (OMP) or ARX parameters. The matrix β_0 is the direct transmission term.

By shifting a time step, one obtains

$$\begin{aligned} \boldsymbol{y}(k+1) &= \alpha_1 \boldsymbol{y}(k) + \alpha_2 \boldsymbol{y}(k-1) + \cdots + \alpha_p \boldsymbol{y}(k-p+1) \\ &\quad + \beta_0 \boldsymbol{u}(k+1) + \beta_1 \boldsymbol{u}(k) + \beta_2 \boldsymbol{u}(k-1) + \cdots + \beta_p \boldsymbol{u}(k-p+1) \end{aligned} \tag{2}$$

Define the following quantities:

$$\begin{aligned} \alpha_1^{(1)} &= \alpha_1\alpha_1 + \alpha_2 \\ \alpha_2^{(1)} &= \alpha_1\alpha_2 + \alpha_3 \\ &\vdots \\ \alpha_{p-1}^{(1)} &= \alpha_1\alpha_{p-1} + \alpha \\ \alpha_p^{(1)} &= \alpha_1\alpha_p \end{aligned} \tag{3}$$

and

$$\begin{aligned} \beta_1^{(1)} &= \alpha_1\beta_1 + \beta_2 \\ \beta_2^{(1)} &= \alpha_1\beta_2 + \beta_3 \\ &\vdots \\ \beta_{p-1}^{(1)} &= \alpha_1\beta_{p-1} + \beta_p \\ \beta_p^{(1)} &= \alpha_1\beta_p \end{aligned} \tag{4}$$

and

$$\beta_0^{(1)} = \alpha_1\beta_0 + \beta_1 \tag{5}$$

Substituting $\boldsymbol{y}(k)$ from Eq. (1) into Eq. (2) yields

$$\begin{aligned} \boldsymbol{y}(k+1) &= \alpha_1^{(1)} \boldsymbol{y}(k-1) + \alpha_2^{(1)} \boldsymbol{y}(k-2) + \cdots + \alpha_p^{(1)} \boldsymbol{y}(k-p) \\ &\quad + \beta_0 \boldsymbol{u}(k+1) + \beta_0^{(1)} \boldsymbol{u}(k) + \beta_1^{(1)} \boldsymbol{u}(k-1) \\ &\quad + \beta_2^{(1)} \boldsymbol{u}(k-2) + \cdots + \beta_p^{(1)} \boldsymbol{u}(k-p) \end{aligned} \tag{6}$$

The output measurement at time step $k+1$ can be expressed as the sum of past input and output data with the absence of the output measurement at time step k. By induction, one may express the output measurement at the time step $k+j$ by

$$\begin{aligned} \boldsymbol{y}(k+j) &= \alpha_1^{(j)} \boldsymbol{y}(k-1) + \alpha_2^{(j)} \boldsymbol{y}(k-2) + \cdots + \alpha_p^{(j)} \boldsymbol{y}(k-p) \\ &\quad + \beta_0 \boldsymbol{u}(k+j) + \beta_0^{(1)} \boldsymbol{u}(k+j-1) + \cdots + \beta_0^{(j)} \boldsymbol{u}(k) \\ &\quad + \beta_1^{(j)} \boldsymbol{u}(k-1) + \beta_2^{(j)} \boldsymbol{u}(k-2) + \cdots + \beta_p^{(j)} \boldsymbol{u}(k-p) \end{aligned} \tag{7}$$

where

$$\begin{aligned} \alpha_1^{(j)} &= \alpha_1^{(j-1)}\alpha_1 + \alpha_2^{(j-1)} \\ \alpha_2^{(j)} &= \alpha_1^{(j-1)}\alpha_2 + \alpha_3^{(j-1)} \\ &\vdots \\ \alpha_{p-1}^{(j)} &= \alpha_1^{(j-1)}\alpha_{p-1} + \alpha_p^{(j-1)} \\ \alpha_p^{(j)} &= \alpha_1^{(j-1)}\alpha_p \end{aligned} \tag{8}$$

and

$$\begin{aligned} \beta_1^{(j)} &= \alpha_1^{(j-1)}\beta_1 + \beta_2^{(j-1)} \\ \beta_2^{(j)} &= \alpha_1^{(j-1)}\beta_2 + \beta_3^{(j-1)} \\ &\vdots \\ \beta_{p-1}^{(j)} &= \alpha_1^{(j-1)}\beta_{p-1} + \beta_p^{(j-1)} \\ \beta_p^{(j)} &= \alpha_1^{(j-1)}\beta_p \end{aligned} \tag{9}$$

and

$$\beta_0^{(j)} = \alpha_1^{(j-1)}\beta_0 + \beta_1^{(j-1)} \tag{10}$$

Note that $\alpha_i^{(0)} = \alpha_i$ and $\beta_i^{(0)} = \beta_i$ for any possible integer $1, 2, \ldots,$ including 0 if applicable. With some algebraic operation, Eq. (10) can also be expressed by

$$\begin{aligned} \beta_0^{(0)} &= \beta_0 \\ \beta_0^{(k)} &= \beta_k + \sum_{i=1}^{k} \alpha_i \beta_0^{(k-i)} \qquad \text{for} \quad k = 1, \ldots, p \\ \beta_0^{(k)} &= \sum_{i=1}^{p} \alpha_i \beta_0^{(k-i)} \qquad \text{for} \quad k = p+1, \ldots, \infty \end{aligned} \tag{11}$$

Similar to Eq. (11), $\alpha_1^{(j)} = \alpha_1^{(j-1)}\alpha_1 + \alpha_2^{(j-1)}$ can also be written as

$$\begin{aligned} \alpha_1^{(0)} &= \alpha_1 \\ \alpha_1^{(k)} &= \alpha_{k+1} + \sum_{i=1}^{k} \alpha_i \alpha_1^{(k-i)} \qquad \text{for} \quad k = 1, \ldots, p-1 \\ \alpha_1^{(k)} &= \sum_{i=1}^{p} \alpha_i \alpha_1^{(k-i)} \qquad \text{for} \quad k = p, \ldots, \infty \end{aligned} \tag{12}$$

Observation of Eqs. (11) and (12) reveals that $\beta_0^{(j)}$ and $\alpha_1^{(j)}$ for $j > p$ is a linear combination of its past p parameters weighted by the parameters $\alpha_1, \alpha_2, \ldots, \alpha_p$. This property is very useful in developing predictive control designs. The quantities $\beta_0^{(i)}$ $(i = 0, 1, \ldots)$ are, in fact, the pulse response sequence.[32] On the other hand, the quantities $\alpha_1^{(i)}$ $(i = 0, 1, \ldots)$ are the observer gain Markov parameters, which can be used to compute an observer for state estimation.[32]

Let the index j be $j = 1, 2, \ldots, q$. Equation (7) produces the following multistep output prediction:

$$\boldsymbol{y}_p(k+q) = \mathcal{T}' \boldsymbol{u}_{q+p}(k) + \mathcal{B}' \boldsymbol{u}_p(k-p) + \mathcal{A}' \boldsymbol{y}_p(k-p) \tag{13}$$

where

$$\boldsymbol{y}_p(k+q) = \begin{bmatrix} \boldsymbol{y}(k+q) \\ \boldsymbol{y}(k+q+1) \\ \vdots \\ \boldsymbol{y}(k+q+p-1) \end{bmatrix} \qquad \boldsymbol{u}_{q+p}(k) = \begin{bmatrix} \boldsymbol{u}(k) \\ \boldsymbol{u}(k+1) \\ \vdots \\ \boldsymbol{u}(k+q+p-1) \end{bmatrix} \qquad \boldsymbol{y}_p(k-p) = \begin{bmatrix} \boldsymbol{y}(k-p) \\ \boldsymbol{y}(k-p+1) \\ \vdots \\ \boldsymbol{y}(k-1) \end{bmatrix} \tag{14}$$

and $\boldsymbol{u}_p(k-p)$ is identical to $\boldsymbol{y}_p(k-p)$ with $\boldsymbol{y}$ replaced by $\boldsymbol{u}$:

$$\mathcal{T}' = \begin{bmatrix} \beta_0^{(q)} & \cdots & \beta_0^{(1)} & \beta_0 & \cdots & 0 \\ \beta_0^{(q+1)} & \cdots & \beta_0^{(2)} & \beta_0^{(1)} & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \beta_0^{(q+p-1)} & \cdots & \beta_0^{(p)} & \beta_0^{(p-1)} & \cdots & \beta_0 \end{bmatrix}$$

$$\mathcal{B}' = \begin{bmatrix} \beta_p^{(q)} & \beta_{p-1}^{(q)} & \cdots & \beta_1^{(q)} \\ \beta_p^{(q+1)} & \beta_{p-1}^{(q+1)} & \cdots & \beta_1^{(q+1)} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_p^{(q+p-1)} & \beta_{p-1}^{(q+p-1)} & \cdots & \beta_1^{(q+p-1)} \end{bmatrix} \tag{15}$$

$$\mathcal{A}' = \begin{bmatrix} \alpha_p^{(q)} & \alpha_{p-1}^{(q)} & \cdots & \alpha_1^{(q)} \\ \alpha_p^{(q+1)} & \alpha_{p-1}^{(q+1)} & \cdots & \alpha_1^{(q+1)} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_p^{(q+p-1)} & \alpha_{p-1}^{(q+p-1)} & \cdots & \alpha_1^{(q+p-1)} \end{bmatrix}$$

The superscript on β and α signifies the time shift. The quantity $\boldsymbol{y}_p(k+q)$ represents the output vector with a total of p data points for each sensor from time step $k+q$ to $k+q+p-1$, whereas $\boldsymbol{y}_p(k-p)$ includes the p data from $k-p$ to $k-1$. Similarly, $\boldsymbol{u}_{q+p}(k)$ has $q+p$ input data points starting from the time step k, and $\boldsymbol{u}_p(k-p)$ has p input data points from $k-p$. The matrix $\mathcal{T}'$ is formed from the parameters, $\beta_0, \beta_0^{(1)}, \ldots,$ and $\beta_0^{(q+p-1)}$ (the pulse response sequence).

The vector $\boldsymbol{y}_p(k+q)$ in Eq. (13) consists of three terms. The first term is the input vector $\boldsymbol{u}_{q+p}(k)$ including inputs from the current time step k to the future time step $k+q+p-1$. Relative to the same time k, the second and third terms, $\boldsymbol{u}_p(k-p)$ and $\boldsymbol{y}_p(k-p)$, are input and output vectors from the past time step $k-p$ to $k-1$, respectively. The future input vector $\boldsymbol{u}_{q+p}(k)$ is to be determined for feedback control. The matrices $\mathcal{B}'$ and $\mathcal{A}'$ may be computed from OMPs α_i $(i = 1, 2, \ldots, p)$ and β_i $(i = 1, 2, \ldots, p)$, or directly from input and output data.

Deadbeat Predictive Control Designs

Several deadbeat control algorithms have been developed[32] using the multistep output prediction, Eq. (13). Among these algorithms, the direct algorithm (see the Appendix) uses the input and output data directly without using α_i $(i = 1, 2, \ldots, p)$ and β_i $(i = 1, 2, \ldots, p)$ to first compute $\mathcal{B}'$ and $\mathcal{A}'$ and then design a deadbeat predictive controller. The goal was to make the direct algorithm suitable for real-time implementation in the sense that the deadbeat controller may be updated at every sampling interval. Unfortunately, it involves a matrix inverse that is difficult, if not impossible, to compute it recursively. To overcome the computational difficulty, an alternative algorithm is developed in this section.

Let $\mathcal{T}'$ be partitioned into two parts such that Eq. (13) becomes

$$\boldsymbol{y}_p(k+q) = \mathcal{T}_o \boldsymbol{u}_p(k+q) + \mathcal{T}_c \boldsymbol{u}_q(k) + \mathcal{B}'\boldsymbol{u}_p(k-p) + \mathcal{A}'\boldsymbol{y}_p(k-p) \tag{16}$$

where

$$\boldsymbol{u}_p(k+q) = \begin{bmatrix} \boldsymbol{u}(k+q) \\ \boldsymbol{u}(k+q+1) \\ \vdots \\ \boldsymbol{u}(k+q+p-1) \end{bmatrix}$$

$$\boldsymbol{u}_q(k) = \begin{bmatrix} \boldsymbol{u}(k) \\ \boldsymbol{u}(k+1) \\ \vdots \\ \boldsymbol{u}(k+q-1) \end{bmatrix} \tag{17}$$

and

$$\mathcal{T}_o = \begin{bmatrix} \beta_0 & 0 & \cdots & 0 \\ \beta_0^{(1)} & \beta_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \beta_0^{(p-1)} & \beta_0^{(p-2)} & \cdots & \beta_0 \end{bmatrix}$$

$$\mathcal{T}_c = \begin{bmatrix} \beta_0^{(q)} & \beta_0^{(q-1)} & \cdots & \beta_0^{(1)} \\ \beta_0^{(q+1)} & \beta_0^{(q)} & \cdots & \beta_0^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_0^{(q+p-1)} & \beta_0^{(q+p-2)} & \cdots & \beta_0^{(p)} \end{bmatrix} \tag{18}$$

Both $\mathcal{T}_o$ of $pm \times pr$ and $\mathcal{T}_c$ of $pm \times qr$ are formed from system pulse response (system Markov parameters). Note that m is the number of outputs, p is the order of the ARX model, r is the number of inputs, and q is an integer. Given any input and output sequence $\boldsymbol{u}(k)$ and $\boldsymbol{y}(k)$, Eq. (16) must be satisfied and can be used for identifying coefficient matrices $\mathcal{T}_o$, $\mathcal{T}_c$, $\mathcal{A}'$, and $\mathcal{B}'$.

Note that the matrix $\mathcal{T}_c$ of $pm \times qr$ is a Hankel-like matrix[32] that has rank n where n is the order of the system. For the case where $qr \geq pm \geq n$, there are qr elements in $\boldsymbol{u}_q(k)$ of $qr \times 1$ with only n independent equations in Eq. (16). The first step in deriving a feedback and/or a feedforward controller is to write the equation for control input $\boldsymbol{u}_q(k)$ that forces the system to follow a given output $\boldsymbol{y}$ history in time. As a result, Eq. (16) provides multiple solutions for $\boldsymbol{u}_q(k)$ with the minimum-norm solution expressed by

$$\begin{aligned} \boldsymbol{u}_q(k) &= \mathcal{T}_c^{\dagger}\boldsymbol{y}_p(k+q) - \mathcal{T}_c^{\dagger}\mathcal{T}_o\boldsymbol{u}_p(k+q) \\ &\quad - \mathcal{T}_c^{\dagger}\mathcal{B}'\boldsymbol{u}_p(k-p) - \mathcal{T}_c^{\dagger}\mathcal{A}'\boldsymbol{y}_p(k-p) \end{aligned} \tag{19}$$

or in matrix form

$$\boldsymbol{u}_q(k) = \begin{bmatrix} -\mathcal{T}_c^{\dagger}\mathcal{A}' & -\mathcal{T}_c^{\dagger}\mathcal{B}' & \mathcal{T}_c^{\dagger} & -\mathcal{T}_c^{\dagger}\mathcal{T}_o \end{bmatrix} \begin{bmatrix} \boldsymbol{y}_p(k-p) \\ \boldsymbol{u}_p(k-p) \\ \boldsymbol{y}_p(k+q) \\ \boldsymbol{u}_p(k+q) \end{bmatrix} \tag{20}$$

where $\dagger$ is pseudoinverse. For the case where $qr = pm$, Eq. (20) is unique. To simplify Eq. (20), define the following notations:

$$F_c = \begin{bmatrix} -\mathcal{T}_c^{\dagger}\mathcal{A}' & -\mathcal{T}_c^{\dagger}\mathcal{B}' \end{bmatrix}, \qquad F_o = \begin{bmatrix} \mathcal{T}_c^{\dagger} & -\mathcal{T}_c^{\dagger}\mathcal{T}_o \end{bmatrix} \tag{21}$$

and

$$\boldsymbol{v}_p(k-p) = \begin{bmatrix} \boldsymbol{y}_p(k-p) \\ \boldsymbol{u}_p(k-p) \end{bmatrix}, \qquad \boldsymbol{v}_p(k+q) = \begin{bmatrix} \boldsymbol{y}_p(k+q) \\ \boldsymbol{u}_p(k+q) \end{bmatrix} \tag{22}$$

where both F_c and F_o are $qr \times p(m+r)$ matrices and both $\boldsymbol{v}_p(k-p)$ and $\boldsymbol{v}_p(k+q)$ are $(pm+pr) \times 1$ column vectors. Thus, Eq. (20) becomes

$$\boldsymbol{u}_q(k) = [F_c \quad F_o] \begin{bmatrix} \boldsymbol{v}_p(k-p) \\ \boldsymbol{v}_p(k+q) \end{bmatrix} \tag{23}$$

Equation (23) is another form of the finite difference model for system identification. For any given input and output data, there exists a set of F_c and F_o satisfying Eq. (23). Using Eq. (23) to develop a deadbeat controller is shown in the following.

Let us assume that the input vector $\boldsymbol{u}_q(k)$ is chosen such that

$$\boldsymbol{u}_q(k) = F_c \boldsymbol{v}_p(k-p) \tag{24}$$

To satisfy Eq. (23), the following equation must hold:

$$F_o \boldsymbol{v}_p(k+q) = \begin{bmatrix} \mathcal{T}_c^{\dagger} & -\mathcal{T}_c^{\dagger}\mathcal{T}_o \end{bmatrix} \begin{bmatrix} \boldsymbol{y}_p(k+q) \\ \boldsymbol{u}_p(k+q) \end{bmatrix} = 0 \tag{25}$$

If $\mathcal{T}_c^{\dagger}$ of $qr \times pm$ is full rank pm with $qr \geq pm$, and if $\boldsymbol{u}_p(k+q)$ is set to zero, then $\boldsymbol{y}_p(k+q)$ becomes zero. As a result, the control action $\boldsymbol{u}_q(k)$ computed from Eq. (24) is a deadbeat controller that makes $\boldsymbol{y}_p(k+q)$ to zero after q time steps. The integer q is, thus, called the deadbeat control horizon, whereas the integer p is commonly referred to as the system identification (or observer) horizon. The condition $qr \geq pm$ means that the control action should not be faster than the state observation. That makes the physical sense.

In view of Eq. (17), the first r rows of $\boldsymbol{u}_q(k)$ is the input vector $\boldsymbol{u}(k)$ at time k. Define F_{c1} and F_{o1} as the first r rows of F_c and F_o, respectively. The control action at $\boldsymbol{u}(k)$ should be

$$\boldsymbol{u}(k) = [\text{the first } r \text{ rows of } F_c]\, \boldsymbol{v}_p(k-p) = F_{c1}\boldsymbol{v}_p(k-p) \tag{26}$$

where F_{c1} is the control gain matrix to be determined. The control gain matrix is constant for a linear time-invariant system. Otherwise, it may be time varying.

The first r rows of Eq. (23) produce

$$\boldsymbol{u}(k) = (\text{the first } r \text{ rows of } [F_c \quad F_o]) \begin{bmatrix} \boldsymbol{v}_p(k-p) \\ \boldsymbol{v}_p(k+q) \end{bmatrix} = [F_{c1} \quad F_{o1}] \begin{bmatrix} \boldsymbol{v}_p(k-p) \\ \boldsymbol{v}_p(k+q) \end{bmatrix} \tag{27}$$

Equation (27) indicates that the input $\boldsymbol{u}(k)$ is related to the past input sequence $\boldsymbol{u}(k-p)$ to $\boldsymbol{u}(k-1)$ and output sequence $\boldsymbol{y}(k-p)$ to $\boldsymbol{y}(k-1)$ and future input sequence $\boldsymbol{u}(k+q)$ to $\boldsymbol{u}(k+q+p-1)$ and output sequence $\boldsymbol{y}(k+q)$ to $\boldsymbol{y}(k+q+p-1)$. There is a total of q time steps gap from k to $k+q-1$. A different integer q produces a different set of coefficient matrices F_{c1} and F_{o1} that satisfies Eq. (27).

Recursive Least-Squares Algorithm

There are many recursive algorithms[12] available to solve the least-squares problem. The classical least-squares method is the most straightforward approach and is also the basis for the others. The classical recursive method is briefly described here.

Because the time index k in Eq. (27) is a dummy variable, let us set $k = k' - p - q$ and treat k' as the current time. Equation (27) can be written in a compact matrix form

$$\boldsymbol{u}(k'-q-p) = [F_{c1} \quad F_{o1}] \begin{bmatrix} \boldsymbol{v}_p(k'-q-2p) \\ \boldsymbol{v}_p(k'-p) \end{bmatrix} = F\boldsymbol{z}_p(k'-1) \tag{28}$$

where

$$F = [F_{c1} \quad F_{o1}], \qquad \boldsymbol{v}(k') = \begin{bmatrix} \boldsymbol{y}(k') \\ \boldsymbol{u}(k') \end{bmatrix} \tag{29}$$

and

$$\boldsymbol{z}_p(k'-1) = \begin{bmatrix} \boldsymbol{v}(k'-q-2p) \\ \vdots \\ \boldsymbol{v}(k'-q-p-1) \\ \boldsymbol{v}(k'-p) \\ \vdots \\ \boldsymbol{v}(k'-1) \end{bmatrix} \tag{30}$$

First, define the following quantities:

$$\boldsymbol{G}_p(k') = \frac{\boldsymbol{z}_p^T(k')P_p(k'-1)}{1 + \boldsymbol{z}_p^T(k')P_p(k'-1)\boldsymbol{z}_p(k')} \tag{31}$$

$$\hat{\boldsymbol{u}}(k'-q-p+1) = \hat{F}(k')\boldsymbol{z}_p(k') \tag{32}$$

Next, compute the following quantities:

$$P_p(k') = P_p(k'-1)[I - \boldsymbol{z}_p(k')\boldsymbol{G}_p(k')] \tag{33}$$

$$\hat{F}(k'+1) = \hat{F}(k') + [\boldsymbol{u}(k'-q-p+1) - \hat{\boldsymbol{u}}(k'-q-p+1)]\boldsymbol{G}_p(k') \tag{34}$$

Equations (31–34) constitute the fundamental recursive least-squares (RLS) formulation for identifying the gain matrix F including F_{c1} for the deadbeat controller design and F_{o1} for the need of system identification. The initial values of $P_p(0)$ and $\hat{Y}_p(1)$ can be either assigned or obtained by performing a small batch least squares after gathering a sufficient number of data. If initial values are to be chosen, $P_p(0)$ and $\hat{F}(1)$ can be assigned as $dI_{2p(r+m)}$ and $0_{r\times 2p(r+m)}$, respectively, where d is a large positive number. The positive constant d is the only parameter required for the initialization. The proper choice of d is based on practical experience. It is known from the RLS algorithm that the initialization introduces a bias into the parameter estimate $\hat{F}(k')$ produced by the RLS method. For large data lengths, the exact value of the initialization constant is not important.

The computational steps for the recursive deadbeat control method are summarized in the following.

1) Form the vector $\boldsymbol{z}_p(k')$ as shown in Eq. (30) with the new input $\boldsymbol{v}(k')$ as the last $r+m$ rows.

2) Compute the gain vector $\boldsymbol{G}_p(k')$ by inserting $P_p(k'-1)$ and $\boldsymbol{z}_p(k')$ in Eq. (31). In this step, one should compute $\boldsymbol{z}_p^T(k')P_p(k'-1)$ first, and then use the result to calculate $[\boldsymbol{z}_p^T(k')P_p(k'-1)]\boldsymbol{z}_p(k')$.

3) Compute the estimated input $\hat{\boldsymbol{u}}(k'-q-p+1)$ by substituting $\hat{F}(k')$ and $\boldsymbol{z}_p(k')$ into Eq. (32). Note that the time step $k'-q-p+1$ for the estimated input $\hat{\boldsymbol{u}}(k'-q-p+1)$ is $q+p-1$ behind than the current time k'.

4) Update $P_p(k'-1)$ to obtain $P_p(k')$ with $\boldsymbol{z}_p(k')$ formed from the first step and $\boldsymbol{G}_p(k')$ computed from the second step.

5) Update $\hat{F}(k')$ to obtain $\hat{F}(k'+1)$ from Eq. (34) with the input signal $\boldsymbol{u}(k'-q-p+1)$, the estimated input $\hat{\boldsymbol{u}}(k-q-p+1)$, and the computed gain $\boldsymbol{G}_p(k')$.

No matrix inverse is involved in these computational steps. Updating $P_p(k')$ and $\boldsymbol{G}_p(k')$ takes more time than computing other quantities. The recursive procedure derived for updating the least-squares solution $\hat{F}(k')$ is very general in the sense that it is valid for any linear equation.

It is noted that some accuracy may be lost when a least-squares problem is solved using the classical approach as described in this section. The reason is that the input and output data are squared to compute the data correlation. There is another method based on orthogonal transformation to avoid the computation of data correlation for the least-squares estimate. The method is commonly called a square root method[8] because it works with the square root of the data correlation.

As soon as the gain matrix F_{c1} is identified from the RLS solution of Eq. (34), the control force at time $k'+1$ can then be computed from Eq. (26) to yield

$$\boldsymbol{u}(k'+1) = F_{c1}\boldsymbol{v}_p(k'+1-p) \tag{35}$$

where $\boldsymbol{v}_p(k'+1-p)$ is defined from Eq. (22). Observe that the control signal $\boldsymbol{u}(k'-q-p)$ on the left-hand side of the identification process, Eq. (28), is $q+p+1$ steps further behind than $\boldsymbol{u}(k'+1)$ in the control equation (35).

Feedback and Feedforward for Disturbance Input

In addition to the control input, there may be other disturbance inputs applied to the system. Some types of disturbances come from the known sources that can be measured. Another type of disturbances is not known, but its correlation is known. This section addresses the predictive feedback designs including feedforward from the disturbance inputs that are measurable or predictable.

With the disturbance input involved, the multistep output prediction equation becomes

$$\boldsymbol{y}_p(k+q) = \mathcal{T}_c\boldsymbol{u}_{cq}(k) + \mathcal{T}_d'\boldsymbol{u}_{d(q+p)}(k) + \mathcal{T}_o\boldsymbol{u}_{cp}(k+q) + \mathcal{B}_c'\boldsymbol{u}_{cp}(k-p) + \mathcal{B}_d'\boldsymbol{u}_{dp}(k-p) + \mathcal{A}'\boldsymbol{y}_p(k-p) \tag{36}$$

where

$$\boldsymbol{u}_{d(q+p)}(k) = \begin{bmatrix} \boldsymbol{u}_d(k) \\ \boldsymbol{u}_d(k+1) \\ \vdots \\ \boldsymbol{u}_d(k+q+p-1) \end{bmatrix}$$

$$\boldsymbol{u}_{cq}(k) = \begin{bmatrix} \boldsymbol{u}_c(k) \\ \boldsymbol{u}_c(k+1) \\ \vdots \\ \boldsymbol{u}_c(k+q-1) \end{bmatrix} \tag{37}$$

The vectors $\boldsymbol{y}_p(k+q)$ and $\boldsymbol{y}_p(k-p)$ are defined in Eq. (14). The subscripts c and d are the quantities related to the control force and the disturbance force, respectively. The other quantities $\boldsymbol{u}_{cp}(k+q)$ and $\boldsymbol{u}_{cp}(k-p)$ and quantity $\boldsymbol{u}_{dp}(k-p)$ are similarly defined as $\boldsymbol{u}_{cq}(k)$ and $\boldsymbol{u}_{d(q+p)}(k)$, respectively, shown in Eq. (37). The form of the matrix $\mathcal{T}_d'$ associated with the disturbances $\boldsymbol{u}_d$ is similar to $\mathcal{T}'$ defined in Eq. (15). The matrix $\mathcal{T}_c'$ is a $pm \times qr_c$ matrix, where r_c is the number of control inputs, and $\mathcal{T}_d'$ is a $pm \times qr_d$ matrix, where r_d is the number of disturbance inputs. The forms of $\mathcal{B}_c'$ and $\mathcal{B}_d'$ are also similar but corresponding to different type of forces. Note that $\mathcal{T}_c$, $\mathcal{T}_o$, and $\mathcal{B}_c'$ are quantities associated with the control force $\boldsymbol{u}_c(k)$.

An equation similar to Eq. (19) can, thus, be derived as

$$\boldsymbol{u}_{cq}(k) = \mathcal{T}_c^\dagger \boldsymbol{y}_p(k+q) - \mathcal{T}_c^\dagger \mathcal{T}_o \boldsymbol{u}_{cp}(k+q) - \mathcal{T}_c^\dagger \mathcal{T}_d' \boldsymbol{u}_{d(q+p)}(k) - \mathcal{T}_c^\dagger \mathcal{A}' \boldsymbol{y}_p(k-p) - \mathcal{T}_c^\dagger \mathcal{B}_c' \boldsymbol{u}_{cp}(k-p) - \mathcal{T}_c^\dagger \mathcal{B}_d' \boldsymbol{u}_{dp}(k-p) \tag{38}$$

or in matrix form

$$\boldsymbol{u}_{cq}(k) = \begin{bmatrix} -\mathcal{T}_c^\dagger \mathcal{A}' & -\mathcal{T}_c^\dagger \mathcal{B}_c' & -\mathcal{T}_c^\dagger \mathcal{B}_d' & \mathcal{T}_c^\dagger & -\mathcal{T}_c^\dagger \mathcal{T}_o & -\mathcal{T}_c^\dagger \mathcal{T}_d' \end{bmatrix} \times \begin{bmatrix} \boldsymbol{y}_p(k-p) \\ \boldsymbol{u}_{cp}(k-p) \\ \boldsymbol{u}_{cd}(k-p) \\ \boldsymbol{y}_p(k+q) \\ \boldsymbol{u}_{cp}(k+q) \\ \boldsymbol{u}_{d(q+p)}(k) \end{bmatrix} \tag{39}$$

Define the following notations:

$$F_c' = \begin{bmatrix} -\mathcal{T}_c^\dagger \mathcal{A}' & -\mathcal{T}_c^\dagger \mathcal{B}_c' & -\mathcal{T}_c^\dagger \mathcal{B}_d' \end{bmatrix}$$
$$F_{co}' = \begin{bmatrix} \mathcal{T}_c^\dagger & -\mathcal{T}_c^\dagger \mathcal{T}_o \end{bmatrix}, \qquad F_d' = -\mathcal{T}_c^\dagger \mathcal{T}_d' \tag{40}$$

and

$$\boldsymbol{v}_p(k-p) = \begin{bmatrix} \boldsymbol{y}_p(k-p) \\ \boldsymbol{u}_{cp}(k-p) \\ \boldsymbol{u}_{cd}(k-p) \end{bmatrix}, \qquad \boldsymbol{v}_{cp}(k+q) = \begin{bmatrix} \boldsymbol{y}_p(k+q) \\ \boldsymbol{u}_{cp}(k+q) \end{bmatrix} \tag{41}$$

where F_c' is a $qr_c \times p(m+r_c+r_d)$ matrix but F_{co}' is a $qr_c \times p(m+r_c)$ matrix. The quantity $\boldsymbol{v}_p(k-p)$ is a $p(m+r_c+r_d) \times 1$ column vector, whereas $\boldsymbol{v}_{cp}(k+q)$ is a $p(m+r_c) \times 1$ column vector. Thus, Eq. (39) becomes

$$\boldsymbol{u}_{cq}(k) = [F_c' \quad F_{co}' \quad F_d'] \begin{bmatrix} \boldsymbol{v}_p(k-p) \\ \boldsymbol{v}_{cp}(k+q) \\ \boldsymbol{u}_{d(q+p)}(k) \end{bmatrix} \tag{42}$$

For any given input and output data, there exists a set of F_c', F_{co}', and F_d' satisfying Eq. (42).

Let us assume that the input vector $\boldsymbol{u}_{cq}(k)$ is chosen such that

$$\boldsymbol{u}_{cq}(k) = F_c' \boldsymbol{v}_p(k-p) \tag{43}$$

From Eq. (42), the output after q time steps is then governed by

$$F_{co}' \boldsymbol{v}_{cp}(k+q) + F_d' \boldsymbol{u}_{d(q+p)}(k) = 0$$

or from Eq. (39)

$$\mathcal{T}_c^\dagger \boldsymbol{y}_p(k+q) = \mathcal{T}_c^\dagger \mathcal{T}_o \boldsymbol{u}_{cq}(k+q) + \mathcal{T}_c^\dagger \mathcal{T}_d' \boldsymbol{u}_{d(q+p)}(k) \tag{44}$$

Here we have assumed that the disturbance is measurable. If the disturbance is not predictable or measurable, Eq. (43) is not valid. Equation (44) indicates that the output vector from time $k+q$ is generated by the control vector from $k+q$ and the disturbance vector from k. If the disturbance $\boldsymbol{u}_d(k)$ such as the random signal is not predictable, then we cannot use the future disturbance signal, i.e., beyond the current time k, for a feedforward design for the control force $\boldsymbol{u}_c(k)$ at time k. As a result, the feedforward design included in the control law, Eq. (43), is the only way that can be implemented, because it uses only the disturbance signal before time k.

From Eq. (43), the control action $\boldsymbol{u}_c(k)$ at time k is

$$\boldsymbol{u}_c(k) = (\text{the first } r \text{ rows of } F_c')\boldsymbol{v}_p(k-p) = F_{c1}' \boldsymbol{v}_p(k-p) \tag{45}$$

where F_{c1}' is the control gain matrices to be determined from Eq. (42). The first r rows of Eq. (42) produce

$$\boldsymbol{u}_c(k) = (\text{the first } r \text{ rows of } [F_c' \quad F_{co}' \quad F_d']) \begin{bmatrix} \boldsymbol{v}_p(k-p) \\ \boldsymbol{v}_{cp}(k+q) \\ \boldsymbol{u}_{d(q+p)}(k) \end{bmatrix} = \begin{bmatrix} F_{c1}' & F_{co1}' & F_{d1}' \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_p(k-p) \\ \boldsymbol{v}_{cp}(k+q) \\ \boldsymbol{u}_{d(q+p)}(k) \end{bmatrix} \tag{46}$$

which can also be computed recursively, as shown in the preceding section. Note again that the time index k is a dummy integer. The gain matrix F_{c1}' is actually updated in a different timescale than that for the estimated input. Indeed in practice, the integer k in Eq. (46) should be reset to be k' such that $k = k' - q - p$. Equation (46) can then be rewritten as

$$\boldsymbol{u}_c(k'-q-p) = [F_{c1}' \quad F_{co1}' \quad F_{d1}'] \begin{bmatrix} \boldsymbol{v}_p(k'-q-2p) \\ \boldsymbol{v}_{cp}(k'-p) \\ \boldsymbol{u}_{d(q+p)}(k'-q-p) \end{bmatrix} \tag{47}$$

When k' is treated as the current time step, the data vectors $\boldsymbol{v}_p(k'-q-2p)$, $\boldsymbol{v}_{cp}(k'-p)$, and $\boldsymbol{u}_{d(q+p)}(k'-q-p)$ are all known and given under the condition that the disturbance is measurable. As soon as the gain matrix F_{c1}' is identified from the RLS solution of Eq. (47), the control force at time $k'+1$ can then be computed from Eq. (42) to yield

$$\boldsymbol{u}_c(k'+1) = F_{c1}' \boldsymbol{v}_p(k'+1-p) \tag{48}$$

where $\boldsymbol{v}_p(k'+1-p)$ is defined from Eq. (41). Again, note that the control signal $\boldsymbol{u}_c(k'-q-p)$ on the left-hand side of the identification process, Eq. (47), is $q+p+1$ steps further behind than $\boldsymbol{u}_c(k'+1)$ in the control equation (48).

Numerical Example

A simple spring–mass–damper system is used to illustrate various controllers. Several different cases will be discussed ranging from single-input/single-output to multi-input/multi-output. First, the noise-free case is shown, and then the case with additive measurement noise is discussed.

Consider a three-degree-of-freedom spring–mass–damper system

$$M\ddot{w} + \Xi\dot{w} + Kw = \boldsymbol{u}$$

where

$$M = \begin{bmatrix} m_1 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & m_3 \end{bmatrix}, \qquad \Xi = \begin{bmatrix} \zeta_1 + \zeta_2 & -\zeta_2 & 0 \\ -\zeta_2 & \zeta_2 + \zeta_3 & -\zeta_3 \\ 0 & -\zeta_3 & \zeta_3 \end{bmatrix}$$

$$K = \begin{bmatrix} k_1 + k_2 & -k_2 & 0 \\ -k_2 & k_2 + k_3 & -k_3 \\ 0 & -k_3 & k_3 \end{bmatrix}$$

$$w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}, \qquad \boldsymbol{u} = \begin{bmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \\ \boldsymbol{u}_3 \end{bmatrix}$$

and m_i, k_i, and ζ_i, $i = 1, 2, 3$, are the mass, spring stiffness, and damping coefficients, respectively. For this system, the order of the equivalent state-state representation is six ($n = 6$). The control force applied to each mass is denoted by $\boldsymbol{u}_i$, $i = 1, 2, 3$. The variables w_i, $i = 1, 2, 3$, are the positions of the three masses measured from their equilibrium positions. In the simulation, $m_1 = m_2 = m_3 = 1$ kg, $k_1 = k_2 = k_3 = 1000$ N/m, $\zeta_1 = \zeta_2 = \zeta_3 = 0.1$ N-s/m. The system is sampled at 25 Hz ($\Delta t = 0.04$ s). Let the measurements $\boldsymbol{y}_i$ be the accelerations of the three masses, $\boldsymbol{y}_i = \ddot{w}_i$, $i = 1, 2, 3$.

Let us consider a single-control-input and single-output case, where the control input to the system is the force on the first mass, i.e., $u_c = u_1$, and the output is the acceleration of the third mass, i.e., $y = \ddot{w}_3$, (noncollocated actuator–sensor). Therefore, the smallest order of the ARX model p is six corresponding to a deadbeat observer, and the smallest value for q is also six corresponding to a deadbeat controller, which will bring the entire system to rest in exactly six time steps. Note that this is a nonminimum-phase system.

Consider the case where the controller is computed with $q = 6$. Let the initial guess for $P_p(0)$ and $\hat{F}(1)$ shown in Eqs. (31) and (32) be $1000 I_{24 \times 24}$ and $0_{1 \times 24}$, respectively. The initial input signal before the control action is on is a sequence of normally distributed random numbers with zero mean and unit variance. Let the control action be turned on at the data point 30. In other words, we do not wait very long to close the system loop as soon as the first nonzero vector $\bar{v}_p(k)$ defined in Eq. (30) is formed. Figure 1 shows the open-loop and closed-loop histories of input and output. The solid line is the open-loop response and the dash-dot line is the closed-loop response. The control gain starts with a zero vector and ends with the controller

$$\begin{aligned} \boldsymbol{u}_c(k) = {}& \\ & -0.3846\boldsymbol{u}_c(k-1) + 0.7218\boldsymbol{u}_c(k-2) + 0.2535\boldsymbol{u}_c(k-3) \\ & - 0.0682\boldsymbol{u}_c(k-4) - 0.0150\boldsymbol{u}_c(k-5) + 0.0000\boldsymbol{u}_c(k-6) \\ & - 0.9823\boldsymbol{y}(k-1) - 0.7300\boldsymbol{y}(k-2) - 1.2898\boldsymbol{y}(k-3) \\ & - 0.1816\boldsymbol{y}(k-4) + 0.1909\boldsymbol{y}(k-5) - 0.0008\boldsymbol{y}(k-6) \end{aligned}$$

The controller converges to the one using the batch approach shown in the Appendix. It is seen from Fig. 1 that it takes more than six steps after the control action is on to make the control gain converge. As soon as the control gain converges, the system becomes deadbeat within six steps.

Let the output be added with some measurement noise so that the signal-to-noise ratio is 10.5, i.e., the output norm divided by the noise norm. The noise is random normally distributed with zero mean. Set the values of p and q to be the same as that for the deadbeat, i.e., $p = q = 6$. Again, let the control action be turned on at the data point 30. The open-loop and closed-loop time histories are shown in Fig. 2. The control gain starts with a zero vector and ends with the controller

$$\begin{aligned} \boldsymbol{u}_c(k) = {}& \\ & -0.0198\boldsymbol{u}_c(k-1) + 0.6153\boldsymbol{u}_c(k-2) + 0.0326\boldsymbol{u}_c(k-3) \\ & + 0.0169\boldsymbol{u}_c(k-4) + 0.0355\boldsymbol{u}_c(k-5) - 0.0356\boldsymbol{u}_c(k-6) \\ & - 0.3581\boldsymbol{y}(k-1) - 0.3840\boldsymbol{y}(k-2) - 0.6640\boldsymbol{y}(k-3) \\ & - 0.0735\boldsymbol{y}(k-4) + 0.0772\boldsymbol{y}(k-5) - 0.1345\boldsymbol{y}(k-6) \end{aligned}$$

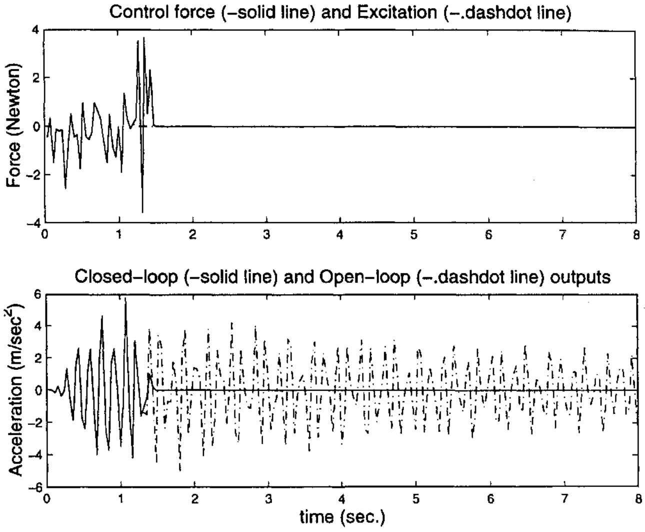

Fig. 1 Open-loop and closed-loop time histories for the noise-free case with $p = q = 6$.

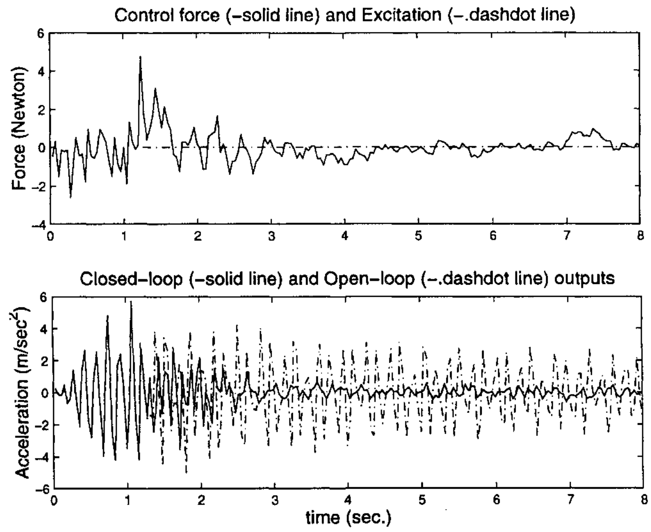

Fig. 2 Open-loop and closed-loop time histories with $p = q = 6$ and output noise.

Although the final control gain looks considerably different from the one shown earlier for the noise-free case, the performance for both noise-free and noisy cases is quite similar. Increasing the value of q does not seem to improve the performance. In some cases with q larger than p, the performance is much worse than for the case with $p = q$ when the control action is turned on too early. Given sufficient time steps for the control gain to converge to a reasonable level, the performance may be improved somewhat, particularly for the case where $q > p$. One may raise the question of whether the recursive controller design works for the case where the order of the controller is smaller than the order of the system, i.e., p is smaller than 6 for this example. Let us choose $p = q = 4$. Using the same set of data for the preceding case, the input and output time histories are shown in Fig. 3. Obviously, the performance is as good as the ones shown earlier. In practice, the order of a system is unknown and, thus, there is a great possibility that the values of p may be smaller than the true one.

Assume that there is a measurable disturbance applied at the second mass. The disturbance signal is assumed to be random normally distributed with zero mean and unit variance. The same measurement noise as given for the preceding case is also added to the output. Set the values of p and q to be $p = q = 6$. The open-loop and closed-loop time histories are shown in Fig. 4. The control action starts at the data point 30. Equation (47) is used in this case

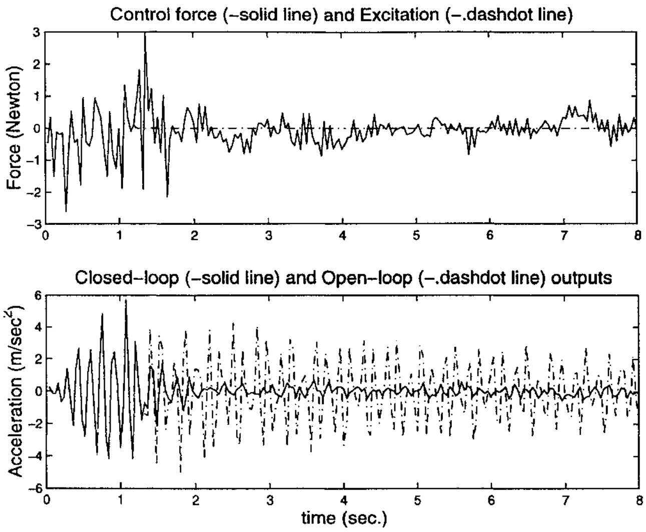

Fig. 3 Open-loop and closed-loop time histories with $p = q = 4$ and output noise.

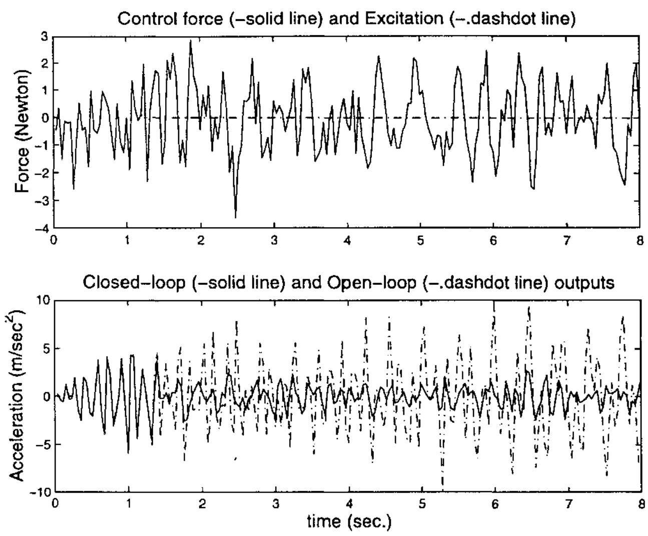

Fig. 4 Open-loop and closed-loop (feedback and feedforward) time histories with $p = q = 6$ and output noise.

to determine the feedback and feedforward gains. The converged controller is

$$
\begin{aligned}
u_c(k) = \\
&-0.1094u_c(k-1) + 0.4573u_c(k-2) - 0.1879u_c(k-3) \\
&+ 0.0327u_c(k-4) + 0.0551u_c(k-5) + 0.0174u_c(k-6) \\
&- 0.3812y(k-1) - 0.2879y(k-2) - 0.7695y(k-3) \\
&+ 0.0397y(k-4) - 0.0709y(k-5) - 0.1063y(k-6) \\
&- 0.4078u_d(k-1) - 0.0095u_d(k-2) + 0.4452u_d(k-3) \\
&+ 0.0722u_d(k-4) + 0.1334u_d(k-5) + 0.3071u_d(k-6)
\end{aligned}
$$

The closed-loop output response is considerably reduced in comparison with the open-loop response.

Several other cases (not shown) have been studied including random and periodic disturbances with or without feedforward. It indicates that, given sufficient time steps before the control action is on for the control gain to converge, the recursive control design works quite well, and the results are similar to those shown earlier. On the other hand, a small amount of dither can be added to the control signal to speed up the rate of convergence for the control gain. The dither is band-limited white noise and will ensure that the closed-loop system remains persistently excited. Dither provides one way to ensure an accurate system identification in a closed-loop system. Of course, the dither may be turned off as soon as the control gain converges to a reasonable level.

Concluding Remarks

A new recursive predictive control method has been presented. System identification was reformulated in such a way that it fits better for predictive controller designs. In other words, the conventional thinking in system identification has been reoriented to focus on the control design process. The conventional procedure for any controller designs includes two steps, i.e., first perform system identification within an acceptable level of accuracy and then conduct a controller design. The error in system identification will likely be accumulated and carried through the controller design process. As a result, the conventional approach tends to introduce more error in the controller design than system identification itself. The method derived in this paper uses a different approach for system identification to eliminate the additional controller design process. For noise-free cases, both conventional and new approaches produce an identical result if the predictive controller is unique. For nonunique controllers, the new approach provides the control gain smaller in norm than that from the conventional approach. This is because, instead of minimizing the output error, the new approach minimizes the input error to compute the control gain. The proposed recursive method has a considerable advantage of computational speed. For noisy cases, numerical simulations have showed that the new method is more robust than the other methods.

Appendix: Direct Algorithm

There are two predictive control designs, namely, the indirect algorithm and the direct algorithm. The indirect algorithm is based on Eq. (1) with the assumption that the parameters $\alpha_1, \alpha_2, \ldots, \alpha_p$ and $\beta_0, \beta_1, \ldots, \beta_p$, are given a priori. The direct algorithm uses the input and output data directly, without explicitly involving the parameters $\alpha_1, \alpha_2, \ldots, \alpha_p$ and $\beta_0, \beta_1, \ldots, \beta_p$ to compute feedback control parameters. To achieve the goal, first start with Eqs. (16–18) and form the following input and output matrices:

$$
Y_s(k) = [y_s(k) \quad y_s(k+1) \quad \cdots \quad y_s(k+N-1)]
$$

$$
= \begin{bmatrix} y(k) & y(k+1) & \cdots & y(k+N-1) \\ y(k+1) & y(k+2) & \cdots & y(k+N) \\ \vdots & \vdots & \ddots & \vdots \\ y(k+s-1) & y(k+s) & \cdots & y(k+s+N-2) \end{bmatrix} \tag{A1}
$$

$$
U_s(k) = [u_s(k) \quad u_s(k+1) \quad \cdots \quad u_s(k+N-1)]
$$

$$
= \begin{bmatrix} u(k) & u(k+1) & \cdots & u(k+N-1) \\ u(k+1) & u(k+2) & \cdots & u(k+N) \\ \vdots & \vdots & \ddots & \vdots \\ u(k+s-1) & u(k+s) & \cdots & u(k+s+N-2) \end{bmatrix}
$$

where N is an integer. The integer s is a dummy integer that can be either p or q. For example, the matrices $Y_p(k+q)$ and $U_p(k+q)$ are defined by replacing s by p and k by $k+q$. Similarly, the matrices $Y_p(k-p)$ and $U_p(k-p)$ are defined by replacing s by p and k by $k-p$. The data matrices $U_p(k+q)$ and $Y_p(k+q)$ include the input and output data up to the time step $k+q+p+N-2$, whereas $U_p(k-p)$ and $Y_p(k-p)$ have data up to the time step $k+N-2$.

Application of Eq. (16) yields

$$
Y_p(k+q) = \mathcal{T}_o U_p(k+q) + \mathcal{T}_c U_q(k) + \mathcal{B}' U_p(k-p) + \mathcal{A}' Y_p(k-p) \tag{A2}
$$

or

$$
Y_p(k+q) = [\mathcal{T}_o \quad \mathcal{T}_c \quad \mathcal{B}' \quad \mathcal{A}'] \begin{bmatrix} U_p(k+q) \\ U_q(k) \\ U_p(k-p) \\ Y_p(k-p) \end{bmatrix} \tag{A3}
$$

Let the integers p, q, and N be chosen large enough such that Eq. (A3) produces the following least-squares solution:

$$[\mathcal{T}_o \quad \mathcal{T}_c \quad \mathcal{B}' \quad \mathcal{A}'] = Y_p(k+q)\begin{bmatrix} U_p(k+q) \\ U_q(k) \\ U_p(k-p) \\ Y_p(k-p) \end{bmatrix}^{\dagger} \tag{A4}$$

where † is the pseudoinverse. At this moment, all input and output data are measured from the open-loop system, before any control action begins.

Let the control action be turned on at time step k and ended at $k+q$. In other words, the control action occurs only from $\boldsymbol{u}(k)$ to $\boldsymbol{u}(k+q-1)$, beyond which the control action is zero, i.e., $\boldsymbol{u}(k+q) = \boldsymbol{u}(k+q+1) = \cdots = 0$. Under this condition, Eq. (16) produces the following equation:

$$\boldsymbol{y}_p(k+q) = \mathcal{T}_c \boldsymbol{u}_q(k) + \mathcal{B}' \boldsymbol{u}_p(k-p) + \mathcal{A}' \boldsymbol{y}_p(k-p) \tag{A5}$$

Thus, it is clear that the following equality:

$$\boldsymbol{u}_q(k) = -[\mathcal{T}_c]^{\dagger}[\mathcal{B}' \boldsymbol{u}_p(k-p) + \mathcal{A}' \boldsymbol{y}_p(k-p)] \tag{A6}$$

will bring $\boldsymbol{y}_p(k+q)$ to rest, i.e.,

$$\boldsymbol{y}_p(k+q) = \begin{bmatrix} \boldsymbol{y}(k+q) \\ \boldsymbol{y}(k+q+1) \\ \vdots \\ \boldsymbol{y}(k+q+p-1) \end{bmatrix} = 0$$

under the condition that $[\mathcal{T}_c \mathcal{T}_c]^{\dagger} = I$ (identity matrix). With the triple $[\mathcal{T}_c, \mathcal{B}', \mathcal{A}']$ identified from Eq. (A4), the control law from Eq. (A6) can be applied to compute the control gain parameters as

$$\begin{aligned} \boldsymbol{u}(k) &= -\text{first } r \text{ rows of } \left\{[\mathcal{T}_c]^{\dagger}\right\}[\mathcal{B}' \boldsymbol{u}_p(k-p) + \mathcal{A}' \boldsymbol{y}_p(k-p)] \\ &= \alpha_1^c \boldsymbol{y}(k-1) + \alpha_2^c \boldsymbol{y}(k-2) + \cdots + \alpha_p^c \boldsymbol{y}(k-p) \\ &\quad + \beta_1^c \boldsymbol{u}(k-1) + \beta_2^c \boldsymbol{u}(k-2) + \cdots + \beta_p^c \boldsymbol{u}(k-p) \end{aligned} \tag{A7}$$

where the superscript c identifies the control parameters. The feedback control parameters $\alpha_1^c, \ldots, \alpha_p^c$ and $\beta_1^c, \beta_2^c, \ldots, \beta_p^c$ are to be used to compute the current control signal $\boldsymbol{u}(k)$ using the past p input and output measurements. The control action is supposed to bring the output to zero for all time steps larger than $k+q$. Along with the desired zero input $\boldsymbol{u}(k+q)$ and beyond, the system should be at rest, i.e., deadbeat, beyond time step $k+q$. That is in theory. In practice, when the system has input and output uncertainties, the control action can only bring the output down to the level of uncertainties.

References

[1]Kwakernaak, H., and Sivan, R., *Linear Optimal Control Systems*, Wiley-Interscience, New York, 1972.

[2]Kailath, T., *Linear Systems*, Prentice–Hall, Englewood Cliffs, NJ, 1980.

[3]Juang, J. N., Lim, K. B., and Junkins, J. L., "Robust Eigensystem Assignment for Flexible Structures," *Journal of Guidance, Control, and Dynamics*, Vol. 12, No. 3, 1989, pp. 381–387.

[4]Juang, J. N., and Phan, M., "Robust Controller Designs for Second-Order Dynamic Systems: A Virtual Passive Approach," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 5, 1992, pp. 1192–1198.

[5]Morris, K. A., and Juang, J. N., "Dissipative Controller Designs for Second-Order Dynamic Systems," *IEEE Transactions on Automatic Control*, Vol. 39, No. 5, 1994, pp. 1056–1063.

[6]Goodwin, G. C., and Sin, K. S., *Adaptive Filtering, Prediction, and Control*, Prentice–Hall, Englewood Cliffs, NJ, 1984.

[7]Ydstie, B. E., "Extended Horizon Adaptive Control," *Proceedings of the 9th IFAC World Congress*, Vol. 7, Pergamon, Oxford, England, UK, 1984, pp. 133–138.

[8]Astrom, K. J., and Wittenmark, B., *Adaptive Control*, 2nd ed., Addison–Wesley, Reading, MA, 1995, Chap. 11.

[9]Phan, M., Horta, L. G., Juang, J.-N., and Longman, R. W., "Linear System Identification via an Asymptotically Stable Observer," *Journal of Optimization Theory and Applications*, Vol. 79, No. 1, 1993, pp. 59–86.

[10]Juang, J.-N., Phan, M., Horta, L. G., and Longman, R. W., "Identification of Observer/Kalman Filter Markov Parameters: Theory and Experiments," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 2, 1993, pp. 320–329.

[11]Phan, M., Horta, L. G., Juang, J.-N., and Longman, R. W., "Improvement of Observer/Kalman Filter Identification (OKID) by Residual Whitening," *Journal of Vibrations and Acoustics*, Vol. 117, April 1995, pp. 232–239.

[12]Juang, J.-N., *Applied System Identification*, Prentice–Hall, Englewood Cliffs, NJ, 1994.

[13]Kida, T., Ohkami, Y., and Sambongi, S., "Poles and Transmission Zeros of Flexible Spacecraft Control Systems," *Journal of Guidance, Control, and Dynamics*, Vol. 8, No. 2, 1985, pp. 208–213.

[14]Williams, T., "Transmission Zero Bounds for Large Structures, with Applications," *Journal of Guidance, Control, and Dynamics*, Vol. 12, No. 1, 1989, pp. 33–38.

[15]Williams, T., and Juang, J. N., "Sensitivity of the Transmission Zeros of Flexible Space Structures," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 2, 1992, pp. 368–375.

[16]Williams, T., and Juang, J. N., "Pole/Zero Cancellations in Flexible Space Structures," *Journal of Guidance, Control, and Dynamics*, Vol. 13, No. 4, 1990, pp. 684–690.

[17]Williams, T., "Model Order Effects on the Transmission Zeros of Flexible Space Structures," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 2, 1992, pp. 540–543.

[18]Williams, T., "Transmission Zeros of Non-Collocated Flexible Structures: Finite-Dimensional Effects," AIAA Paper 92-5040, April 1992.

[19]Lin, J.-L., and Juang, J.-N., "Sufficient Conditions for Minimum-Phase Second-Order Linear Systems," *Journal of Vibration and Control*, Vol. 1, No. 2, 1995, pp. 183–199.

[20]Hyland, D. C., "Neural Network Architectures for On-Line System Identification and Adaptively Optimized Control," *Proceedings of the 30th IEEE Conference on Decision and Control*, Vol. 3, Inst. of Electrical and Electronics Engineers, New York, 1991, pp. 2552–2557.

[21]Hyland, D. C., and Juang, J.-N., "Series Parallel Approach to Identification of Dynamic Systems," U.S. Patent No. 5680513, Oct. 1997.

[22]Phan, M., Juang, J.-N., and Hyland, D. C., "On Neural Networks in Identification and Control of Dynamic Systems," *Wave Motion, Intelligent Structures, and Nonlinear Mechanics*, edited by S. Guran and D. Inman, World Scientific, River Edge, NJ, 1995, pp. 194–225.

[23]Soeterboek, R., *Predictive Control: A Unified Approach*, Prentice–Hall, Englewood Cliffs, NJ, 1992.

[24]Richalet, J., Rault, A., Testud, J. L., and Papon, J., "Model Predictive Heuristic Control: Applications to Industrial Processes," *Automatica*, Vol. 14, No. 5, 1978, pp. 413–428.

[25]De Keyser, R. M. C., and Van Cauwenberghe, A. R., "A Self-Tuning Multi-Step Predictor Application," *Automatica*, Vol. 17, No. 1, 1979, pp. 167–174.

[26]Peterka, V., "Predictor-Based Self-Tuning Control," *Automatica*, Vol. 20, No. 1, 1984, pp. 39–50.

[27]Clarke, D. W., Mohtadi, C., and Tuffs, P. S., "Generalized Predictive Control—Part I. The Basic Algorithm," *Automatica*, Vol. 23, No. 2, 1987, pp. 137–148.

[28]Clarke, D. W., Mohtadi, C., and Tuffs, P. S., "Generalized Predictive Control—Part II. Extensions and Interpretations," *Automatica*, Vol. 23, No. 2, 1987, pp. 149–160.

[29]Mosca, E., *Optimal, Predictive, and Adaptive Control*, Prentice–Hall, Englewood Cliffs, NJ, 1995.

[30]Bialasiewicz, J. T., Horta, L. G., and Phan, M., "Identified Predictive Control," *Proceedings of the American Control Conference*, Baltimore, MD, 1994, pp. 3243–3247.

[31]Phan, M. G., and Juang, J.-N., "Predictive Feedback Controllers for Stabilization of Linear Multivariable Systems," *Journal of Guidance, Control, and Dynamics* (to be published).

[32]Juang, J.-N., and Phan, M. G., "Deadbeat Predictive Controllers," NASA TM-112862, May 1997.