NASA/TM-2008-215320

A Computational Procedure for Identifying Bilinear Representations of Nonlinear Systems Using Volterra Kernels

Raymond G. Kvaternik and Walter A. Silva
Langley Research Center, Hampton, Virginia

June 2008

NASA/TM-2008-215320

A Computational Procedure for Identifying Bilinear Representations of Nonlinear Systems Using Volterra Kernels

Raymond G. Kvaternik and Walter A. Silva
Langley Research Center, Hampton, Virginia

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

June 2008

The use of trademarks or names of manufacturers in this report is for accurate reporting and does not constitute an official endorsement, either expressed or implied, of such products or manufacturers by the National Aeronautics and Space Administration.

Available from:

NASA Center for AeroSpace Information (CASI)
7115 Standard Drive
Hanover, MD 21076-1320
(301) 621-0390

National Technical Information Service (NTIS)
5285 Port Royal Road
Springfield, VA 22161-2171
(703) 605-6000

Table of Contents

Abstract

A computational procedure for identifying the state-space matrices corresponding to discrete bilinear representations of nonlinear systems is presented. A key feature of the method is the use of first- and second-order Volterra kernels (first- and second-order pulse responses) to characterize the system. The present method is based on an extension of a continuous-time bilinear system identification procedure given in a 1971 paper by Bruni, di Pillo, and Koch. The analytical and computational considerations that underlie the original procedure and its extension to the title problem are presented and described, pertinent numerical considerations associated with the process are discussed, and results obtained from the application of the method to a variety of nonlinear problems from the literature are presented. The results of these exploratory numerical studies are decidedly promising and provide sufficient credibility for further examination of the applicability of the method.

Nomenclature

$A_d, B_d, C_d, D_d, N_{dj}$	state-space matrices of discrete bilinear system (j = 1, 2, …, r)
$A_c, B_c, C_c, D_c, N_{cj}$	state-space matrices of continuous bilinear system (j = 1, 2, …, r)
G, H, K_j	subsidiary matrices appearing in Bruni, di Pillo, and Koch paper (j = 1, 2, …, r)
$h_i(\tau_1,\cdots,\tau_i)$	ith order continuous kernel
k, s	discrete time variables/indices
$k_1^{(p)}(k)$	first-order discrete Volterra kernels (p = 1, 2, …, r)
$k_2^{(p,q)}(k_1,k_2)$	second-order discrete Volterra kernels (p, q = 1, 2, …, r)
$\mathcal{K}_1$	matrix of first-order discrete Volterra kernels
$\mathcal{K}_2^{(j)}$	matrices of second-order discrete Volterra kernels (j = 1,2,…,r)
m	number of outputs/responses
MIMO	Multi-Input/Multi-Output
MISO	Multi-Input/Single Output
n	number of state variables

ncomp	number of components computed for each second-order kernel
N	number of time points in time history of discretized problem
r	number of inputs
SID	System IDentification
SISO	Single Input/Single Output
t, τ	continuous time variables
dt	continuous time increment
Δt	discrete time increment
T	time lag between double pulse inputs used for computing components of second-order kernels ($T = i\,\Delta T$ and $i\,dT$; $i = 0, 1, 2, 3, \ldots, ncomp\text{-}1$)
dT	continuous time increment ($dT = j\,dt$; j an integer ≥ 1) for varying time lag T
ΔT	discrete time increment ($\Delta T = j\,\Delta t$; j an integer ≥ 1) for varying time lag T
u	input vector
u_i	i^{th} component of input vector u
$w_1^{(p)}(\tau_1)$	first-order continuous kernels of Bruni et al. ($p = 1, 2, \ldots, r$)
$w_2^{(p,q)}(\tau_1, \tau_2)$	second-order continuous kernels of Bruni et al. ($p, q = 1, 2, \ldots, r$)
$W_1\ (\tau_1)$	matrix of first-order continuous kernels of Bruni et al.
$W_2^{(j)}(\tau_1, \tau_2)$	matrices of second-order continuous kernels of Bruni et al. ($j = 1,2,...,r$)
X	state vector
y	output vector
y_1, y_2, y_{12}	subsidiary pulse responses used to calculate kernels

$\delta_{-1}(t)$	continuous-time unit step function
δ_0	discrete-time unit pulse
$(\ )_c$	continuous-time quantity
$(\ )_d$	discrete-time quantity
$(\ )_{cid}$	identified continuous quantity
$(\ )_{did}$	discretized version of identified continuous quantity
$(\ )^{-1}$	matrix inverse
$(\ )^T$	matrix or vector transpose

Introduction

A research program with the objective of developing computationally-efficient, state-space-based techniques for the use of reduced-order computational fluid dynamics (CFD) aerodynamic models in nonlinear aeroservoelastic (NL-ASE) analysis of aerospace vehicles was initiated in the Aeroelasticity Branch at NASA Langley Research Center in 2000. The scope of the program is depicted in figure 1 and includes three major task areas: (1) reduced-order modeling of the aerodynamics acting on the system; (2) identification of an approximate bilinear state-space model of the nonlinear aerodynamics; and (3) aeroelastic or aeroservoelastic analysis of the coupled system. A key feature of the method is the use of aerodynamic Volterra kernels (aerodynamic pulse responses) to provide the basis for a reduced-order model (ROM) of the aerodynamics acting on the system (figure 1, box 1). The kernels are assumed to be identified using the output of a CFD aerodynamics model. Initial attention in implementing the computations associated with the tasks in boxes 1 and 2 was directed at the identification of linearized state-space CFD aerodynamic models for coupling to linear state-space structural dynamics models of aerospace vehicles and subsequent aeroelastic analysis of the coupled system. Both single input/single output (SISO) and multi-input/multi-output (MIMO) systems were addressed in these studies. However, in all cases involving multiple inputs the kernels were calculated assuming that there was no nonlinear coupling between inputs and outputs. In spite of this simplification, studies to-date indicate that state-space aerodynamic models identified using this assumption can sometimes provide acceptable results, particularly for systems that are not highly nonlinear. The results of some initial exploratory numerical investigations conducted using this approach are reported in references 1-3. Some later and more substantive applications in both computational and experimental aeroelasticity are reported in references 4-7.

To better account for nonlinear aeroelastic phenomena in analytical predictions, there is clearly a need to extend the subject methodology to include more of the nonlinear aerodynamics than is possible by simply using the expedient of linearizing the underlying nonlinear CFD equations about a nonlinear steady-state solution. Such linearized models are oftentimes inadequate, in which case there is a need for an improved approximation having a wider range of applicability in representing the behavior of the nonlinear system. This long-term need was recognized at the time the research program was being defined and led to the decision to include an improved treatment of aerodynamic nonlinearity by using a bilinear state-space representation to approximate the nonlinear aerodynamics (see box 2 of figure 1). Bilinear equations describe a form of nonlinearity in which the system is separately linear in state and input but not jointly, i.e., the nonlinearity appears as a product of state and input. In spite of their special form, bilinear equations have a demonstrated capability to model nonlinear behavior in a rather large variety of engineering and non-engineering disciplines (see, for example, refs. 8-12). It was their established range of applicability and their relative simplicity that prompted the selection of bilinear equations to represent the aerodynamic nonlinearities in the present work. Because of the key role to be played by this bilinear representation, it was clear that the extent to which bilinear state-space models can be used to represent nonlinear systems had to be demonstrated. For this reason, it was decided that all relevant computations associated with any proposed new procedure would be demonstrated using simple nonlinear systems before being applied to problems involving any real CFD calculations.

The bilinear system identification portion of the planned research program (fig. 1, box 2) began in 2002 with the decision to first note what types of relevant formulations were available in the literature and then to identify those that could be either adopted as-is or easily modified to meet the requirements of the present investigation. The primary requirement of any candidate approach was that it be Volterra-kernel based for compatibility with the procedures already established and implemented in the research program. Contending procedures also needed to have adequate verification by an attendant example. Also, because CFD models are discrete, a discrete-time formulation was required.

The authors conducted a survey of the literature dealing with the identification of state-space matrices for bilinear systems that met the selection criteria noted above. Many papers were examined and evaluated (see, for example, refs. 13-19). With one exception, no suitable procedures were found. Most were either SISO formulations and not easily extendable to the multi-input case, or theoretical expositions of MIMO formulations that did not lend themselves to practical implementation. Also, none of the excluded methods were Volterra-kernel based as required by the selection criteria. These papers also lacked examples that demonstrated the key computational steps in their procedures. The most promising approach found was in a 1971 paper by *Bruni, di Pillo,* and *Koch* (ref. 13). Those authors studied the relationship between the output of a bilinear system expressed in state-space form and the output of a nonlinear system expressed as an infinite series of Volterra kernel-based convolution integrals of increasing order. They defined a clear and direct procedure for identifying the state-space matrices of continuous bilinear systems using a subset of kernels through second order appearing in the series expansion. They also provided a suitable example illustrating their procedure for a multi-input/single-output (MISO) case. Although their method defined a continuous-time bilinear system identification process using continuous kernels, the procedure was deemed to have the potential for extension to the discrete case. For these reasons, the decision was made to go forward with an evaluation of the procedure of reference 13 in expectation of its eventual extension to the discrete case[1].

The purpose of this report is twofold: (1) To summarize the analytical, computational, and numerical considerations that underlie both the original continuous-time bilinear realization procedure of *Bruni, di Pillo,* and *Koch* (ref. 13) and its extension to the discrete case of interest herein; and (2) To present results from a suite of numerical calculations that were made to evaluate the new procedure. The presentation begins with a brief overview of bilinear equations. This is followed by a discussion of Volterra series, the kernels that underlie those series, and the role of these kernels in convolution-based solutions of system response to arbitrary inputs. The continuous-time bilinear system identification procedure of *Bruni, di Pillo,* and *Koch* (ref. 13) is summarized in the next section. The strategy used for developing and verifying an extension of their method to discrete problems is then outlined. This is followed by a discussion of how the key computational steps of their method were analytically and numerically confirmed by the present authors, culminating with a reproduction of the numerical results presented in their paper. The pulse loading patterns that are imposed to calculate the discrete Volterra kernels needed in a discrete formulation are then described. Based on the insight gained from performing the aforementioned tasks, a candidate computational procedure emerged for addressing the subject problem of identifying bilinear state-space representations of discrete nonlinear systems, which is described in the next section. There then follows a brief discussion of the manner in which the continuous equations employed in the illustrative examples were discretized for numerical solution. Illustrative results are then presented, first from applications of the new procedure to several nonlinear and bilinear SISO problems from the literature, and then from applications using a modified version of the MISO equations of reference 13. Emphasis in this section is on showing the accuracy of the response time histories computed using the discrete bilinear models obtained using the method introduced in this report. This section also includes some time histories computed using the first- and second-order kernels directly in a convolution-based solution of the original nonlinear equations. Pertinent numerical issues associated with both the original and extended methods are addressed where appropriate throughout the report. The report concludes with the discussion of a block diagram summarizing the sequence of tasks associated with the new computational procedure. The discussion of several relevant aspects of the present development is relegated to the appendices. These include: (1) Formulas for computing first- and second-order Volterra kernels directly if the continuous bilinear state-space matrices are known (Appendix A); (2) A schematic depiction of pulse loading patterns for calculating first- and second-order kernels for discrete systems with two inputs (Appendix B); and (3) A collection of algebraic-like rules for imposing the pulse loading patterns employed for computing the subsidiary responses for calculating the kernels for the general MIMO case (Appendix C).

Bilinear Equations

As mentioned earlier, bilinear equations are a class of nonlinear equations that have been used successfully in the mathematical modeling of nonlinear phenomena and processes in a wide

[1] While this task was underway, a procedure for continuous-time bilinear system identification based on using a set of responses to a series of varying-length step inputs appeared in the literature (ref. 20). This method may offer a practical, non-kernel-based computational approach that could serve as an alternative to the method of reference 13.

variety of disciplines. Bilinear equations are ordinary differential equations that are separately linear in state and input but not jointly. That is, the nonlinearity involves a product of state and input. Thus, the time-invariant state-space equations for a continuous bilinear system may be written as

$$\begin{aligned}
\underset{n\times 1}{\dot{X}_c(t)} &= \underset{(n\times n)}{A_c}\,\underset{(n\times 1)}{X_c(t)} + \sum_{j=1}^{r} \underset{(n\times n)}{N_{c\,j}}\,\underset{(n\times 1)}{X_c(t)}\,\underset{(1\times 1)}{u_{c\,j}(t)} + \underset{(n\times r)}{B_c}\,\underset{(r\times 1)}{u_c(t)} \\
\underset{m\times 1}{y_c(t)} &= \underset{(m\times n)}{C_c}\,\underset{(n\times 1)}{X_c(t)} + \underset{(m\times r)}{D_c}\,\underset{(r\times 1)}{u_c(t)}
\end{aligned} \tag{1}$$

where X_c is the $n\times 1$ state vector, u_{cj} is the j^{th} component of the $r\times 1$ input vector u_c, y_c is the $m\times 1$ output vector, and A_c, B_c, C_c, D_c, N_{cj} are real constant matrices of appropriate dimensions. The corresponding discrete-time equations may be written as

$$\begin{aligned}
X(k+1) &= A_d\; X(k) \; + \sum_{j=1}^{r} N_{d\,j}\, X(k)\, u_j(k) + B_d\; u(k) \\
y(k) &= C_d\, X(k) \; + \; D_d\; u(k)
\end{aligned} \tag{2}$$

where the interpretation and sizes of the matrices A_d, B_d, C_d, D_d, and N_{dj} and the vectors X, u, and y are the same as the corresponding quantities in equations 1. In the discrete form, time has been discretized such that $t = k\,\Delta t$, where k is a discrete time index and Δt is a discrete time increment omitted from the equation to simplify notation.

Bilinear systems represent a rather simple class of nonlinear systems. In spite of their simplicity, however, bilinear mathematical models have been (and continue to be) used to describe a wide variety of nonlinear phenomena and processes in such fields as engineering, physics, chemistry, biology, medicine, ecology, agriculture, and economics (see, for example, refs. 8-12). The control of distillation processes in the chemical industry, for instance, depends to a large extent on the bilinear equations resulting from approximations made to the nonlinear equations governing those processes. Bilinear equations can arise naturally during the course of deriving governing equations of motion for a system or process that is inherently bilinear, as a consequence of an a priori approximation made to a nonlinear system, or as a result of applying some type of formal “linearization” technique such as Carleman linearization (ref. 21) to a nonlinear system of equations. The reader is referred to references 8-12, 15, and 22-23 for discussions dealing with the mathematical aspects of bilinear systems and the nuances of their behavior.

Because of the ubiquitous nature of bilinear equations, a rather sizable literature has arisen that deals with the realization (i.e., identification) of the state-space matrices appearing in equations 1 and 2. However, while techniques for identifying the state-space matrices *ABCD* for both discrete and continuous linear systems are well established and a number of reputable and proven methods are available, the same level of maturity has not yet emerged for bilinear systems. In an attempt to address this need as it applies to the reduced-order modeling of unsteady aerodynamics in CFD-based aeroelastic analyses, this report presents and evaluates a computational procedure for identifying the state-space matrices for bilinear representations of nonlinear discrete-time systems. Because of the key role played by Volterra kernels in the subject development, some introductory remarks dealing with such kernels are given in the next section.

Volterra Series, Volterra Kernels, and Convolutions

The response of a causal, time-invariant, finite-memory, nonlinear system to an arbitrary input may be expressed as an infinite sum of multidimensional convolution integrals of increasing order. Representation of nonlinear systems in this manner is treated extensively in the literature (see, for example, references 24-28). For continuous-time MIMO systems, such series can be written as

$$\begin{aligned} y(t) = y_o + \int_0^t h_1(\tau_1)\, u(t-\tau_1)\, d\tau + \int_0^t\int_0^t h_2(\tau_1,\tau_2)\, u(t-\tau_1)\, u(t-\tau_2)\, d\tau_1 d\tau_2 \\ + \sum_{i=3}^{\infty}\int_0^t \cdots \int_0^t h_i(\tau_1,\cdots,\tau_i) u(t-\tau_1)\cdots u(t-\tau_i)\, d\tau_1 \cdots d\tau_i \end{aligned} \tag{3}$$

where *y(t)* is an $m \times 1$ response vector, $u(t)$ is an $r \times 1$ input vector, $h_1(\tau_1)$ is a first-order kernel, $h_2(\tau_1,\tau_2)$ is a second-order kernel, etc. Any nonzero steady-state contribution to the total response is represented by the $m \times 1$ constant vector y_o. The value of y_o is known based on the steady-state value of the system at a particular condition. Representations of the responses of nonlinear systems having the form given in equation 3 are called Volterra series and the kernels appearing in them are called Volterra kernels. The forms of the higher-order kernels appearing in such series are not unique[2] and different forms appear in the literature depending on the domain of integration chosen for their definition. Both symmetric and non-symmetric (e.g., triangular) kernels are common in the literature. However, because non-symmetric kernels can be easily converted to symmetrical kernels without affecting the input/output relation, there is no loss of generality in assuming symmetric kernels in a formulation (refs. 24-25). The form of the Volterra series given in equation 3 assumes that the higher-order kernels are symmetric.

The Volterra series representation of nonlinear systems as given in equation 3 may be viewed as an extension of the method of linear convolution to nonlinear systems. The first

[2] Volterra kernels are, however, invariant characteristics of a nonlinear system in the same sense that system Markov parameters uniquely characterize a linear system.

integral appearing in equation 3 is the linear convolution integral that gives the linear contribution to the total response. The second integral in the series may be regarded as a second-order convolution integral that provides the second-order contribution to the total response. Similar interpretations may be made for the successively higher-order integrals in the series. The first-order kernel $h_1(\tau_1)$ characterizes the linear behavior of the system and is identical to the unit impulse response of the linear (or linearized) system. The second-order kernel $h_2(\tau_1, \tau_2)$ characterizes the behavior of the nonlinear system to two separate unit impulses applied at two varying points in time. Thus, the kernel $h_2(\tau_1, \tau_2)$ is a two-dimensional function. It is a function of time τ_1 and a function of time lag $T = \tau_2 - \tau_1$ so that for every T that is used, a new function of time τ_1 is defined. These functions of time are referred to as components of the second-order kernel. Analogous interpretations may be made for the higher-order kernels. Consequently, the kernels appearing in the sequence of higher-order convolution integrals account in an increasing manner for the nonlinearity in the system. The higher-order kernels are also a function of the amplitude of the impulse used for identification (ref. 3).

Volterra theory, as exemplified by equation 3, states that any time-invariant nonlinear system with finite memory can be modeled as an infinite sum of multi-dimensional convolution integrals. The assumption of time invariance means that the equations describing the system are not explicit functions of time. The assumption of finite or fading memory implies that the kernels are decaying functions of time and decay to zero in a finite amount of time. Equation 3 indicates that an infinite number of convolution terms, and thus kernels, are formally required for such representations. This suggests that Volterra series models may be impractical for highly nonlinear problems, with convergence being either slow or nonexistent. However, any nonlinear system that exhibits fading memory, in the sense that past inputs have a diminishing influence on present outputs, can be approximated to arbitrary accuracy by a truncated Volterra series (ref. 24). Fortunately, many nonlinear systems encountered in engineering satisfy this requirement (i.e., are of the fading-memory type) and may be approximated by series that include only a few of the lower-order integrals, oftentimes, only the first two or three. However, irrespective of the number of terms retained, the radius of convergence of a Volterra series is limited, analogous to the radius of convergence of the Taylor series expansion of a function. Also, the truncation error associated with using a finite Volterra series approximation grows with input amplitude. Thus, Volterra series representations of nonlinear systems converge on bounded time intervals and for bounded inputs (refs. 24-25). The actual limits in any given case depend primarily on the type of nonlinearity, the strength of the nonlinearity, and the magnitude and type of input. In spite of these restrictions, Volterra series have been and continue to be used successfully to model nonlinear processes and phenomena in a variety of fields, as pointed out earlier. It is only rather recently, however, that discrete Volterra kernels have been promoted and established as a practical computational tool for reduced-order modeling of unsteady aerodynamics in CFD-based aeroelastic analyses (refs. 1-3).

The discrete-time version of the Volterra series given in equation 3 may be written as

$$
\begin{aligned}
y(k) = y_o + \sum_{s_1=1}^{N} h_1(s_1)u(k-s_1) + \sum_{s_1=1}^{N}\sum_{s_2=1}^{N} h_2(s_1,s_2)u(k-s_1,k-s_2) \\
+ \sum_{i=3}^{\infty}\sum_{s_1=1}^{N}\cdots\sum_{s_i=1}^{N} h_i(s_1,\cdots,s_i)u(k-s_1)\cdots u(k-s_i)\,ds_1\cdots ds_i
\end{aligned}
\tag{4}
$$

where N is the number of points in the discretized time history, and the interpretation of the other quantities follows from a comparison with the comparable quantities in equation 3 for the continuous case. The reader is referred to references 3, 12, 22, 24-25, and 27 for discussions dealing with the theoretical aspects of Volterra series representation of nonlinear systems and the calculation of relevant kernels.

Bruni, di Pillo, and Koch (ref. 13) studied the relationship between the models represented by equations 1 and 3 and developed the continuous-time bilinear system identification (SID) procedure that was selected as the basis for devising a procedure addressing the broader problem of identifying bilinear state-space representations of discrete-time nonlinear systems which is of interest here. Their procedure is summarized in the next section.

Procedure of Bruni, di Pillo, and Koch for Identification of Continuous Bilinear Systems

The procedure introduced by *Bruni, di Pillo, and Koch* (ref. 13) for the identification of the state-space matrices of a continuous bilinear system using Volterra kernels is summarized in this section. Only the major steps in their procedure are presented here, with minimum discussion. The reader is referred to their paper for a thorough discussion of the relevant theoretical aspects underlying the development, including theorems and assumptions.

Reference 13 starts with the time-invariant state-space equations of a continuous bilinear system in the form

$$
\begin{aligned}
\dot{X}_c &= A_c X_c + B_c u + \sum_{j=1}^{r} u_j N_{cj} X_c \\
y &= C_c X_c + D_c u
\end{aligned}
\tag{5}
$$

where u_j is the j-th component of the $r \times 1$ input vector u, X_c is the $n \times 1$ vector of state variables, y is the $m \times 1$ vector of outputs, and A_c, B_c, C_c, D_c, and N_{cj} are matrices of suitable dimensions. The output of a nonlinear, time-invariant system may be related to its input by a functional series of kernels (assumed to be known) which, for MIMO systems, may be written as

$$y(t)=\sum_{i=1}^{\infty}\underbrace{\int_0^t\int_0^t\cdots\int_0^t}_{i}\sum_{j_1=1}^{r}\sum_{j_2=1}^{r}\cdots\sum_{j_i=1}^{r}w_i^{(j_1,j_2,\ldots,j_i)}(\tau_1,\tau_2,\ldots,\tau_i)\bullet$$

$$u_{j_1}(t-\tau_1)u_{j_2}(t-\tau_2)\cdots u_{j_i}(t-\tau_i)d\tau_1,d\tau_2,\ldots,d\tau_i \tag{6}$$

The kernels w_i appearing in equation 6 are triangular kernels so that $\tau_i \geq \tau_{i-1} \geq \ldots \geq \tau_2 \geq \tau_1 \geq 0$.

As mentioned earlier, systems can often be treated as finite-memory systems, in which case the first summation appearing in equation 6 may be approximated by a finite summation. The index i denotes the order of the kernels and r denotes the number of inputs. The expansion of equation 6 is characterized for every order i by r^i kernels $w_i^{j_1,j_2,\ldots,j_i}(\tau_1,\tau_2,\ldots,\tau_i)$. For example, for the case of 2 inputs (r = 2), there are 2^1 = 2 first-order kernels and 2^2 = 4 second-order kernels. While investigating the relationship between the models given by equations 5 and 6, *Bruni, di Pillo, and Koch* were led to a procedure for identifying the bilinear state-space matrices in equation 5 using the first- and second-order kernels from equation 6. An ancillary result that emerged from their studies was a general equation for computing the kernels which appear in equation 6 given the state-space matrices which define the bilinear model of equation 5. The principal steps in their identification procedure are described briefly below.

The first major step of their procedure is to compute the first-order kernels $w_1^{(j)}$ of the linearized system

$$\begin{aligned}\dot{X}_c &= A_c X_c + B_c u\\ y &= C_c X_c + D_c u\end{aligned} \tag{7}$$

obtained from equation 5 by setting $N_{cj} = 0$. This is a straightforward calculation as these kernels are identical to the impulse responses of a linear system and can be computed in the same way. The collection of r first-order kernels $w_1^{(j)}$ can be assembled into an $m \times r$ matrix

$$\underset{m\times r}{W_1}(\tau_1)=\left[\underset{m\times 1}{w_1^{(1)}(\tau_1)}\;\;\underset{m\times 1}{w_1^{(2)}(\tau_1)}\cdots\underset{m\times 1}{w_1^{(r)}(\tau_1)}\right] \tag{8}$$

which is the impulse response matrix of the linearized system. The use of any realization procedure suitable for time-invariant linear systems applied to the matrix $W_1(\tau_1)$ will yield the state-space matrices A_c, B_c, C_c, and D_c.

The next major step in their procedure is the determination of the matrices N_{cj}. To this end, the paper first defines the $n \times n$ matrices

$$G=\int_{t_1}^{t_2}[C_c e^{A_c\tau}]^T C_c e^{A_c\tau}d\tau \tag{9a}$$

$$H = \int_{t_1}^{t_2} e^{A_c \tau} B_c \, [e^{A_c \tau} B_c]^T \, d\tau \tag{9b}$$

The matrices G and H are computed using the matrices A_c, B_c, and C_c obtained above and will be nonsingular for a controllable and observable system. The setting of the time limits of integration $[t_1, t_2]$ is discussed in a later section.

The '$m \times 1$' second-order kernels $w_2^{(j_1, j_2)}$ are collected into the r '$m \times r$' matrices

$$\underset{m \times r}{W_2^j(\tau_1, \tau_2)} = \Big[\underset{m \times 1}{w_2^{(j,1)}(\tau_1, \tau_2)} \quad \underset{m \times 1}{w_2^{(j,2)}(\tau_1, \tau_2)} \cdots \underset{m \times 1}{w_2^{(j,r)}(\tau_1, \tau_2)} \Big] \quad , \quad j = 1, 2, ..., r \tag{10}$$

Note that although each of the second-order kernels is ostensibly $m \times 1$ in size, they are actually each composed of a series of $m \times 1$ components whose number is determined by the number of time lags T used to evaluate the kernels, as discussed earlier. Thus, each kernel must actually be regarded as an $m \times ncomp$ matrix for computational purposes, where *ncomp* is the number of components (and thus time lags T) used to define the kernel. These second-order kernels and the matrices A_c, B_c, C_c determined above are then used to compute the $n \times n$ matrices K_j defined by

$$K_j = \int_{t_1}^{t_2} \int_{t_1+\tau_1}^{t_2+\tau_1} [C_c e^{A_c \tau_1}]^T \, W_2^j(\tau_1, \tau_2) \, [e^{A \, (\tau_2 - \tau_1)} B_c]^T \, d\tau_2 \, d\tau_1 \quad , \quad j = 1, 2, \ldots r \tag{11}$$

The desired $n \times n$ matrices N_{cj} are then given by

$$N_{cj} = G^{-1} K_j H^{-1} \quad , \qquad j = 1, 2, \ldots, r \tag{12}$$

This completes the identification of the state-space matrices A_c, B_c, C_c, D_c, and N_{cj} defining the continuous bilinear system given in equation 5.

Strategy for Developing and Verifying Extension to Discrete Systems

A strategy for extending the continuous-time method of reference 13 to discrete systems that are of interest in this report was defined by the present authors and is summarized here. Clearly, the first task was to perform a careful review of the paper to understand the key mathematical operations that had to be performed. The next task was to verify the analytical and numerical results for the illustrative example given in the paper. This would allow intermediate numerical results associated with the various stages of analysis using the bilinear equations of the example to serve as a basis for evaluating numerical results obtained at similar stages of analysis while developing the new procedure. This task would also provide experience in the application of relevant numerical integration techniques as well as identify any numerical issues that might be associated with such computations. The next major step was to devise the pulse loading patterns needed for computing the discrete second-order off-diagonal kernels that arise in the case of multi-inputs to complement the known patterns needed for computing the on-diagonal second-order kernels. A subtask of this step was to verify the first- and second-order discrete kernels computed using these pulses by comparing them to the discretized versions of the continuous kernels for the example problem in reference 13. The next task was to define a candidate procedure for extending their method to the more general problem of identifying bilinear state-space representations of nonlinear discrete-time systems while retaining as much of the already-established procedure of reference 13 as possible. The final step was to evaluate the candidate procedure by applying it to a variety of nonlinear equations and comparing response time histories obtained using the identified bilinear model with results obtained from direct numerical solution of the governing nonlinear equations. These tasks are discussed in the remaining sections of this report.

Confirmation of Bruni, di Pillo, and Koch Results

The first major task of the verification process was to carefully review the analytical and computational bases for the method of bilinear system identification presented by *Bruni, di Pillo, and Koch* in reference 13. Once the method was understood, the algebraic manipulations that are implied by equations 9, 11, and 12 above were carried out both analytically and numerically using their example to reproduce the results shown in the paper. This critical evaluation provided the insight (and confidence) needed to proceed with the subject development. The key aspects of this evaluation are summarized below.

Example Problem

The example used by *Bruni, di Pillo, and Koch* (ref. 13) to illustrate the application of their method considers the following continuous-time bilinear system, with $n = 2$, $r = 2$, and $m = 1$:

$$\begin{aligned} \dot{x}_1 &= -x_1 + x_1 u_2 + x_2 u_2 + u_1 \\ \dot{x}_2 &= x_1 - 2x_2 + x_1 u_1 + x_2 u_1 + u_2 \\ y &= x_2 \end{aligned} \tag{13}$$

The corresponding first-order state-space matrix equations are

$$\dot{X} = A_c X + \sum_{j=1}^{r} N_{cj} X u_j + B_c u \tag{14a}$$

$$y = C_c X_c + D_c u \tag{14b}$$

where the state matrix A_c, force matrix B_c, output matrix C_c, direct transmission matrix D_c, bilinear matrices N_{c1} and N_{c2}, force input vector u, and state vector X are given by

$$A_c = \begin{bmatrix} -1 & 0 \\ 1 & -2 \end{bmatrix}; \quad B_c = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; \quad C_c = \begin{bmatrix} 0 & 1 \end{bmatrix}; \quad D_c = \begin{bmatrix} 0 & 0 \end{bmatrix};$$
$$N_{c1} = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}; \quad N_{c2} = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}; \quad X = \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix}; \quad u = \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} \qquad (15)$$

Volterra Kernels

Substituting A_c, B_c, C_c, N_{c1}, and N_{c2} given above into equations A1 and A2 from Appendix A, noting that

$$e^{A_c t} = \begin{bmatrix} e^{-t} & 0 \\ e^{-t} - e^{-2t} & e^{-2t} \end{bmatrix} \qquad (16)$$

and then carrying out the operations indicated in those equations leads to

$$\begin{aligned} w_1^{(1)}(\tau_1) &= e^{-\tau_1}(1 - e^{-\tau_1})\delta_{-1}(\tau_1) \\ w_1^{(2)}(\tau_1) &= e^{-2\tau_1}\,\delta_{-1}(\tau_1) \end{aligned} \qquad (17)$$

and

$$\begin{aligned} w_2^{(1,1)}(\tau_1,\tau_2) &= (2e^{-\tau_1-\tau_2} - e^{-2\tau_2})\delta_{-1}(\tau_1)\delta_{-1}(\tau_2 - \tau_1) \\ w_2^{(1,2)}(\tau_1,\tau_2) &= e^{-2\tau_2}\delta_{-1}(\tau_1)\,\delta_{-1}(\tau_2 - \tau_1) \\ w_2^{(2,1)}(\tau_1,\tau_2) &= (2e^{-\tau_2} + e^{-2\tau_2} - 2e^{-\tau_1-\tau_2} - e^{\tau_1-2\tau_2})\delta_{-1}(\tau_1)\delta_{-1}(\tau_2 - \tau_1) \\ w_2^{(2,2)}(\tau_1,\tau_2) &= (e^{\tau_1-2\tau_2} - e^{-2\tau_2})\delta_{-1}(\tau_1)\delta_{-1}(\tau_2 - \tau_1) \end{aligned} \qquad (18)$$

as the expressions for the first- and second-order Volterra kernels, respectively. The requirement $\tau_2 \geq \tau_1 \geq 0$ that is imposed in reference 13 implies that the second-order kernels are triangular (rather than symmetric). That is, they are nonzero only in the domain of space defined by $\tau_2 \geq \tau_1 \geq 0$. The terms $\delta_{-1}(\tau_1)$ and $\delta_{-1}(\tau_2 - \tau_1)$ are unit step functions that were introduced in reference 13 for notational convenience to allow some upper limits of integration to be extended to the same value of time. In practice, such step functions are usually not shown but are assumed to be part of the kernels for brevity of notation.

The expressions given in equations 17 and 18 were verified following the steps noted above. Then, substituting A_c, B_c, and C_c from equation 15 into equation 9, using equation 16, and integrating analytically over the time interval [t_1, t_2] = [0, 1] as in reference 13 resulted in the *G* and *H* matrices shown in the paper.

The K_j are calculated using equation 11, which involves the evaluation of a double integral. Substituting A_c, B_c, and C_c from equation 15 and the second-order kernels from equation 18 into equation 11, using equation 16, and integrating the resulting expression analytically first over $d\tau_2$ from τ_1 to $1+\tau_1$ and then over $d\tau_1$ from 0 to 1 as in reference 13 yield the matrices K_1 and K_2 shown in the paper. This integration is facilitated if a change of variable is made from (τ_1,τ_2) to (τ_1,T) according to $(\tau_2=\tau_1+T)$, where *T* represents the varying time lag used to compute the components of the second-order kernels. The integration limits for *T* become [t_1, t_2], the same as for τ_1, as indicated in equation 19 below:

$$K_j=\int_{t_1}^{t_2}\int_{t_1+\tau_1}^{t_2+\tau_1}[\,]\,d\tau_2 d\tau_1 \Rightarrow \int_{t_1}^{t_2}\int_{t_1}^{t_2}[\,]\,dT d\tau_1 \tag{19}$$

Introducing this change of variable in equations 17 and 18, the expressions for the first- and second-order Volterra kernels take the form

$$\begin{aligned} w_1^{(1)}(t)&=e^{-t}(1-e^{-t})\delta_{-1}(t)\\ w_1^{(2)}(t)&=e^{-2t}\,\delta_{-1}(t) \end{aligned} \tag{20}$$

and

$$\begin{aligned} w_2^{(1,1)}(t,T)&=(2e^{-2t-T}-e^{-2(t+T)})\delta_{-1}(t)\delta_{-1}(T)\\ w_2^{(1,2)}(t,T)&=e^{-2(t+T)}\delta_{-1}(t)\,\delta_{-1}(T)\\ w_2^{(2,1)}(t,T)&=(2e^{-(t+T)}+e^{-2(t+T)}-2e^{-2t-T}-e^{-t-2T})\delta_{-1}(t)\delta_{-1}(T)\\ w_2^{(2,2)}(t,T)&=(e^{-t-2T}-e^{-2(t+T)})\delta_{-1}(t)\delta_{-1}(T) \end{aligned} \tag{21}$$

The limits of integration on the variables τ_2 or *T* inherently contain the intended effect of the step functions (i.e., imposing the requirement $\tau_2 \ge \tau_1$ or $T \ge 0$) and there is no need for explicit consideration of the step functions in equations 18 or 21 in either the analytical or numerical evaluation of the K_j integrals. Therefore, all components of the second-order kernels can be plotted as though starting from time zero for purposes of evaluating the integrals K_j in equation 19. The shifting of the components of the second-order kernels in this manner, in combination with the change of variable noted above, was found to simplify the coding associated with evaluation of the K_j integrals and so was adopted for use in the present work.

The defining integrals for G, H, and K_j given in equations 9 and 11 were also evaluated numerically in anticipation of that requirement in the discrete version of the method. Several combinations of values for the continuous time increments dt and dT were used in the numerical integrations and the results compared to assess the sensitivity of the results to the values employed. For example, using $dt = 0.00001$ sec, $dT = 0.0001$ sec, and $ncomp = 10000$ produced results that were in exact agreement with the analytical results shown in the paper. G and H were found to be much less sensitive to the values used than the K_j , with values of dt as large as 0.01 giving results that are within 2% of the analytical values of G and H. Although the K_j were more sensitive to the values used for dt and dT, it was found that the resulting values for N_{cj} obtained from equation 12 were not unduly sensitive to the values of dt and dT as long as the behavior of the influential portions of the kernels were adequately included in the numerical evaluation of the integrals. For example, good results were obtained using the much coarser values $dt = 0.01$, $dT = 0.1$, and $ncomp = 10$. These results suggest that acceptable results for the computed system response time histories that are ultimately of interest can be obtained using a coarser (and therefore more computationally efficient) set of time values in the numerical evaluation of those integrals.

The continuous first- and second-order kernels defined by equations 20 and 21 are shown in figures 2 and 3, respectively, where they are plotted versus time index k over a time interval representing three seconds using $dt = 0.01$ sec, $dT = 0.1$ sec, and $ncomp = 10$ to evaluate those equations. The 10 components of each second-order kernel that are shown are all plotted as though originating at time $t = 0$. However, as mentioned earlier, the presence of the unit step functions in the defining expressions (equations 18 or 21) implies that the components are each actually shifted in time to the right. This shift is by an amount that is an increasing integer multiple of dT according to $T = idT$, for $i = 0, 1, 2, \ldots, ncomp\text{-}1$. Note that the pulse response functions have a finite or fading memory in that they all decay rather quickly to essentially zero.

It should be noted that the time index k is used on the time axes in figures 2 and 3 rather than the actual time t. While either variable can be used in plotting time histories because they are related according to $t = k\,\Delta t$, $k = 0,1,2,\ldots,N\text{-}1$, time index is used throughout this report.

Conversion of Continuous Kernels to Discrete Kernels

The first- and second-order kernels given by equations 20 and 21 (and plotted in figures 2 and 3) are continuous kernels because they were obtained from equations A1 and A2 using the continuous bilinear state-space matrices given in equation 15. Also, recall that the second-order kernels w_2 are triangular kernels as they are defined for $\tau_1 \le \tau_2$ (i.e., $T \ge 0$). The present adaptation of the method of reference 13 to address the title problem is based on using the extension of an existing discrete kernel calculation procedure that produces symmetric second-order kernels (refs. 1, 3, 24). To enable direct comparisons of the discrete kernels obtained using that procedure with the continuous kernels of reference 13, one needs to discretize the first-order continuous kernels and both discretize and symmetrize the continuous second-order kernels. This conversion is made by multiplying the first-order continuous kernels by the time step Δt and the continuous second-order kernels by $(\Delta t)^2/2$. The result of applying this continuous-to-discrete conversion to the continuous kernels in figures 2 and 3 leads to the discrete kernels shown in figures 4 and 5.

Some Relevant Observations and Comments

A number of observations were made while making the calculations associated with the studies noted above. Several of those pertinent to computational issues are noted below:

(1) The lower limit of time integration t_1 is always 0. However, the upper limit of integration t_2 need not be 1 as used in the numerical example of reference 13. The upper limit must be chosen so that it includes all the significant contributory portions of the relevant Volterra kernels. To this end, one needs to inspect the computed components of the second-order kernels to identify their upper range of influence (i.e., the time interval over which their contribution to the integrals defining K_j is not negligible) and then set the upper limit of integration accordingly.

(2) Introduction of the time lag T in the double integrals for K_j via a change in variable allows for the direct interpretation of the components comprising the kernels as time-lagged quantities and facilitates computer coding. This was done in the present implementation.

(3) The numerical evaluation of the K_j integrals is facilitated considerably if all the components of the second-order kernels are shifted to the time origin as was done in reference 13. This approach was also employed in the present implementation.

(4) Numerical evaluation of the integrals defining G, H, and K_j is straightforward and provides satisfactory results for the N_j if the time increments used in the integration are small enough to capture the time-varying behavior of the kernels over their time range of influence.

(5) Continuous kernels are discretized by multiplying the first-order kernels by the time step Δt and the second-order kernels by $(\Delta t)^2$.

(6) Triangular second-order kernels (as assumed in reference 13) are made symmetric by dividing them by two.

Pulse Loading Patterns for Calculating Discrete Volterra Kernels

As stated in the Introduction, the objective of the present investigation is to develop a computational procedure for using the first- and second-order Volterra kernels of a discrete nonlinear system to identify the state-space matrices of an approximate discrete bilinear-equation representation of the nonlinear system. Thus, discrete-time, first- and second-order Volterra kernels are central to the present formulation. The calculation of the first-order kernel for a linear system is straightforward. One simply imposes a unit-amplitude pulse of (small) duration Δt at $t = 0$ for each input of interest and computes the corresponding response time histories for all outputs of interest. Functions carrying out this type of calculation for linear equations are, for example, included in the Control System Toolbox of MATLAB (ref. 29). However, here, following references 1 and 3, the relation

$$k_1^{(p)} = 2y_1 - 0.5y_{11} \quad , \qquad p = 1,2,...,r \tag{22}$$

is used to compute the first-order kernels for each input, where y_1 is the response of the nonlinear system to a single unit pulse applied at $t = 0$ and y_{11} is the response of the nonlinear system to two unit pulses applied at $t = 0$. The calculation is done in this manner so that the computed response k_1 is the linear portion of the nonlinear response (i.e., linearized solution about a nonlinear steady-state condition) rather than the purely linear response (i.e., linear response computed using linear equations). If k_1 is computed using equation 22, some of the amplitude dependence of the nonlinear system is accounted for in that kernel.

The calculation of the higher-order kernels is more involved as they are the responses of the nonlinear system to multiple unit pulses, with the number of pulses applied to the system equal to the order of the kernel of interest. Thus, the second-order kernel, k_2, is the response of the system to two separate unit pulses applied at two points in time and is therefore a two-dimensional function of time. The variation of the time difference (i.e., time lag) between the two pulse inputs yields the components of the kernel and allows for the characterization of the second-order memory of the system. Following references 1 and 3, the second-order kernels are computed here using the relation

$$k_2^{(p,q)} = 0.5\,(y_{12} - y_1 - y_2) \quad , \qquad p,q = 1,2,...,r \tag{23}$$

where y_1, y_2, and y_{12} are subsidiary pulse responses corresponding to three separate pulse loading patterns imposed appropriately in time at the inputs of interest. Specifically, y_1 is the response of the system to a unit pulse applied at time t, y_2 is the response of the system to a unit pulse applied at time $(t + T)$, and y_{12} is the response of the system to two unit pulses, one applied at time t and the other at time $t + T$. It should be noted that for time-invariant systems, the y_2 responses can be obtained by simply shifting the y_1 response appropriately in time, thereby effectively requiring the calculation of only 2 subsidiary responses. In all the calculations, the times are chosen to satisfy the relation $t \geq T$, resulting in kernels that are symmetric about $t = T$. The three responses are substituted into equation 23 to yield one component of the second-order kernel. The time lag T is then varied to obtain additional components of the second-order kernel. There are r^2 kernels, $k_2^{(p,q)}$; each kernel has *ncomp* components; each component is an $m\times1$ vector. The first component of k_2 is defined when $T = 0$, i.e., when both pulses are applied at the same point in time. The last computed component of k_2 is defined for $T = (ncomp–1)\,\Delta T$. The number of components needed to accurately define a second-order kernel depends on the nonlinear system under investigation. In general, the greater the nonlinearity, the greater the number of components required. A discussion of the use of pulse inputs for computing symmetric second-order kernels for cases involving a single input or the on-diagonal ($p = q$) second-order kernels for cases involving multiple inputs is given in references 1 and 3. The discrete kernels given by equations 22 and 23 are related to the continuous triangular kernels of reference 13 according to

$$k_1^{(p)} = w_{1d}^{(p)} = w_{1c}^{(p)}\,\Delta t \tag{24a}$$

$$k_2{}^{(p,q)} = w_{2d}{}^{(p,q)} = \frac{1}{2} w_{2c}{}^{(p,q)} (\Delta t)^2 \tag{24b}$$

The complete treatment of the multi-input case, including the calculation of the off-diagonal ($p \neq q$) kernels that account for the full set of input/output interactions associated with the MIMO case, has apparently not been addressed for time problems in the literature and had to be defined for the present study. Based on insight gained from ancillary studies as part of this investigation, the authors proposed pulse-loading patterns that were deemed to be appropriate for computing the off-diagonal kernels in the case of two inputs. These patterns are shown in Appendix B, where they are included in the schematic depiction of the pulse loading patterns needed for computing the full set of first- and second-order kernels for the special case of two inputs. The necessary calculations were coded in MATLAB and provided the results shown in figures 6 and 7. Figure 8 shows the components of the second-order kernel depicted in figure 7 shifted to time index zero. The results of figures 6 and 8 are in excellent agreement with the discretized results of reference 13 as given in figures 4 and 5 and established the credibility of the proposed procedure for calculating off-diagonal kernels.

A pulse-loading process for extending the complete kernel calculation procedure for two inputs to the general case of an arbitrary number of inputs was subsequently devised based on the insight gained in the studies for the case of two inputs. The "algorithm" describing that process is given in Appendix C.

Proposed Computational Procedure

During the course of the work described above, there slowly emerged a notional model for a candidate computational procedure that addresses the broader problem of identifying bilinear state-space representations of discrete nonlinear systems using Volterra kernels that is of interest here. In particular, it became evident quite early that as much of the continuous process of *Bruni, di Pillo, and Koch* (ref. 13) as possible should be used as the basis for extending their procedure to the discrete case. In particular, direct use should be made of their procedure to identify the state-space matrices for continuous bilinear systems as given in equation 14. This expedient clearly portended a blend of continuous and discrete numerical operations and suggested that recourse be made to appropriate MATLAB functions where needed to effect conversions between continuous and discrete linear state-space matrices. Once this key decision was made, the desired extension to the discrete case was greatly facilitated and the remaining steps needed to complete the extension to the discrete case were forthcoming.

Assuming that the nonlinear ODE or PDE of the system of interest have been discretized, the primary steps envisioned for the candidate computational procedure were as follows: (1) Compute the first- and second-order discrete kernels of the discretized nonlinear system. (2) Use the computed first-order kernels in a linear discrete SID algorithm such as ERA (ref. 30) to identify the discrete state-space matrices A_d, B_d, C_d, D_d. (3) Transform the discrete-time matrices A_d and B_d to continuous-time matrices A_c and B_c using the MATLAB function *d2c.m*. No conversion of C_d and D_d is required as the C and D matrices of the continuous model are the same as those for the discrete model (i.e., (C_c, D_c) = (C_d, D_d)) (ref. 30). (4) Transform the first- and second-order discrete kernels into continuous kernels by multiplying them by $1/\Delta t$ and $2/(\Delta t)^2$, respectively. (5) Numerically evaluate the integrals defining G, H, and K_j (eqs. 9 and 11) and then compute N_{cj} (eq. 12).

The matrices A_c, B_c, C_c, D_c, and N_{cj} so determined are the state-space matrices for an approximate continuous bilinear representation of an underlying discrete nonlinear system. With due regard to the ultimate objective of identifying the state-space matrices corresponding to discrete bilinear representations of nonlinear discrete-time systems, the continuous bilinear equations represented by A_c, B_c, C_c, D_c, and N_{cj} must be discretized. Some comments relevant to this aspect of the process are given in the next section.

Discretization and Numerical Solution of Continuous Bilinear Equations

As noted above, the matrices A_c, B_c, C_c, D_c, and N_{cj} define an approximate, continuous-time bilinear model of an underlying nonlinear discrete-time system. This model must be discretized to obtain the desired discrete-time bilinear state-space representation. This can be done, for example, by using one of the Runge-Kutta methods of solution, all of which are self-starting (one-step) methods. The Euler method is a classic example of this approach. If more accuracy is needed, recourse can be made to a non-self-starting (multi-step) predictor/corrector procedure, where the set of needed starting values is obtained by applying a one-step method near the starting point. Discussions of the nuances associated with the application of such techniques may be found in references 31-35, for example. All of these methods approximate the derivatives in the differential equations by appropriate finite-difference expressions involving the dependent and independent variables in the equations.

Euler's method is a first-order Runge-Kutta method and is the simplest one-step method available. However, it is regarded to be of limited value for practical calculations because of its degraded accuracy for large step sizes. Second-order Runge-Kutta algorithms include two formulations that are called improved Euler and modified Euler methods, respectively (ref. 31). The fourth-order Runge-Kutta method is probably the most popular one-step method and is used in the MATLAB ODE solver *ode45*. In spite of the simplicity of Euler's method, it was found to be applicable to most of the numerical calculations that were made as part of the illustrative examples used in this report. In those few cases where it was judged unsatisfactory, recourse was made either to the improved Euler method or to the use of the MATLAB function *c2d* to discretize the bilinear equations in a manner to be described below. When appropriate, the solver *ode45* was used to obtain and compare the response time histories of the original continuous nonlinear equation and its identified continuous bilinear state-space equation representation. Considerable use was also made of the MATLAB functions *lsim*, *dlsim, dimpulse,* and *impulse* as part of the many numerical checks that were made during development of the MATLAB scripts employed for the illustrative examples.

Some Alternative Approaches for Discretizing Bilinear State-Space Equations

As mentioned above, discretization of the continuous-time original and identified bilinear equations used in the Illustrative Results section relied on the use of low-order Runge-Kutta formulas (Euler and improved Euler) and the MATLAB function *c2d*, for convenience. The resulting forms of these equations for the case of two inputs are summarized here.

The Euler forward-difference discretization of equation 14a results in the equation

$$\frac{X(k+1)-X(k)}{\Delta t} = A_c X(k) + B_c u(k) + \sum_{j=1}^{r} u_j(k) N_{cj} X(k) \tag{25}$$

which can be rewritten as

$$X(k+1) = [I + A_c \Delta t] X(k) + B_c \Delta t\, u(k) + \sum_{j=1}^{r} u_j(k) N_{cj} \Delta t\, X(k)$$

or finally

$$X(k+1) = A_d X(k) + B_d\, u(k) + \sum_{j=1}^{r} u_j(k)\, N_{dj}\, X(k) \tag{26}$$

The algebraic definitions of the discrete matrices appearing in equation 26, and their numerical values for $\Delta t = 0.01$, are as follows:

$$\begin{aligned} A_d &= I + A_c\,\Delta t = \begin{bmatrix} 0.99 & 0 \\ 0.01 & 0.98 \end{bmatrix} & \qquad B_d &= B_c \Delta t = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix} \\ N_{d1} &= N_{c1} \Delta t = \begin{bmatrix} 0 & 0 \\ 0.01 & 0.01 \end{bmatrix} & \qquad N_{d2} &= N_{c2} \Delta t = \begin{bmatrix} 0.01 & 0.01 \\ 0 & 0 \end{bmatrix} \end{aligned} \tag{27}$$

If the improved Euler method is applied to equation 14a as described in reference 26, the discrete matrices given in equation 27 are replaced by

$$
\begin{aligned}
A_d &= I + \Delta t\left[3\Delta t A_c A_c + 6A_c\right]/6 = \begin{bmatrix} 0.99005 & 0 \\ 0.0098500 & 0.98020 \end{bmatrix} \\
B_d &= \Delta t\left[3\Delta t A_c B_c + 6B_c\right]/6 = \begin{bmatrix} 0.0099500 & 0 \\ 0.000050000 & 0.0099000 \end{bmatrix} \\
N_{d1} &= \Delta t\left[3\Delta t(A_c N_{c1} + N_{c1}A_c) + 6N_{c1}\right]/6 = \begin{bmatrix} 0 & 0 \\ 0.0099000 & 0.0098000 \end{bmatrix} \\
N_{d2} &= \Delta t\left[3\Delta t(A_c N_{c2} + N_{c2}A_c) + 6N_{c2}\right]/6 = \begin{bmatrix} 0.0099500 & 0.0098500 \\ 0.000050000 & 0.000050000 \end{bmatrix}
\end{aligned}
\tag{28}
$$

The *m*-file *c2d* found in the MATLAB control toolbox converts linear state-space models from continuous-time to discrete-time using the expression [A_d, B_d] = *c2d* (A_c, B_c, Δt). The algorithm underlying this function calculates A_d and B_d using the matrix exponential expression

$$
\text{expm}\begin{bmatrix} A_c & B_c \\ 0 & 0 \end{bmatrix}\Delta t = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} A_c & B_c \\ 0 & 0 \end{bmatrix}\Delta t + \begin{bmatrix} A_c & B_c \\ 0 & 0 \end{bmatrix}^2 \frac{(\Delta t)^2}{2!} + \cdots \tag{29}
$$

from which one obtains (for $\Delta t = 0.01$)

$$
\begin{aligned}
A_d &= I + A_c\Delta t + A_c{}^2\frac{(\Delta t)^2}{2!} + A_c{}^3\frac{(\Delta t)^3}{3!} + \cdots = \begin{bmatrix} 0.99005 & 0 \\ 0.0098512 & 0.98020 \end{bmatrix} \\
B_d &= 0 + B_c\Delta t + A_c B_c\frac{(\Delta t)^2}{2!} + A_c{}^2 B_c\frac{(\Delta t)^3}{3!} + \cdots \\
&= \left[I\Delta t + A_c\frac{(\Delta t)^2}{2!} + A_c{}^2\frac{(\Delta t)^3}{3!} + \cdots \right]B_c = \begin{bmatrix} 0.0099502 & 0 \\ 0.000049503 & 0.0099007 \end{bmatrix}
\end{aligned}
\tag{30}
$$

The relevant discrete equation is again given by equation 26 with $N_{dj} = N_{cj}\Delta t$.

The function *c2d* can also be used to convert bilinear state-space models from continuous-time to discrete-time more directly and more accurately. This is done using the MATLAB expression [A_d, B_d] = *c2d* (A_c+u_1N_{c1}+ u_2N_{c2}, B_c, Δt), where the matrices u_jN_{cj} are input in combination with the input matrix A_c (see, for example, ref. 23). Note that this expression must be invoked at each time step as u_1 and u_2 will typically vary with time. The corresponding expressions for A_d and B_d at each time step can be obtained by replacing A_c in equations 30 with A_c+u_1N_{c1}+ u_2N_{c2} and result in the forms

$$A_d = I + (A_c + u_1 N_{c1} + u_2 N_{c2})\Delta t + (A_c + u_1 N_{c1} + u_2 N_{c2})^2 \frac{(\Delta t)^2}{2!} + \cdots$$

$$\begin{aligned} B_d &= 0 + B_c \Delta t + (A_c + u_1 N_{c1} + u_2 N_{c2}) B_c \frac{(\Delta t)^2}{2!} + \cdots \\ &= \left[I\Delta t + (A_c + u_1 N_{c1} + u_2 N_{c2}) \frac{(\Delta t)^2}{2!} + \cdots \right] B_c \end{aligned} \tag{31}$$

The relevant discrete equation now has the linear form

$$X(k+1) = A_d X(k) + B_d\, u(k) \tag{32}$$

rather than the bilinear form of equation 26 because the effects of the bilinear terms are incorporated into the calculated values of A_d and B_d given in equation 31.

The output equations corresponding to the various forms of equation 26 shown above all have the form

$$y(k) = C_d X(k) + D_d u(k) \tag{33}$$

where C_d and D_d are the matrices previously computed as part of the linear SID process in step 2 of the proposed computational procedure described earlier.

Some Additional Comments Related to Numerical Computations

It has been implicitly assumed that the continuous nonlinear differential equations of the original system have been discretized using finite-difference techniques and that an approximate continuous bilinear model of those equations has been obtained as described in this report. In this process, it is assumed that the time increment Δt used to discretize the temporal derivatives is sufficiently small to ensure that the unit-amplitude pulses that are imposed on the discrete model to compute its Volterra kernels produce a response that is effectively the impulse response of that system. CFD models resulting from finite-difference discretizations of specializations of the Navier-Stokes equations, for example, readily meet this requirement as the time increments typically used for that purpose are quite small. The same value of Δt can be used to discretize the identified continuous bilinear model to obtain the desired discrete bilinear model.

The size of Δt also affects other aspects of the numerical calculations. Specifically, its value must be chosen small enough to adequately evaluate both the single integrals defining G and H and the double integrals defining K_j. The size of Δt also influences the value chosen for the discrete time increment ΔT used to vary the time lag T. This consideration is driven by the need to balance the smallness of Δt with the different (but usually larger) smallness acceptable for ΔT to reduce computation time in the numerical evaluation of the double integrals. The size of Δt also influences (through ΔT) the number of components, *ncomp*, which need to be computed for the second-order kernels to ensure having the fidelity needed to represent system behavior.

As mentioned earlier, the solutions for the matrices G, H, K_j, and N_j given in the example of reference 13 were confirmed by the present authors as part of the vetting process. This was done both analytically (by formally integrating the relevant integral expressions) and then numerically (using $\Delta t = 0.00001$, $\Delta T = 0.0001$, and *ncomp* = 10000). Note that the matrices G, H, and K_j are intermediate matrices that are computed and then used to calculate the desired matrices N_j. Since all of these matrices must be calculated numerically in practice, it was of interest to obtain an indication of how much the final matrices N_j (and hence the response time histories obtained from the identified bilinear equations) could vary with changes in Δt, ΔT, and *ncomp*. Numerical simulations in which these parameters were varied systematically were performed to elicit this information. These simulations indicated that the results were relatively insensitive to the values chosen for those parameters as long as they represented a triplet of values that adequately captured the time-varying behavior of the significant portions of the kernels used to characterize the nonlinear system.

The discrete-time state-space matrices A_d, B_d, C_d, D_d identified using *ERA* were found to provide (via *d2c.m*) excellent approximations to the continuous-time *ABCD* needed to identify N_{cj} using the method of reference 13. This was exemplified by the fact that *ode45.m* solutions obtained using the original and identified continuous linear equations were in exact agreement in all the cases investigated.

Illustrative Results

Clearly there is a need to verify that the discrete kernels and the resulting discrete bilinear state-space equations obtained by the method of this report are adequate representations of the underlying nonlinear system equations. An acceptable level of confirmation can be obtained by evaluating the ability of the identified bilinear equations to predict the same response as the original nonlinear equations when subjected to the same inputs. To ensure that the new computational procedure is evaluated sufficiently and provides a wide range of comparative results with which to judge the validity and capability of the procedure, it was decided to subject the new procedure to a broad collection of test cases. The suite of test cases includes five SISO equations and one MISO equation. Five of the equations are continuous differential equations and one is a difference equation. Three of the equations are bilinear and three are nonlinear. Several input time histories were also defined and used in the present investigation. Many of these are novel in that they are atypical inputs devised to excite the systems in interesting and unusual ways. The set of inputs includes classical inputs such as steps and sinusoids and several novel inputs consisting of various combinations of different types of inputs (pulses, steps, ramps, sinusoids, and random).

As noted earlier, it is assumed that the continuous nonlinear differential equations describing the original or actual system have been discretized appropriately using established finite-difference modeling techniques. These discretized equations are the starting point for the present procedure and are used to compute both the first- and second-order discrete kernels needed by the procedure, and the response time histories that are regarded as the actual (or baseline) time histories of the original system to the imposed loadings. The continuous bilinear state-space representation obtained by the present method is discretized using one of the discretization approaches described earlier in this report. Of course, if the original continuous equations are already bilinear in form as in some of the illustrative examples, one of those approaches can be used to discretize the original equations. Pulse loading patterns were imposed as discussed earlier and depicted in Appendix B to compute the discrete kernels needed. The first-order kernels were computed using equation 22 while the second-order kernels were computed using equation 23.

The set of results presented for each example begins with the presentation of the discrete time histories of the first- and second-order discrete kernels that were computed using the pulse loading method of this report. In the present case, these kernels serve to characterize the dynamics of the systems treated and (when plotted) provide a pictorial indication of the extent of their temporal memory and hence time range of influence. Comparisons of the time histories obtained from the original and identified equations subject to several types of inputs intended to exercise the identification procedure are presented next. These time history comparisons served as the primary metric for evaluation of the method proposed in this report. Linearized versions of the original equations (either nonlinear or bilinear) and the linearized versions of the identified bilinear equations always gave results that were essentially identical and these comparisons are not presented. Nonlinear response time histories obtained by summing the first- plus second-order convolutions of the computed kernels with the imposed inputs are also presented and compared to their corresponding nonlinear responses for several of the examples. The first-order convolution responses were in all cases basically the same as the linear solutions and are not shown. For completeness, both the continuous-time state-space matrices that were identified and their discrete-time versions are presented for the examples.

Comment on MATLAB Scripts Used for Numerical Calculations

Several MATLAB scripts or programs were written and used for the studies that provided the numerical results for this report. Each illustrative example had its own program, thereby allowing each program to be tailored to the nuances of the system treated. This individuality also provided for easy incorporation of modifications to address what-if questions and computational issues. The programs also contain considerable superfluous coding intended for providing the alternative computational paths discussed earlier that could be activated for checking key computational aspects of the scripts. Such supplementary coding was also used to include options for selecting from a suite of several different types of inputs. Although there were some limited studies conducted to identify the effects of varying the parameters Δt, ΔT, and *ncomp*, there were no serious attempts to improve the overall results by tuning those parameters. Finally, as mentioned earlier, three existing MATLAB functions play a key role in the new methodology: *ERA* from the SOCIT Toolbox (ref. 30) and *d2c* and *c2d* from the MATLAB Control Toolbox (ref. 29).

Kernel Calculations, Convolutions, and Bilinear System Identification for SISO Systems

Five examples were selected from the literature to illustrate the SISO case: two bilinear systems and three nonlinear systems. To better demonstrate the identification capabilities of the new procedure, several novel inputs were devised and imposed on both the original and identified system equations and the resulting time histories compared. Zero initial conditions were assumed for the calculations. As mentioned earlier, these time history comparisons served as the primary metric for evaluation of the new procedure.

Example 1: Bilinear Equation

The bilinear differential equation given by

$$\ddot{y} + a_1 \dot{y} + a_o y = b_o u + n_o y u \tag{34}$$

was used in reference 1 to illustrate the calculation of discrete Volterra kernels using pulse inputs. The state-space form of this equation can be written as

$$\begin{aligned} \dot{X} &= A_c X + N_c X u + B_c u \\ y &= C_c X + D_c u \end{aligned} \tag{35}$$

where

$$A_c = \begin{bmatrix} 0 & 1 \\ -a_o & -a_1 \end{bmatrix} \quad N_c = \begin{bmatrix} 0 & 0 \\ n_o & 0 \end{bmatrix} \quad B_c = \begin{Bmatrix} 0 \\ b_o \end{Bmatrix} \quad C_c = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad D_c = \begin{bmatrix} 0 \end{bmatrix} \tag{36}$$

The constants in these matrices were arbitrarily chosen as $a_o = 2$, $a_I = 3$, $n_o = -6$, and $b_o = 1$ in the subject example. An input consisting of a single sawtooth wave ramping up from 0 to 1.0 over the time interval $t = 0.1$ to 1.1 seconds and zero elsewhere was applied to the system. Using a time step of 0.01 seconds, time histories were computed for 600 time points. The time between computed components of the second-order kernel was varied in increments of 10 time steps (ΔT=10Δt). Components of the second-order kernel were computed for 30 values of the time lag T. Thus, pertinent parameters for the numerical calculations are: $N = 600$, $\Delta t = 0.01$ sec, $\Delta T = 0.1$ sec, and *ncomp* = 30.

The state-space matrices corresponding to an Euler forward-difference approximation of the first-order form of the governing equations given by equation 35 are

$$
\begin{aligned}
A_d &= I + A_c\Delta t = \begin{bmatrix} 1 & 0.01 \\ -0.02 & 0.97 \end{bmatrix} \\
B_d &= B_c\Delta t = \begin{bmatrix} 0 \\ 0.01 \end{bmatrix} \\
N_d &= N_c\Delta t = \begin{bmatrix} 0 & 0 \\ -0.06 & 0 \end{bmatrix} \\
C_d &= C_c = \begin{bmatrix} 1 & 0 \end{bmatrix}; \quad D_d = D_c = \begin{bmatrix} 0 \end{bmatrix}
\end{aligned}
\tag{37}
$$

If the improved Euler method is used to discretize equations 35 the matrices above become

$$
\begin{aligned}
A_d &= I + \Delta t\left[3\Delta t A_c A_c + 6A_c\right]/6 = \begin{bmatrix} 0.99990 & 0.009850 \\ -0.019700 & 0.97035 \end{bmatrix} \\
B_d &= \Delta t\left[3\Delta t A_c B_c + 6B_c\right]/6 = \begin{bmatrix} 0.00005 \\ 0.00985 \end{bmatrix} \\
N_d &= \Delta t\left[3\Delta t(A_c N_c + N_c A_c) + 6N_c\right]/6 = \begin{bmatrix} -0.00030 & 0 \\ -0.05910 & -0.00030 \end{bmatrix} \\
C_d &= C_c = \begin{bmatrix} 1 & 0 \end{bmatrix}; \quad D_d = D_c = \begin{bmatrix} 0 \end{bmatrix}
\end{aligned}
\tag{38}
$$

Either finite-difference approximation is acceptable for this system and provides the discretized equations that can be used to compute both the kernels and response time histories of the original system. These kernels are then used in the procedure of this report to identify the state-space matrices of a new continuous bilinear representation of the original system. Either approximation can also be used to discretize the continuous bilinear equation that is identified by the process. For consistency, however, the same finite-difference approximation should be applied to both the original and identified equations. The improved Euler method was employed here.

The state-space matrices of the identified continuous bilinear model are

$$
\begin{aligned}
A_{cid} &= \begin{bmatrix} \text{-1.1147e-002} & \text{-1.2313e+000} \\ \text{1.5971e+000} & \text{-2.9887e+000} \end{bmatrix} \\
B_{cid} &= \begin{bmatrix} \text{-1.4956e+000} \\ \text{2.4450e+001} \end{bmatrix} \\
C_{cid} &= [\text{-2.8813e-002} \quad \text{-1.7605e-003}]]
\end{aligned}
\tag{39a}
$$

$$D_{cid} = [0]$$

$$N_{cid} = \begin{bmatrix} \text{-2.3027e-001} & \text{-1.3404e-002} \\ \text{4.2240e+000} & \text{2.4316e-001} \end{bmatrix}$$

These matrices were substituted into equation 38 to obtain the improved Euler discretized version of the identified continuous bilinear model. The state-space matrices of the resulting discrete bilinear model are

$$A_{did} = \begin{bmatrix} \text{9.9979e-001} & \text{-1.2128e-002} \\ \text{1.5732e-002} & \text{9.7046e-001} \end{bmatrix}$$

$$B_{did} = \begin{bmatrix} \text{-1.6461e-002} \\ \text{2.4072e-001} \end{bmatrix}$$

$$C_{did} = [\text{-2.8813e-002} \quad \text{-1.7605e-003}]] \tag{39b}$$

$$D_{did} = [0]$$

$$N_{did} = \begin{bmatrix} \text{-2.5635e-003} & \text{-1.3282e-004} \\ \text{4.1608e-002} & \text{2.0978e-003} \end{bmatrix}$$

The numerical results presented in reference 1 for both the first- and second-order kernels and the linear and bilinear responses to the single sawtooth input were replicated as a prelude to investigating system response to additional types of inputs (sine, random, up/down ramps, double sawtooths, oscillatory exponential, and steps). The computed first- and second-order kernels are shown in figures 9 and 10, respectively. Figure 11 shows the components of the second-order kernel shifted to time index zero for computation of K_j. The time histories computed using the identified discrete bilinear equations were in excellent agreement with the time histories computed using the discretized original bilinear equations for all of the input cases investigated. For example, figure 12 shows the responses of the linear system (N_d = 0), the original bilinear system, and the identified bilinear system to a unit step input imposed at $t = 0$ (k = 0). The agreement between the original bilinear and identified bilinear responses, the primary metric for evaluation of the present procedure, is excellent over the time range shown. While there is a maximum difference of 120% between the linear and the original bilinear responses, the maximum difference between the original bilinear and identified bilinear responses is less than 1%. A similar set of response time histories obtained for the double sawtooth input shown in figure 13 is presented in figure 14. The agreement between the original bilinear and identified bilinear responses is also excellent in this case but the corresponding differences (31% and 7%, respectively) are not as dramatic as for the step input case in figure 12.

Example 2: Nonlinear Circuit

The governing nonlinear equation for the current in a series circuit consisting of a linear inductance, a nonlinear resistance, and a voltage source is described by a Riccati equation having the form

$$\frac{dy}{dt} + \alpha y + \varepsilon y^2 = u(t) \tag{40}$$

This equation was used in reference 3 to illustrate the discrete-time application of the unit-impulse-based continuous kernel identification technique that was introduced in reference 24. The equation was discretized using an Euler forward-difference approximation according to

$$\begin{gathered} \frac{y_{i+1} - y_i}{\Delta t} + \alpha y_i + \varepsilon y_i^2 = u_i \\ y_{i+1} = y_i + \Delta t\left[u_i - \alpha y_i - \varepsilon y_i^2\right] \end{gathered} \tag{41}$$

using a time step of 0.01. The parameters *α* and ε were set to 0.1 and 0.001, respectively, and correspond to the Case 2 example in reference 3. The required sets of pulse time histories needed for calculating the kernels were computed for 5000 time points. The first-order kernel was computed using equation 22. Relevant components of the second-order kernel were computed using equation 23. The first- and second-order kernels of equation 40 are plotted in figures 15 and 16, respectively. Pertinent parameters for the numerical calculations are: N = 5000, Δt = 0.01 sec, ΔT = 1 sec, and *ncomp* = 20.

The identified continuous bilinear model is

$$\begin{aligned} A_{cid} &= [\text{-1.0005e-001}] \\ B_{cid} &= [\text{-4.4734e+000}] \\ C_{cid} &= [\text{-2.2366e-001}] \\ D_{cid} &= [0] \\ N_{cid} &= [\text{-7.1993e-003}] \end{aligned} \tag{42a}$$

These matrices are substituted into equation 25 to obtain the Euler discretized version of the identified continuous bilinear model. The resulting discrete bilinear model is

$$
\begin{aligned}
A_{did} &= [9.9900\text{e-}001] \\
B_{did} &= [-4.4734\text{e-}002] \\
C_{did} &= [-2.2366\text{e-}001] \\
D_{did} &= [0] \\
N_{did} &= [-7.1993\text{e-}005]
\end{aligned}
\tag{42b}
$$

The responses of this system were first determined for the same series of step inputs applied in reference 3 to confirm the ability to replicate the linear and nonlinear results presented there. These initial calculations made direct use of the first- and second-order kernels to compute the linear and nonlinear responses to the step inputs employing the same convolution techniques used in reference 3. Following these confirmations, the method of this report was applied to the subject equation to obtain the discrete bilinear model in equation 42b, and the time histories computed for several types of inputs. For example, computed response time histories for a unit step excitation of the linearized Riccati equation, nonlinear Riccati equation, and the identified bilinear equation are shown in figure 17. There is a maximum difference of 7% between the linear and the nonlinear responses, but a maximum difference of about 1% between the nonlinear and identified bilinear responses. A similar set of time histories was also obtained for the more complicated input shown in figure 18 and is presented in figure 19. The input of figure 18 is a combination of an exponentially-decaying sinusoid, a double sawtooth wave, and three pulses. It should be noted that the identified bilinear response tracks the nuances of the nonlinear response well over the entire time interval shown. The corresponding maximum differences in the responses for this composite input are 15% and 9%, respectively.

Example 3: Nonlinear Burgers Equation

The one-dimensional nonlinear partial differential equation given in equation 43

$$
\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \upsilon \frac{\partial^2 u}{\partial x^2} \tag{43}
$$

is known as Burgers equation and is often used as a simplified model of the Navier-Stokes equations for evaluating the performance of numerical methods in CFD calculations. This equation was used in reference 3 to illustrate the effectiveness of discrete-time Volterra kernels when applied to a simple CFD model. Equation 43 is usually written in the form (ref. 36)

$$
\frac{\partial u}{\partial t} + \frac{\partial E}{\partial x} = \upsilon \frac{\partial^2 u}{\partial x^2} \tag{44}
$$

prior to applying a finite-difference approximation, where $E = u^2/2$.

Applying a first-order forward difference approximation in time and a second-order central difference approximation in space to the continuous 1-D viscous Burgers equation given above yields

$$\frac{u(i,j+1)-u(i,j)}{\Delta t}=-\left[\frac{E(i+1,j)-E(i-1,j)}{2\Delta x}\right]+\nu\left[\frac{u(i+1,j)-2u(i,j)+u(i-1,j)}{(\Delta x)^2}\right] \quad (45)$$

or simply

$$u(i,j+1)=u(i,j)+\Delta t(\nu\overline{A}-\overline{B}) \quad (46)$$

where

$$\overline{A}=\left[\frac{u(i+1,j)-2u(i,j)+u(i-1,j)}{(\Delta x)^2}\right]$$

$$(47)$$

$$\overline{B}=\left[\frac{E(i+1,j)-E(i-1,j)}{2\Delta x}\right]$$

and i and j are the discrete space and time indices, respectively.

The numerical solution was implemented using a forward-in-time, central-in-space first-order Euler method with 40 spatial grid points, 400 time points, a time step of 0.01, and a spatial step of 0.2, as in reference 3. Viscosity v was set to unity. The end-point boundary condition grid point (grid point #1) is perturbed and the response to this perturbation is computed at the fifth grid point. Pertinent parameters for the numerical calculations are: $N = 400$, $\Delta t = 0.01$ sec, $\Delta x = 0.2$, $\Delta T = 0.01$ sec, and $ncomp = 10$.

The numerical results presented for the Burgers equation example in reference 3 were verified using the new coding associated with the present procedure. Because Burgers equation is a simplified version of the type of equation solved in CFD work, this equation was subjected to a variety of inputs in addition to those imposed in reference 3 to provide a broader base of results from which to evaluate the new procedure.

A low level of random excitation is included as part of several of the inputs used to excite the system to simulate a signal contaminated by noise. This random excitation is generated using the MATLAB function *randn.m*, which generates a random signal having a standard normal distribution with mean zero, variance one and standard deviation one, and then multiplied by a small number (usually 0.2) to set the level of the desired noise. The resulting random signal is then either used as-generated or smoothed slightly by taking a moving average of its time history over a small number of time steps (usually 2-5) using the MATLAB function *filtfilt*.

A key step of any linear system identification process is to estimate the order of the system to be identified. The ERA algorithm used in the present formulation does this by allowing the user to examine the magnitudes of the singular values that are obtained from a singular value decomposition (SVD) of the block Hankel matrix $H(0)$ formed from the pulse responses (Markov parameters) of the system and to specify how many should be regarded as significant. (See, for example, the ERA-related discussions in reference 30.) The SVD of the Hankel matrix obtained for Burgers equation yields 6 nonzero singular values with no clear demarcation between those that can be regarded as large and significant and those that can be regarded as small and negligible. Assuming 5 significant singular values [3] so that the order of the identified system is 5, the matrices identified for the approximate continuous bilinear model are:

$$A_{cid} = \begin{bmatrix} \text{-2.1725e+000} & \text{-2.9801e+000} & \text{1.7722e-001} & \text{7.6759e-002} & \text{-1.2702e-002} \\ \text{2.1458e+001} & \text{-3.8535e+001} & \text{1.4621e+001} & \text{3.1587e+000} & \text{-2.9964e-001} \\ \text{-6.8876e-002} & \text{-1.4481e+001} & \text{1.3185e-001} & \text{-3.2124e+000} & \text{1.2813e-001} \\ \text{3.1557e+001} & \text{-1.2808e+002} & \text{6.1621e+000} & \text{-8.6268e+001} & \text{2.2075e+001} \\ \text{-1.3765e+001} & \text{5.4086e+001} & \text{-1.7055e+000} & \text{4.9460e+001} & \text{-1.4774e+001} \end{bmatrix}$$

$$B_{cid} = \begin{bmatrix} \text{-2.0856e+001} \\ \text{8.8328e+001} \\ \text{4.0119e-001} \\ \text{1.4960e+002} \\ \text{-6.5060e+001} \end{bmatrix} \tag{48a}$$

$$C_{cid} = [\text{-7.9471e-002 -1.1843e-002 1.0676e-003 1.8120e-004 -8.7526e-006}]$$

$$D_{cid} = [\text{3.9062e-003}]$$

$$N_{cid} = \begin{bmatrix} \text{6.1416e-001} & \text{-1.1680e+000} & \text{3.0564e-001} & \text{-7.3164e-001} & \text{3.5517e-001} \\ \text{-4.1150e+000} & \text{7.2938e+000} & \text{-2.1442e+000} & \text{3.9978e+000} & \text{-2.0396e+000} \\ \text{-4.3236e-001} & \text{9.2021e-001} & \text{-2.0730e-001} & \text{5.4167e-001} & \text{-2.3519e-001} \\ \text{-1.1660e+001} & \text{2.4539e+001} & \text{-5.5829e+000} & \text{1.4912e+001} & \text{-6.5899e+000} \\ \text{6.0556e+000} & \text{-1.2417e+001} & \text{2.9426e+000} & \text{-7.3781e+000} & \text{3.3137e+000} \end{bmatrix}$$

[3] Although the results presented for this example were obtained assuming 5 significant singular values in the SID process, it was found that the selection of as few as two singular values led to acceptable results.

The continuous bilinear equations represented by these matrices were discretized using the first-order Euler method of equations 27 resulting in the discrete bilinear model

$$A_{did}=\begin{bmatrix} 9.7828\text{e-}001 & -2.9801\text{e-}002 & 1.7722\text{e-}003 & 7.6759\text{e-}004 & -1.2702\text{e-}004 \\ 2.1458\text{e-}001 & 6.1465\text{e-}001 & 1.4621\text{e-}001 & 3.1587\text{e-}002 & -2.9964\text{e-}003 \\ -6.8876\text{e-}004 & -1.4481\text{e-}001 & 1.0013\text{e+}000 & -3.2124\text{e-}002 & 1.2813\text{e-}003 \\ 3.1557\text{e-}001 & -1.2808\text{e+}000 & 6.1621\text{e-}002 & 1.3732\text{e-}001 & 2.2075\text{e-}001 \\ -1.3765\text{e-}001 & 5.4086\text{e-}001 & -1.7055\text{e-}002 & 4.9460\text{e-}001 & 8.5226\text{e-}001 \end{bmatrix}$$

$$B_{did}=\begin{bmatrix} -2.0856\text{e-}001 \\ 8.8328\text{e-}001 \\ 4.0119\text{e-}003 \\ 1.4960\text{e+}000 \\ -6.5060\text{e-}001 \end{bmatrix} \tag{48b}$$

$$C_{did}=[-7.9471\text{e-}002\ -1.1843\text{e-}002\ \ 1.0676\text{e-}003\ \ 1.8120\text{e-}004\ -8.7526\text{e-}006]$$

$$D_{did}=[3.9062\text{e-}003]$$

$$N_{did}=\begin{bmatrix} 6.1416\text{e-}003 & -1.1680\text{e-}002 & 3.0564\text{e-}003 & -7.3164\text{e-}003 & 3.5517\text{e-}003 \\ -4.1150\text{e-}002 & 7.2938\text{e-}002 & -2.1442\text{e-}002 & 3.9978\text{e-}002 & -2.0396\text{e-}002 \\ -4.3236\text{e-}003 & 9.2021\text{e-}003 & -2.0730\text{e-}003 & 5.4167\text{e-}003 & -2.3519\text{e-}003 \\ -1.1660\text{e-}001 & 2.4539\text{e-}001 & -5.5829\text{e-}002 & 1.4912\text{e-}001 & -6.5899\text{e-}002 \\ 6.0556\text{e-}002 & -1.2417\text{e-}001 & 2.9426\text{e-}002 & -7.3781\text{e-}002 & 3.3137\text{e-}002 \end{bmatrix}$$

Representative results from these numerical simulations are summarized in figures 20-31. As before, the primary comparisons are those involving the responses obtained from the nonlinear and identified bilinear equations. First- plus second-order convolution responses were also computed to provide an alternative solution of the nonlinear equation. These latter responses were obtained by convolving the input time histories with the first- and second-order kernels, respectively, and then adding the resulting time histories.

The first- and second-order kernels for Burgers equation are shown in figures 20 and 21, respectively. Note that all the nonzero-lag components of the second-order kernel are small and decay quickly with time compared to the first (zero-lag) component, indicating that the first component will be the dominant contributor to the second-order portion of the total response. Computed time histories delineating the extent to which the present method provides a suitable bilinear model for the subject equation for four different inputs are presented below.

A comparison of the linear, nonlinear, and identified bilinear equation responses to the noisy unit step input of figure 22 is presented in figure 23. The noisy portion of the step excitation is represented by an amplitude 0.2 random signal with no smoothing. In this case, there is a maximum difference of 17% between the linear and the nonlinear responses, but a maximum difference of 5% between the nonlinear and identified bilinear responses.

A similar comparison of responses to the noisy sine wave input of figure 24 is given in figure 25. The sine wave has a frequency of 5 Hz, amplitude of 0.5, and is contaminated with an unsmoothed random signal of amplitude 0.2. The response obtained using the identified bilinear model is in good agreement with the nonlinear response over the time range presented. There is a maximum difference of 4% between the nonlinear and identified bilinear responses, but a maximum difference of about 19% between the linear and the nonlinear responses. First- plus second-order convolution responses were also computed for this input and compared to the nonlinear response to provide an alternative (direct Volterra kernel-based) solution to the problem. These results are summarized in figure 26. The convolution response is also in good agreement with the nonlinear response throughout the time range, with a maximum difference of 6%.

A comparison of the linear, nonlinear, and identified bilinear equation responses to the random input of figure 27 is presented in figure 28. The random signal used in this case had unit amplitude and was smoothed by passing it through a length 5 moving-average filter. The differences between the nonlinear and linearized versions of Burgers equation for the random input are small. However, where there are notable differences the identified bilinear equation response is in good to excellent agreement with the nonlinear response. There is a maximum difference of 14% between the linear and the nonlinear responses, but a maximum difference of about 3% between the nonlinear and identified bilinear responses in this case. The nonlinear and first- plus second-order convolution responses for this random input are compared in figure 29. These results are also in good to excellent overall agreement, with the maximum difference in the responses being 12%.

The linear, nonlinear, and identified bilinear equation responses to the composite input depicted in figure 30 are compared in figure 31. The input signal in this case consisted of up-then-down ramps (breaking at time index 100), with a 0.5-amplitude, 5-Hz sine wave added to the down ramp and a smoothed 0.2-amplitude random signal to represent noise added to both the up and down ramps. The major differences in the responses occur in the interval straddling the point where the slope of the ramp input changes sign. There is a maximum difference of 37% between the linear and the nonlinear responses, but a maximum difference of about 14% between the nonlinear and identified bilinear responses. Again, the first- plus second-order convolution responses were also computed for this input and compared to the nonlinear response to provide an alternative solution to the problem. These results are included in figure 31. The maximum difference in the convolution versus nonlinear responses is 13%.

Example 4: Continuous Stirred Tank Reactor

The Van de Vusse Continuous Stirred Tank Reactor (CSTR) used in the chemical processing industry is often used as a benchmark problem by researchers for evaluating nonlinear process control algorithms. Reference 28 develops the nonlinear equations for a SISO process of that reactor and employs the bilinear approximation

$$\begin{aligned}\{\dot{X}\} &= [A]\{X\} + [N]\{X\}u + [B]u \\ \{y\} &= [C]\{X\} + [D]u\end{aligned} \tag{49}$$

that results from the application of Carleman linearization (ref. 24) to those nonlinear equations. The state vector specified for the linearization process contained the customary states, x_1 and x_2, and the second-order products of the states, x_1x_2, x_1^2, x_2^2, arranged in the form

$$\{X\} = \begin{bmatrix} x_1 & x_2 & x_1x_2 & x_1^2 & x_2^2 \end{bmatrix}^T \tag{50}$$

The matrices appearing in equation 49 are given by

$$A = \begin{bmatrix} -144.3 & 0 & 0 & -10 & 0 \\ 50 & -134.3 & 0 & 0 & 0 \\ 0 & 0 & -278.6 & 50 & 0 \\ 0 & 0 & 0 & -288.6 & 0 \\ 0 & 0 & 100 & 0 & -286.6 \end{bmatrix} \tag{51}$$

$$N = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ -1.12 & 7 & -2 & 0 & 0 \\ 14 & 0 & 0 & -2 & 0 \\ 0 & -2.24 & 0 & 0 & -2 \end{bmatrix} \tag{52}$$

$$\begin{aligned} B &= \begin{bmatrix} 7 & -1.12 & 0 & 0 & 0 \end{bmatrix}^T \\ C &= \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \end{bmatrix} \\ D &= \begin{bmatrix} 0 \end{bmatrix} \end{aligned} \tag{53}$$

from which it follows that $n = 5$, $m = 1$, and $r = 1$.

The state variables x_1 and x_2 are perturbation variables for the concentrations of the two components, designated A and B, in the mixture, and u is the inlet flow rate of component A. Note that $y = x_2$, i.e., the concentration of component B. Pertinent parameters for the numerical calculations are: $N = 250$, $\Delta t = 0.001$ hr, $\Delta T = 0.001$ hr, and $ncomp = 50$.

The identified continuous model is

$$A_{cid} = \begin{bmatrix} 4.4547\text{e-}001 & -2.3095\text{e+}002 \\ 8.4449\text{e+}001 & -2.7905\text{e+}002 \end{bmatrix}$$

$$B_{cid} = \begin{bmatrix} 4.3009\text{e-}001 \\ 3.5396\text{e+}001 \end{bmatrix}$$

$$C_{cid} = [-2.3468\text{e-}002\ -3.1357\text{e-}002] \tag{54}$$

$$D_{cid} = [0]$$

$$N_{cid} = \begin{bmatrix} -1.0093\text{e-}000 & 3.3286\text{e-}001 \\ 4.7480\text{e-}001 & -1.1441\text{e-}000 \end{bmatrix}$$

The continuous bilinear equations represented by these matrices were discretized using equations 31-33. Because the A_{did} and B_{did} matrices of this discrete model change at each time point, there is no single set of discrete matrices that define this model over time.

Figures 32 and 33 present the first- and second-order kernels, respectively. Individual step changes of +15 and –20 in the inlet flow rate were imposed separately as in reference 28 to verify the response time histories presented there using linear, nonlinear, and first- plus second-order Volterra kernel models. Subsequently, additional simulations were conducted. For example, figure 34 summarizes the time variations in the concentration of component B using five different system models for the case in which a step input of +15 is imposed first and then changed to –20 at time index 100. The overall agreement of the identified bilinear response with both the nonlinear and original bilinear responses is excellent, with a maximum difference of less than 2%, while the maximum difference between the linear and the nonlinear responses is 57%. The maximum difference in the nonlinear versus convolution responses is 19%.

Example 5: Nonlinear Difference Equation

The fifth and final SISO example uses the nonlinear difference equation

$$\begin{aligned} y(k) = 0.8\,y(k-1) + 0.2\,u(k-1) - 0.1\,y(k-1)\,u(k-1) \\ + 0.05y^2(k-1) - 0.05u^2(k-1) \end{aligned} \tag{55}$$

This equation is the same bilinear difference equation used in an example in reference 28 but augmented with the nonlinear terms involving u^2 and y^2. The present method was first applied to this equation without the additional terms to verify published results to an up-then-down stair-case-type input extending over 110 discrete time points. Because the equation is already in discrete form, $\Delta t = \Delta k = 1$. The pertinent parameters for the numerical calculations are: $N = 250$, $\Delta t = 1$ sec, $\Delta T = 1$ sec, and $ncomp = 10$.

The identified continuous bilinear model is

$$\begin{aligned} A_{cid} &= [\text{-2.1747e-001}] \\ B_{cid} &= [\text{-6.6079e-001}] \\ C_{cid} &= [\text{-3.3681e-001}] \\ D_{cid} &= [0] \\ N_{cid} &= [\text{-1.6199e-001}] \end{aligned} \tag{56a}$$

The continuous bilinear equation represented by these matrices was discretized according to the first-order Euler method of equations 25-26 yielding the discrete bilinear model

$$\begin{aligned} A_{did} &= [\text{7.8253e-001}] \\ B_{did} &= [\text{-6.6079e-001}] \\ C_{did} &= [\text{-3.3681e-001}] \\ D_{did} &= [0] \\ N_{did} &= [\text{-1.6199e-001}] \end{aligned} \tag{56b}$$

Illustrative results for this example are presented in figures 35-40. The first- and second-order kernels that were computed using the nonlinear difference equation are presented in figures 35 and 36, respectively. The responses of the linear, nonlinear, and identified bilinear equations to the step/pulse inputs shown in figure 37 are presented in figure 38. Overall agreement between the nonlinear and identified bilinear responses for these inputs is fair to good over the time range shown. Away from the spikes and dips in figure 38, there is a maximum difference of about 27% between the linear and the nonlinear responses, but a maximum difference of about only 1% between the nonlinear and the identified bilinear responses. Just prior to the sharp drops at time index 100, the maximum differences are 58% and 7%, respectively. A more complicated excitation consisting of a piece-wise combination of staircase, step, ramp, sawtooth, and sinusoidal inputs and depicted in figure 39 was also applied to the system. The first 80 time points of this input are identical to the leading portion of the staircase input used in reference 28. The resulting linear, nonlinear, and identified bilinear equation responses and first- plus second-order convolution responses for this input are compared in figure 40. Note that the identified bilinear response tracks the nuances of the nonlinear response rather well over the entire time interval shown, with a maximum difference of 9%, while the maximum difference between the convolution and nonlinear responses is about 4%. Note that the maximum difference between the liner and the nonlinear responses is 58%.

Viewed as a whole, the results presented above to illustrate the application of the method of reference 13 as extended in this report to SISO discrete problems works well, at least for the set of nonlinear and bilinear test cases investigated. In particular, even though the identified bilinear models were of small order (some as low as one), the responses computed with these models tracked the nuances of the responses obtained from the nonlinear or bilinear governing equations surprisingly well.

Bilinear System Identification Using MISO Equations

The equations of motion in the illustrative example of *Bruni, di Pillo, and Koch* (ref. 13) were used initially as-is by the present authors to obtain the results for the various multi-input cases to be presented below. These equations and their corresponding first- and second-order kernels are shown in equations 14 and 15 and figures 6 and 7, respectively. However, it was found that most of the computed original-versus-identified bilinear time histories, although in good agreement with each other, were not too much different from the computed linear time histories, indicating that the nonlinearities associated with the N_1 and N_2 terms in those equations were relatively weak. To better illustrate the identification capabilities of the present method for systems exhibiting stronger nonlinearities, it was decided to increase those matrices by some factor. Numerical studies indicated that a factor of 4 provided sufficient amplification to increase the nonlinear effects. The sign of N_2 was also changed in anticipation of providing more interesting behavior. The resulting modified version of the original continuous model of reference 13 with the changes underlined is:

$$\left[A_c\right]=\begin{bmatrix}-1 & 0\\ 1 & -2\end{bmatrix};\quad \left[B_c\right]=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix};\quad \left[C_c\right]=\begin{bmatrix}0 & 1\end{bmatrix};\quad \left[D_c\right]=\begin{bmatrix}0 & 0\end{bmatrix};$$

$$\left[N_{1c}\right]=\underline{4}\begin{bmatrix}0 & 0\\ 1 & 1\end{bmatrix};\quad \left[N_{2c}\right]=\underline{-4}\begin{bmatrix}1 & 1\\ 0 & 0\end{bmatrix}; \tag{57}$$

The first- and second-order kernels of the modified bilinear equation are presented in figures 41 and 42, respectively. Comparing these kernels to the comparable kernels shown in figures 4 and 5 for the original (unmodified) equation, it should be noted that the first-order kernels are unchanged (as expected) but that the second-order kernels are different (also as expected).

Comparisons of the linear, modified original bilinear, and identified bilinear equation responses for each of ten input cases are presented below. Pertinent parameters for the numerical calculations are: $N = 300$, $\Delta t = 0.01$ sec, $\Delta T = 0.1$ sec, and $ncomp = 30$.

The identified continuous bilinear model is:

$$A_{cid} = \begin{bmatrix} \text{-1.4390e+000} & \text{-1.1179e+000} \\ \text{-2.2031e-001} & \text{-1.5610e+000} \end{bmatrix}$$

$$B_{cid} = \begin{bmatrix} \text{-7.3142e-001} & \text{-1.6975e+001} \\ \text{1.5481e+001} & \text{-8.5276e+000} \end{bmatrix}$$

$$C_{cid} = [\text{-5.7526e-002 -2.7534e-003}] \tag{58}$$

$$D_{cid} = [0 \quad 0]$$

$$N_{1cid} = \begin{bmatrix} \text{6.2954e+000} & \text{-4.3427e+000} \\ \text{2.7896e+000} & \text{-1.9243e+000} \end{bmatrix}$$

$$N_{2cid} = \begin{bmatrix} \text{-2.4840e-001} & \text{1.7038e-001} \\ \text{5.8616e+000} & \text{-3.9614e+000} \end{bmatrix}$$

The modified original (57) and identified (58) continuous bilinear equations were discretized in accordance with equations 31-33, which form a new model at each time step. The resulting equations were then solved numerically for each of several different inputs and their response time histories compared. Ten sets of results, corresponding to the 10 different loading conditions, are presented below. Zero initial conditions were assumed for all the time history calculations. The modified original bilinear equation (eq. 57) is referred to as the "original bilinear equation" rather than as the "original bilinear equation as modified" for convenience of discussion below.

A comparison of the linear, original bilinear, and identified bilinear equation responses to the u_1 and u_2 step inputs shown in figure 43 is presented in figure 44. Note that the u_1 input is imposed at time zero (time index 0) but the u_2 input is delayed until time index 100. The responses obtained from the original bilinear and identified bilinear equations are in excellent agreement over the time interval shown, with a maximum difference of less than 1.5%.

A comparison of the linear, original bilinear, and identified bilinear equation responses to the u_1 and u_2 sine wave inputs of figure 45 is given in figure 46. The frequencies and amplitudes of u_1 and u_2 are 5 Hz and 2 Hz and 1 and 0.5, respectively. The u_2 input is delayed and not imposed until time index 50. Again, there is excellent agreement between the original and identified bilinear responses, with the differences being essentially 0% over the time interval.

The maximum difference between the linear and original bilinear responses is 13% for time indices greater than 75 but reaches values as high as 200% for time indices below 50.

A similar comparison of responses but this time to the pattern of sinusoidal inputs shown in figure 47 is given in figure 48. The inputs in this case differ from those in figure 45 only in that the frequencies have been interchanged. There is excellent agreement between the original and identified bilinear responses over the entire time interval presented, with a maximum difference of only 4%. However, there is a maximum difference of 150% between the linear and the original bilinear responses.

A comparison of the linear, original bilinear, and identified bilinear equation responses to the random inputs of figure 49 is presented in figure 50. The random signals used in this case were generated using the MATLAB function *randn.m* and then passed through a length 5 moving average filter using the MATLAB function *filtfilt.m*. Note that the imposition of the filtered u_2 input is delayed until time index 45. The undulations of the identified bilinear responses are in excellent agreement with those of the original bilinear equation and show a maximum difference of only 3%. In this case, the maximum difference between the linear and the original bilinear responses is 62%.

Response time histories obtained from the linear, original bilinear, and identified bilinear equations for the ramp inputs depicted in figure 51 are presented in figure 52. Input u_2 was delayed until time index 50. Note that there is a maximum difference of 105% between the linear and the original bilinear responses, but a maximum difference of only 9% between the original bilinear and identified bilinear responses in this case.

Responses of the linear, original bilinear, and identified bilinear equations to a sequence of u_1 and u_2 pulses varying in magnitude, duration, and sign and distributed as depicted in figure 53 are compared in figure 54. The overall agreement between the original bilinear and identified bilinear responses is good, with the maximum difference being 5%. There is a maximum difference of 153% between the linear and the original bilinear responses.

Responses of the linear, original bilinear, and identified bilinear equations to a 5-Hz sine wave on u_1 and a sharp pulse applied at time index 100 on u_2 as shown in figure 55 are presented in figure 56. Again, there is excellent agreement between the original and identified bilinear responses, with a maximum difference of about 5%. In this case, there is a maximum difference of 93% between the linear and the original bilinear responses.

Responses of the linear, original bilinear, and identified bilinear equations to a 2-Hz sine wave on u_1 and a 3-term pulse train on u_2 as shown in figure 57 are compared in figure 58. The response predicted by the identified bilinear equation is in excellent agreement with the behavior obtained from the original bilinear equation and show a maximum difference of only 3%. The maximum difference between the linear and the original bilinear responses is 150%.

Responses of the linear, original bilinear, and identified bilinear equations to the noisy step inputs of figure 59 are presented in figure 60. The u_1 and u_2 steps have amplitudes of +0.1 and −0.1, respectively and noise levels of 0.2 and 0.1, respectively. A drift of 0.002 per time step is also present in the u_2 input signal. The overall agreement between the predicted responses obtained using the original and identified bilinear equations is excellent, with the differences being essentially zero over the entire time interval. There is a maximum difference of 15% between the linear and the original bilinear responses.

The responses obtained for the same set of noisy step inputs depicted in figure 59 but with no drift in u_2 (figure 61) are compared in figure 62. The agreement between the original and identified bilinear responses is reasonably good in this case, with the major differences occurring at the upper end of the time range with a maximum difference of about 15%. There is a maximum difference of 200% between the responses from the linear and original bilinear equations.

Responses of the linear, original bilinear, and identified bilinear equations to the u_1 and u_2 sawtooth/ramp inputs of figure 63 are presented in figure 64. The agreement between the responses of the original and identified bilinear equations in this case is good to excellent depending on where the comparison is made, with the maximum difference being 18%. The maximum difference is 155% between the linear and the original bilinear responses.

Responses of the linear, original bilinear, and identified bilinear equations to a 2-Hz sine wave on u_1 and a series of up and down ramps on u_2 as depicted in figure 65 are given in figure 66. There is excellent overall agreement between the responses of the original and identified bilinear equations over the time interval shown, with the maximum difference being less than 3%. However, there is a maximum difference of 100% between the linear and original bilinear responses.

Taken as a whole, the results presented above to illustrate the application of the method of reference 13 as extended to discrete problems in this report works well, at least for the set of MISO (two input/single output) test cases investigated. Particularly noteworthy is the fact that, in spite of the small order of the identified bilinear model, the responses computed with the identified model predicted the nuances of the responses computed using the original bilinear model quite well.

Resultant Computational Procedure

The results of the illustrative SISO and MISO examples presented above show that the extension of the method of reference 13 proposed herein provides a practical computational procedure for determining a useful discrete bilinear state-space representation of a nonlinear system. A step-by-step outline of the process is given in figure 67, which shows the sequence of tasks associated with carrying out the necessary computations. Supplementary notes follow, keyed by number to the individual tasks:

1. Compute the first- and second-order discrete Volterra kernels: Use the pulse-loading method described in Appendices B and C to compute the first- and second-order discrete kernels for the nonlinear system of interest. For this purpose, it is assumed that the continuous differential equations describing the nonlinear system have been discretized using established

finite-difference modeling techniques appropriate to the problem and equations at hand. See equations 43-47 for an example of such a discretization. The second-order kernels so-computed are symmetric.

2. Identify the discrete state-space matrices A_d,B_d,C_d,D_d: Identify the discrete state-space matrices A_d,B_d,C_d,D_d using any procedure for realization of linear time-invariant systems applied to the first-order kernels computed in step 1. The procedure used here is ERA (ref. 30).

3. Convert the discrete state-space matrices A_d,B_d,C_d,D_d to their continuous counterparts: Use the *d2c.m* script of MATLAB to convert the discrete (A_d, B_d) from step 2 to their continuous counterparts (A_c, B_c). It should be noted that (C_c, D_c) = (C_d, D_d) so no formal conversion of (C_d, D_d) to (C_c, D_c) is required.

4. Convert the discrete symmetric second-order kernels to their continuous triangular counterparts: Following equation 24b, this conversion is made by multiplying the discrete second-order symmetric kernels computed in step 1 by $2/(\Delta t)^2$ and puts them into the form required by the method of reference 13.

The next three steps make direct use of three key steps in the procedure of reference 13.

5. Compute the subsidiary matrices G and H: Use the continuous matrices A_c,B_c,C_c from step 3 in equation 9 to compute the matrices G and H. The lower limit of integration t_1 is 0. The upper limit of integration t_2 is set to the same value used in step 6 below.

6. Compute the subsidiary matrices K_j: Use the continuous matrices A_c,B_c,C_c from step 3 and the continuous second-order kernels from step 4 in equation 11 to compute the matrices K_j. As discussed earlier, the components of the second-order kernels are shifted to the origin (time zero/time index 0) before numerically evaluating the double integrals. The lower limit of integration t_1 is then 0 for each component while the upper limit of integration t_2 is chosen large enough to include all the regions of influence of the second-order kernels.

7. Compute the continuous bilinear state-space matrices N_{cj}: Use the matrices G, H, and K_j from steps 5 and 6 in equation 12 to compute the bilinear state-space matrices N_{cj}.

At this point in the procedure, the matrices A, B, C, D, N_j of a continuous bilinear state-space representation of a nonlinear system have been identified.

8. Discretize the resulting identified continuous bilinear state-space equations as required: The identified continuous bilinear state-space equations must be discretized to obtain the desired discrete bilinear state-space representation of the original nonlinear equations. Finite-difference-based numerical modeling techniques as discussed in this report are suitable for this purpose. Equations 25-33 are illustrative of the form such discretized equations take.

Concluding Remarks

A practical computational procedure for calculating the state-space matrices corresponding to discrete-time bilinear representations of nonlinear systems has been presented. A key feature of the method is the use of discrete first- and second-order Volterra kernels to characterize the system. The present method is based on an innovative extension of the formulation for a continuous-time bilinear system identification procedure published in a 1971 paper by *Bruni, di Pillo, and Koch*. The analytical and computational considerations that underlie their method and its extension herein to the more general problem of identifying bilinear state-space representations of discrete nonlinear systems were described, pertinent numerical aspects associated with the process were discussed, and illustrative results from the application of the new procedure to several nonlinear systems from the literature were presented. Based on the results obtained from these exploratory numerical studies, the proposed procedure is a viable approach for calculating discrete-time bilinear state-space representations for approximating nonlinear systems and warrants further investigation. To this end, additional applications of the method are planned, with emphasis to be given to computational fluid dynamics-based aeroelastic problems.

Appendix A

Formulas for Computing First- and Second-Order Volterra Kernels Using Continuous-Time Bilinear State-Space Matrices

Reference 13 presents a general equation (in the guise of eq. 2.21 in that paper) for computing the continuous Volterra kernels appearing in equation 6 of the present report using the state-space matrices that define the continuous bilinear model of equation 5. The specific expressions for computing the first- and second-order kernels are given below.

First-Order Kernels

$$W_1(t) = \left[w_1^{(1)}(t) \;\; w_1^{(2)}(t) \cdots w_1^{(r)}(t) \right] = C_c e^{A_c t} B_c \tag{A1}$$

Second-Order Kernels

Expressions for the second-order kernels can be written in several forms. For example, defining b_j as the columns of the matrix B_c, i.e., $B_c = [b_1 \;\; b_2 \;\; b_3 \;\; \ldots \;\; b_r]$, the kernels may be expressed analytically as

$$\begin{aligned}
w_2{}^{(1,1)}(t_1,t_2) &= C_c e^{A_c t_1} N_{c1} e^{A_c t_2} b_1 \, \delta_{-1}(t_1)\delta_{-1}(t_2 - t_1) \\
w_2{}^{(1,2)}(t_1,t_2) &= C_c e^{A_c t_1} N_{c1} e^{A_c t_2} b_2 \, \delta_{-1}(t_1)\delta_{-1}(t_2 - t_1) \\
&\vdots \\
w_2{}^{(1,r)}(t_1,t_2) &= C_c e^{A_c t_1} N_{c1} e^{A_c t_2} b_r \, \delta_{-1}(t_1)\delta_{-1}(t_2 - t_1) \\
\\
w_2{}^{(2,1)}(t_1,t_2) &= C_c e^{A_c t_1} N_{c2} e^{A_c t_2} b_1 \, \delta_{-1}(t_1)\delta_{-1}(t_2 - t_1) \\
w_2{}^{(2,2)}(t_1,t_2) &= C_c e^{A_c t_1} N_{c2} e^{A_c t_2} b_2 \, \delta_{-1}(t_1)\delta_{-1}(t_2 - t_1) \\
&\vdots \\
w_2{}^{(2,r)}(t_1,t_2) &= C_c e^{A_c t_1} N_{c2} e^{A_c t_2} b_r \, \delta_{-1}(t_1)\delta_{-1}(t_2 - t_1) \\
&\vdots \\
w_2{}^{(r,1)}(t_1,t_2) &= C_c e^{A_c t_1} N_{cr} e^{A_c t_2} b_1 \, \delta_{-1}(t_1)\delta_{-1}(t_2 - t_1) \\
w_2{}^{(r,2)}(t_1,t_2) &= C_c e^{A_c t_1} N_{cr} e^{A_c t_2} b_2 \, \delta_{-1}(t_1)\delta_{-1}(t_2 - t_1) \\
&\vdots \\
w_2{}^{(r,r)}(t_1,t_2) &= C_c e^{A_c t_1} N_{cr} e^{A_c t_2} b_r \, \delta_{-1}(t_1)\delta_{-1}(t_2 - t_1)
\end{aligned} \tag{A2a}$$

These kernels may also be expressed more compactly as

$$
\begin{aligned}
W_2^{(1)}(t_1,t_2) &= \left[w_2^{(1,1)}(t_1,t_2) \;\; w_2^{(1,2)}(t_1,t_2) \;\; \cdots \;\; w_2^{(1,r)}(t_1,t_2) \right] \\
&= C_c e^{A_c t_1} N_{c1} e^{A_c t_2} B_c \, \delta_{-1}(t_1) \delta_{-1}(t_2 - t_1) \\
\\
W_2^{(2)}(t_1,t_2) &= \left[w_2^{(2,1)}(t_1,t_2) \;\; w_2^{(2,2)}(t_1,t_2) \;\; \cdots \;\; w_2^{(2,r)}(t_1,t_2) \right] \\
&= C_c e^{A_c t_1} N_{c2} e^{A_c t_2} B_c \, \delta_{-1}(t_1) \delta_{-1}(t_2 - t_1) \\
&\;\;\vdots \\
W_2^{(r)}(t_1,t_2) &= \left[w_2^{(r,1)}(t_1,t_2) \;\; w_2^{(r,2)}(t_1,t_2) \;\; \cdots \;\; w_2^{(r,r)}(t_1,t_2) \right] \\
&= C_c e^{A_c t_1} N_{cr} e^{A_c t_2} B_c \, \delta_{-1}(t_1) \delta_{-1}(t_2 - t_1)
\end{aligned}
\qquad \text{(A2b)}
$$

The kernels w_2 computed using equations A2 are triangular kernels as those expressions follow from a general equation that was derived in reference 13 assuming that $t_2 \geq t_1 \geq 0$. The unit step functions $\delta_{-1}(\;)$ appearing in these equations are usually omitted and assumed to be part of the kernels for convenience of notation.

Equations A1 and A2 can be used to compute analytical expressions for the first- and second-order Volterra kernels for a continuous bilinear system if the state-space matrices of the continuous bilinear equation of the system are known. This is how the kernels used in the example of reference 13 were obtained. The authors of that reference studied the relationships between the mathematical models given by equations 5 and 6 and developed a procedure that enables direct calculation of the state-space matrices for continuous bilinear systems given that system's first- and second-order kernels. In the present work, it is assumed that the underlying system is nonlinear rather than bilinear and the objective is to find a discrete bilinear representation of that nonlinear system.

Appendix B

Schematic Depiction of Pulse Loading Patterns for Calculating First- and Second-Order Volterra Kernels for Discrete Systems with Two Inputs

The pulse loading patterns that were imposed on a discrete system to compute its first- and second-order kernels are easily depicted for the case of two inputs. Because a pictorial representation of this case was used by the authors to help define a generalization to the case of more than two inputs (Appendix C) that representation is presented here for completeness.

The first- and second-order kernels were calculated using two different sets of subsidiary pulse responses which were computed first. Two subsidiary responses comprised the set used to calculate the first-order kernels, while three subsidiary responses comprised the set used to calculate the second-order kernels. These are described below.

First-Order Kernels

Following the procedure described in references 1 and 3, the first-order kernels $k_1^{(1)}$ and $k_1^{(2)}$ are each calculated using two subsidiary pulse responses which are obtained by separately imposing the pulse loadings depicted in figure B1 first to input u_1 and then to input u_2. The first subsidiary response, y_1, is simply the response of the nonlinear system to a unit-amplitude discrete pulse δ_0 imposed at k = 0 (t = 0). The other subsidiary response, y_{11}, is the response of the system to two unit-amplitude pulses $2\delta_0$ imposed at k = 0 (t = 0). Two such subsidiary responses are computed for each input of interest (two in this particular case). The resulting pairs of subsidiary responses are then combined according to

$$k_1^{(p)} = 2y_1{}^{(p)} - 0.5\mathrm{y}_{11}{}^{(p)}; \qquad p = 1,2 \tag{B1}$$

to obtain the first-order kernels. The calculation for the first-order kernel is done in this way so that $k_1^{(j)}$ reflects the linear portion of the nonlinear response (i.e., the linearized solution about a nonlinear steady-state condition rather than the purely linear response (i.e., linear response computed using linear equations). If $k_1^{(j)}$ is calculated this way, it will include some of the effect of any steady-state amplitude dependence that is present in the system.

The schematic depictions of the pulse loading patterns imposed for computing the subsidiary responses y_1 and y_{11} used to calculate the two first-order kernels for this case can be summarized in algebraic-like rules as follows:

Pulses used for computing subsidiary responses for kernel $k_1{}^{(1)}$:

y_1 ~ response to single unit pulse applied to input u_1 at t = 0 (k = 0)

y_{11} ~ response to two unit pulses applied simultaneously to input u_1 at $t = 0$ ($k = 0$)

Pulses used for computing subsidiary responses for kernel $k_1^{(2)}$:

y_1 ~ response to single unit pulse applied to input u_2 at $t = 0$ ($k = 0$)

y_{11} ~ response to two unit pulses applied simultaneously to input u_2 at $t = 0$ ($k = 0$)

Second-Order Kernels

The second-order kernels $k_2^{(p,q)}$ are calculated using three subsidiary pulse responses, which are computed differently depending on whether the kernel is on the diagonal ($p = q$) or off the diagonal ($p \neq q$). There are $r^i = 2^2 = 4$ second-order kernels for this case. Because each of the second-order kernels has *ncomp* components reflecting *ncomp* different time lags between the imposed pulses, each of the three subsidiary responses has *ncomp* components. The pulse-loading patterns needed for computing the subsidiary responses used to calculate the on-diagonal kernels is described in references 1 and 3. Loading patterns for computing the subsidiary responses for calculating the off-diagonal kernels were not available and had to be derived for this report.

Diagonal Kernels ($k_2^{(1,1)}, k_2^{(2,2)}$):

Following the procedure described in references 1 and 3, the subsidiary pulse responses needed for calculating the diagonal kernels $k_2^{(1,1)}$ and $k_2^{(2,2)}$ are obtained by separately imposing the pulse loadings depicted in figure B2, first to input u_1 and then to input u_2. Note that the time increment ΔT for varying the time lag T is depicted as $\Delta T = 4\Delta t$ in figure B2. This choice was arbitrary and done solely for pictorial convenience. In practice, ΔT is set according to $\Delta T = j\ \Delta t$, where j is an integer ≥ 1 chosen consistent with the guidelines discussed in the main body of this report for selecting values for Δt and ΔT. The first set of responses, y_1, is simply the response of the system to a unit-amplitude pulse applied at $k = 0$ ($t = 0$) as shown in figure B2a and used *ncomp* times to complete the first set. The second set, y_2, follows from imposing the pattern of (time-lagged) single unit pulses shown in figure B2b. It should be noted that, in practice, this set of responses does not actually have to be computed because for time-invariant systems as assumed herein they follow directly from the y_1 time history by simply shifting the y_1 curves in time by the appropriate amounts. The third set of responses, y_{12}, follows from imposing the pattern of time-lagged double unit pulses given in figure B2c. Note that the first component of this set is computed with a double unit pulse imposed at $k = 0$ ($t = 0$). Three such sets are computed for each input of interest (two in this case). The resulting sets of subsidiary responses are then combined (component-by-component) according to

$$k_2^{(p,p)} = 0.5\ \left(y_{12}^{(p,p)} - y_1^{(p,p)} - y_2^{(p,p)}\right); \qquad p = 1, 2 \tag{B2}$$

to obtain the second-order diagonal kernels.

Schematic depictions of the pulse loading patterns imposed for computing the subsidiary responses y_1, y_2, and y_{12} used to calculate the two on-diagonal kernels for this case can be summarized in algebraic-like rules as follows:

Pulses used for computing subsidiary responses for kernel $k_2^{(1,1)}$:

y_1 ~ unit pulse applied to input u_1 at $t = 0$ ($k = 0$); input $u_2 = 0$

y_2 ~ unit pulse applied to input u_1 at $t = 0, \Delta T, 2\,\Delta T, \ldots, (ncomp\ \text{-}1)\,\Delta T$; input $u_2 = 0$

y_{12} ~ two unit pulses applied to input u_1: one at $t = 0$ and the other at $t = 0, \Delta T, 2\,\Delta T, \ldots, (ncomp\ \text{-}1)\,\Delta T$; input $u_2 = 0$

Pulses used for computing subsidiary responses for kernel $k_2^{(2,2)}$:

y_1 ~ unit pulse applied to input u_2 at $t = 0$ ($k = 0$); input $u_1 = 0$

y_2 ~ unit pulse applied to input u_2 at $t = 0, \Delta T, 2\,\Delta T, \ldots, (ncomp\ \text{-}1)\,\Delta T$; input $u_1 = 0$

y_{12} ~ two unit pulses applied to input u_2: one at $t = 0$ and the other at $t = 0, \Delta T, 2\,\Delta T, \ldots, (ncomp\ \text{-}1)\,\Delta T$; input $u_1 = 0$

Off-Diagonal Kernels ($k_2^{(2,1)}, k_2^{(1,2)}$):

The off-diagonal second-order kernel $k_2^{(2,1)}$ is calculated using the three subsidiary pulse responses obtained from imposing the pulse loadings depicted in figure B3. Because the pulses are imposed only at integer multiples of ΔT when calculating the second-order kernels, only the ΔT time increments are indicated in figure B3. The first of these, y_1, is the response of the system to a unit-amplitude pulse applied to input u_1 at $k = 0$ ($t = 0$). The same response curve is used *ncomp* times to complete the first set. The second set, y_2, is obtained by imposing the pulse pattern shown in figure B3b to input u_2. The third set, y_{12}, follows from imposing the pattern of pulses given in figure B3c to inputs u_1 and u_2 simultaneously. The resulting sets of subsidiary responses are then combined (component-by-component) according to

$$k_2^{(2,1)} = 0.5\ (\mathrm{y}_{12}^{(2,1)} - \mathrm{y}_1^{(2,1)} - \mathrm{y}_2^{(2,1)}) \qquad \text{(B3)}$$

to obtain the second-order kernel $k_2^{(2,1)}$.

The off-diagonal second-order kernel $k_2^{(1,2)}$ is calculated using the three subsidiary pulse responses obtained from imposing the pulse loadings depicted in figure B4. Again, because the pulses are imposed only at integer multiples of ΔT, only the ΔT time increments are shown in figure B4. The first of these, y_1, is the response of the system to a unit-amplitude pulse applied to input u_2 at $k = 0$ ($t = 0$). The same response curve is used *ncomp* times to complete the first set. The second set of responses, y_2, is obtained by imposing the pulse loading pattern given in figure B4b at input u_1. The third set, y_{12}, follows from imposing the pattern of pulses given in figure B4c to inputs u_1 and u_2 simultaneously. The resulting sets of subsidiary responses are then combined (component-by-component) according to

$$k_2^{(1,2)} = 0.5\ (\mathrm{y}_{12}^{(1,2)} - \mathrm{y}_1^{(1,2)} - \mathrm{y}_2^{(1,2)}) \tag{B4}$$

to obtain the second-order kernel $k_2^{(1,2)}$.

While the pulse loadings used for computing the subsidiary responses needed for calculating the off-diagonal kernels are different (compare figures B3 and B4), it should be noted that not all subsidiary responses mentioned above actually need to be computed because the collection of all pulse loadings (and hence subsidiary responses) is not unique. For example, the following identities are present for the case of two inputs: y_1 of $k_2{}^{(1,1)} = y_1$ of $k_1{}^{(1)} = y_1$ of $k_2{}^{(2,1)}$, y_1 of $k_2{}^{(2,2)} = y_1$ of $k_1{}^{(2)} = y_1$ of $k_2{}^{(1,2)}$, y_2 of $k_2{}^{(2,2)} = y_2$ of $k_2{}^{(2,1)}$, and y_2 of $k_2{}^{(1,1)} = y_2$ of $k_2{}^{(1,2)}$. Thus, a reduction in computational effort can be realized if advantage is taken of such relationships.

The schematic depictions of the pulse loading patterns imposed for computing the subsidiary responses y_1, y_2, and y_{12} used to calculate the two off-diagonal kernels for this case can be summarized in algebraic-like rules as follows:

Pulses used for computing subsidiary responses used for calculating kernel $k_2{}^{(1,2)}$:

$y_1 \sim$ unit pulse applied to input u_2 at $t = 0$ ($k = 0$); input $u_1 = 0$

$y_2 \sim$ unit pulse applied to input u_1 at $t = 0, \Delta T, 2\ \Delta T, \ldots, (ncomp\ \text{-}1)\ \Delta T$; input $u_2 = 0$

$y_{12} \sim$ unit pulses applied to input u_1 at $t = 0, \Delta T, 2\ \Delta T, \ldots, (ncomp\ \text{-}1)\ \Delta T$ and input u_2 at $t = 0$

Pulses used for computing subsidiary responses used for calculating kernel $k_2{}^{(2,1)}$:

$y_1 \sim$ unit pulse applied to input u_1 at $t = 0$ ($k = 0$); input $u_2 = 0$

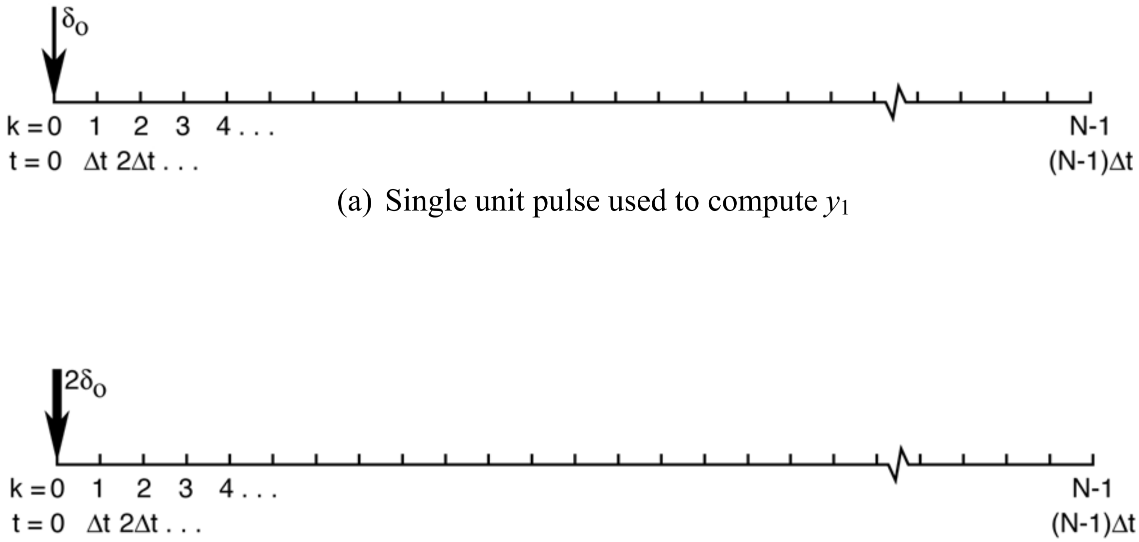
$y_2 \sim$ unit pulse applied to input u_2 at $t = 0, \Delta T, 2\ \Delta T, \ldots, (ncomp\ \text{-}1)\ \Delta T$; input $u_1 = 0$

$y_{12} \sim$ unit pulses applied to input u_2 at $t = 0, \Delta T, 2\ \Delta T, \ldots, (ncomp\ \text{-}1)\ \Delta T$ and input u_1 at $t = 0$

Careful inspection of these algebraic rules reveals a rule for determining which off-diagonal kernel is being calculated given values for the indices p and q. The rule depends on which pulse input is nonzero when computing the subsidiary responses y_1 and y_2. Specifically, whichever input is nonzero when computing y_1 sets the second index (q) and whichever input is nonzero when computing y_2 sets the first index (p).

A Comment on Indexing for the Off-Diagonal Kernels:

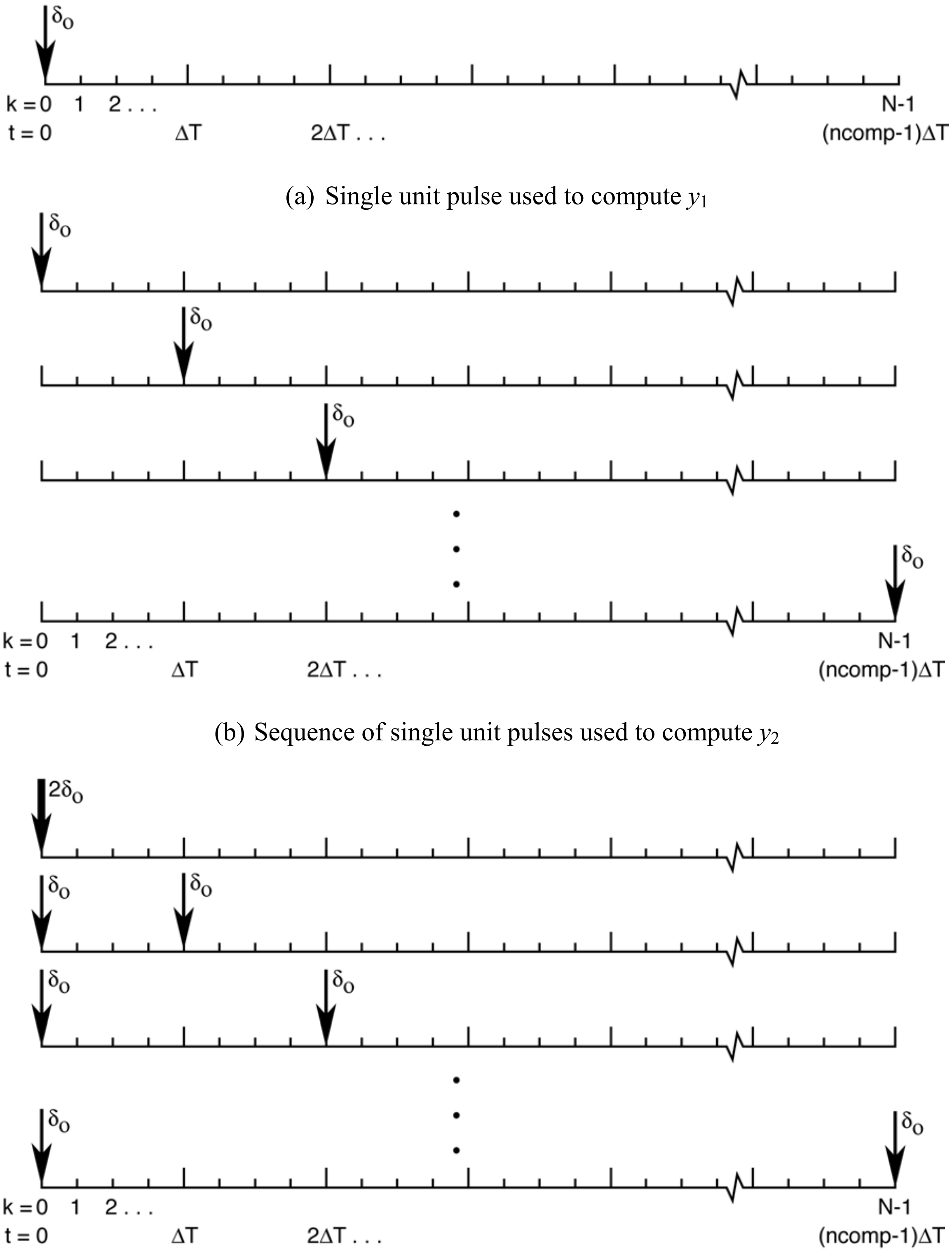
While the pulse loading patterns for computing the subsidiary responses used for calculating the diagonal kernels are well established (refs. 1, 3), the loading patterns needed for computing subsidiary responses for calculating off-diagonal kernels had to be defined. Pulse loading patterns for computing subsidiary pulse responses for calculating what were deemed to be the $k_2^{(2,1)}$ and $k_2^{(1,2)}$ kernels were devised. The kernels that were initially calculated for the example in reference 13 using these kernels were the reverse of the $w_2^{(1,2)}$ and $w_2^{(2,1)}$ kernels given in reference 13. This indicated that the index logic assumed to define the pulse loading patterns was reversed and needed adjustment. This adjustment led to the pulse loading patterns shown in figures B3 and B4 and to the rule noted above for determining which off-diagonal kernel is being computed.



(a) Single unit pulse used to compute y_1

(b) Double unit pulse used to compute y_{11}

Figure B1.- Unit-amplitude pulses imposed separately at inputs u_1 and u_2 to compute subsidiary responses used for calculating first-order kernels $k_1^{(1)}$ and $k_1^{(2)}$.

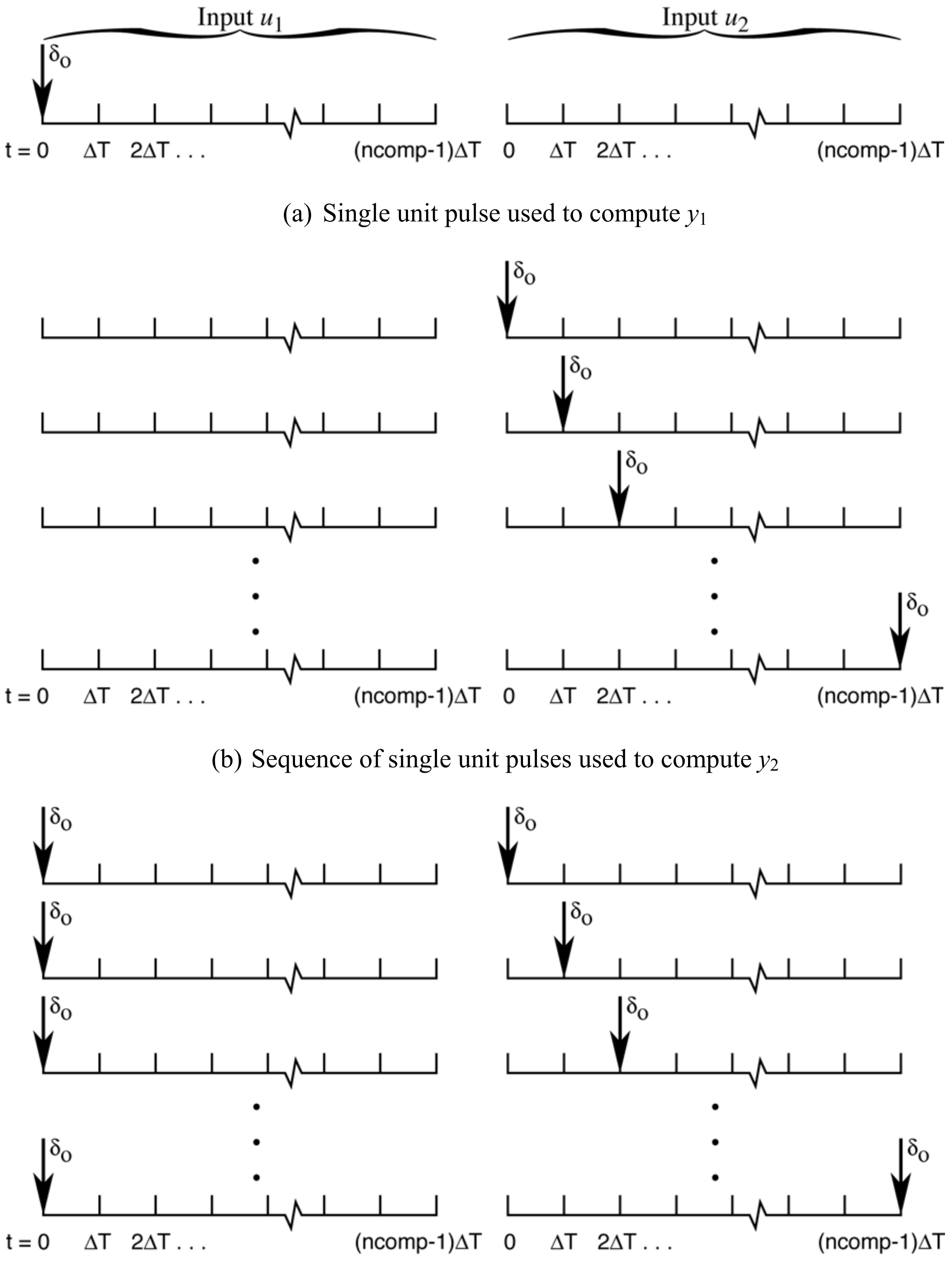


(a) Single unit pulse used to compute y_1

(b) Sequence of single unit pulses used to compute y_2

(c) Sequence of double unit pulses used to compute y_{12}

Figure B2.- Unit-amplitude pulses imposed separately at inputs u_1 and u_2 to compute subsidiary responses used for calculating second-order diagonal kernels $k_2^{(1,1)}$ and $k_2^{(2,2)}$.

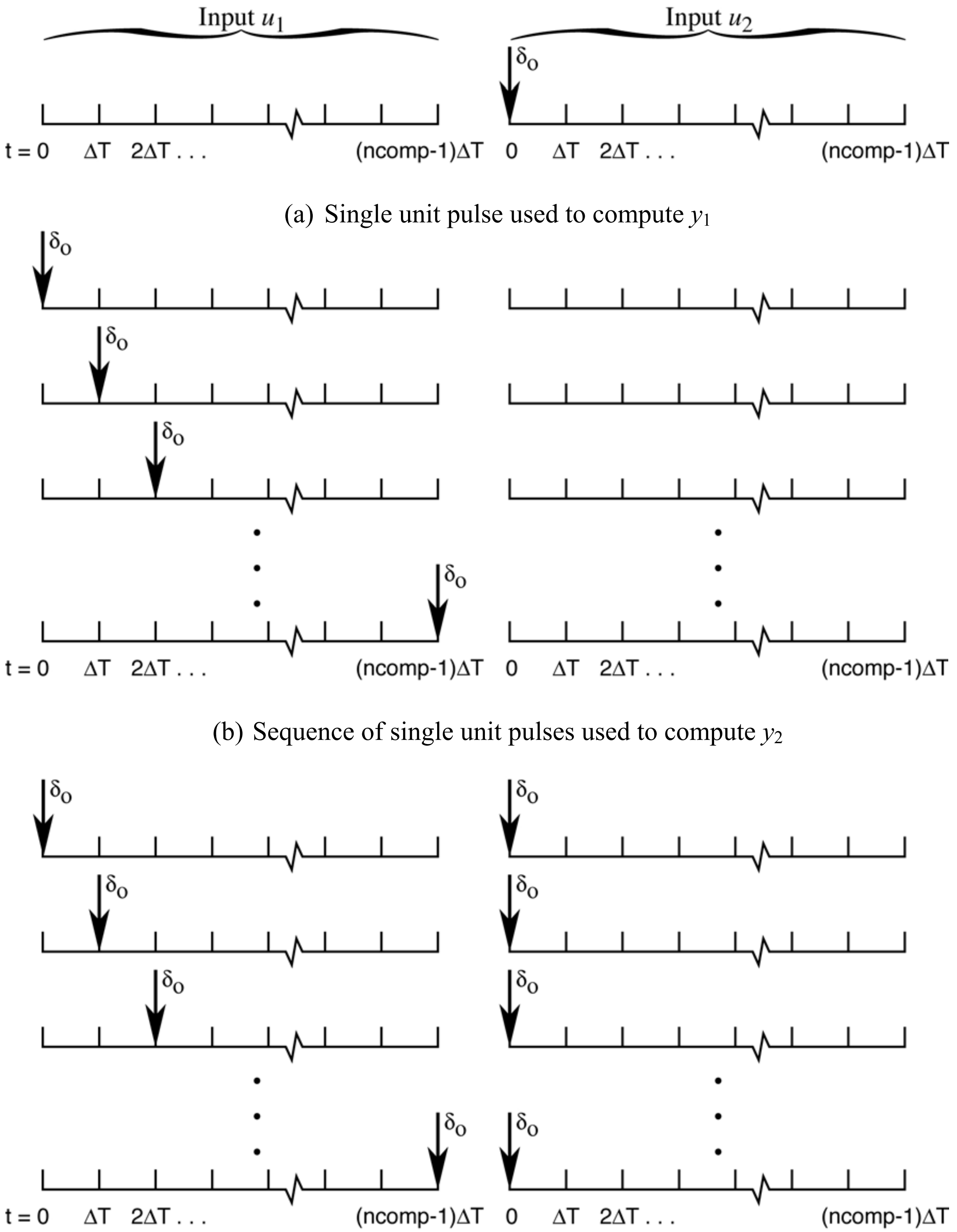


(a) Single unit pulse used to compute y_1

(b) Sequence of single unit pulses used to compute y_2

(c) Sequence of double unit pulses used to compute y_{12}

Figure B3.- Unit-amplitude pulses imposed simultaneously at inputs u_1 and u_2 to compute subsidiary responses used for calculating second-order diagonal kernel $k_2^{(2,1)}$.



(a) Single unit pulse used to compute y_1

(b) Sequence of single unit pulses used to compute y_2

(c) Sequence of double unit pulses used to compute y_1

Figure B4.- Unit-amplitude pulses imposed simultaneously at inputs u_1 and u_2 to compute subsidiary responses used for calculating second-order diagonal kernel $k_2^{(1,2)}$.

Appendix C

Pulse Loading Patterns for Calculating First- and Second-Order Volterra Kernels for Discrete Systems with an Arbitrary Number of Inputs

A schematic depiction of the pulse loading patterns that can be employed for calculating Volterra kernels for discrete systems with more than two inputs along the lines presented in Appendix B for the case of two inputs would be rather unwieldy. For this reason, this appendix provides no schematic depiction of this case but simply provides a collection of algebraic-like rules for imposing the pulse loading patterns employed for computing the subsidiary responses needed for calculating kernels for the general MIMO case.

First-Order Kernels

The first-order kernels $k_1^{(j)}$ for this case are calculated in exactly the same way as in Appendix B, except that there are additional inputs (and hence kernels) to deal with when r is greater than 2. There are r first-order kernels, where each kernel is an $m\times 1$ vector. For convenience, these kernels can be collected into a matrix K_1 according to:

$$K_1 = \left[k_1^{(1)}\ k_1^{(2)} \cdots k_1^{(r)} \right] \tag{C1}$$

The kernels $k_1^{(j)}$ are calculated using two subsidiary pulse responses $y_1^{(j)}$ and $y_{11}^{(j)}$ that are computed first using the following pulse loading patterns:

$y_1^{(j)} \sim$ response to single unit pulse δ_0 applied to input u_j at $t = 0$ ($k = 0$); $j = 1, 2, \ldots., r$

(C2)

$y_{11}^{(j)} \sim$ response to two unit pulses $2\,\delta_0$ applied simultaneously to input u_j at $t = 0$ ($k = 0$); $j = 1, 2, \ldots., r$

The time histories y_1 and y_{11} are calculated for each of the r inputs in turn with all of the other inputs set to zero. The kernels $k_1^{(j)}$ are then calculated using the expression (refs. 1, 3)

$$k_1^{(p)} = 2y_1^{(p)} - 0.5y_{11}^{(p)}; \quad p = 1,2,...,r \tag{C3}$$

The calculation of the first-order kernels is done in this way so that $k_1^{(j)}$ reflects the linear portion of the nonlinear response (i.e., linearized solution about a nonlinear steady-state condition rather than the purely linear response (i.e., linear response computed using linear equations). If $k_1^{(j)}$ is calculated this way, it will include some of the effect of any steady-state amplitude dependence that is present in the system.

Second-Order Kernels

The second-order Volterra kernels $k_2^{(p,q)}$ are calculated using three subsidiary pulse responses, which are computed differently depending on whether the kernel is on the diagonal (*p*=*q*) or off the diagonal (*p*≠*q*). The second-order kernels for the case of *r* inputs can be collected into matrices $\mathcal{K}_2^{(j)}$ according to:

$$\begin{aligned} \mathcal{K}_2^{(1)} &= \left[k_2^{(1,1)}\ k_2^{(1,2)} \cdots k_2^{(1,r)} \right] \\ \mathcal{K}_2^{(2)} &= \left[k_2^{(2,1)}\ k_2^{(2,2)} \cdots k_2^{(2,r)} \right] \\ &\vdots \\ \mathcal{K}_2^{(r)} &= \left[k_2^{(r,1)}\ k_2^{(r,2)} \cdots k_2^{(r,r)} \right] \end{aligned} \qquad \text{(C4)}$$

Each of the second-order kernels is composed of a finite number (*ncomp*) of *m*×1 vector components rather than a single vector as are the first-order kernels. Each of these components is computed using subsidiary pulse responses $y_1^{(p,q)}$, $y_2^{(p,q)}$, and $y_{12}^{(p,q)}$ that are calculated using pulse loading patterns to be described below. The pulse loading patterns that are imposed to compute the subsidiary responses differ depending on whether a diagonal ($p = q$) or an off-diagonal ($p \neq q$) kernel is being calculated. However, all the second-order kernels are computed (component-by-component) using the same general equation (refs. 1, 3):

$$k_2^{(p,q)} = 0.5\ (\mathrm{y}_{12}^{(p,q)} - \mathrm{y}_1^{(p,q)} - \mathrm{y}_2^{(p,q)})\ ; \qquad p,q = 1, 2, ..., r \qquad \text{(C5)}$$

There are $r^i = r^2$ second-order kernels. Each kernel is defined by *ncomp* *m*×1 vector components. Each of the subsidiary responses used to calculate the components is an *m*×1 vector. Also, by virtue of assuming that $t_1 \leq t_2$ while computing the components defining the kernels, the kernels computed by the present procedure are (inherently) symmetric.

The specific pulse loadings that are imposed to compute the subsidiary pulse responses used to calculate the second-order diagonal and off-diagonal kernels for the general MIMO case are described below. The reader may find it helpful to refer to the schematic depiction of the simpler case of two inputs presented in Appendix B while considering this more general case.

***Diagonal kernels* ($k_2^{(p,p)}$):**

The second-order diagonal kernels $k_2^{(p,p)}$ for this case are calculated in exactly the same way as in Appendix B, except that there are additional inputs (and hence kernels) to deal with when *r* is greater than 2. Because *p* = 1, 2, …, *r*, there are *r* second-order diagonal kernels. The subsidiary pulse responses y_1, y_2, and y_{12} are computed using the following pulse loading patterns:

y_1 ~ unit pulse δ_0 applied to input u_p at $t = 0$ ($k = 0$); inputs $u_j = 0,\ j \neq p;\ p, j = 1, 2, \ldots, r$

y_2 ~ unit pulse δ_0 applied to input u_p at $t = 0, \Delta T, 2\,\Delta T, \ldots, (ncomp\ \text{-}1)\,\Delta T$; inputs $u_j = 0,\ j \neq p;$ $p, j = 1, 2, \ldots, r$

y_{12} ~ two unit pulses $2\,\delta_0$ applied to input u_p: one at $t = 0$ and the other at $t = 0, \Delta T, 2\,\Delta T, \ldots,$ $(ncomp\ \text{-}1)\,\Delta T$; inputs $u_j = 0,\ j \neq p;\ p, j = 1, 2, \ldots, r$

Again, as in the case of two inputs treated in Appendix B, y_2 can be obtained from y_1 by time shifting y_1 appropriately.

Collecting the three sets of subsidiary pulse response time histories corresponding to each input, the diagonal second-order kernels are computed (component-by-component) using the expression

$$k_2^{(p,p)} = 0.5\ (\mathrm{y}_{12}^{(p,p)} - \mathrm{y}_1^{(p,p)} - \mathrm{y}_2^{(p,p)})\ ; \qquad p = 1, 2, \ldots, r \tag{C6}$$

Off-diagonal kernels ($k_2^{(p,q)}$):

For the off-diagonal kernels, $p \neq q$ and the subsidiary pulse responses y_1, y_2, and y_{12} are computed using the following pulse loading patterns, where $(p, q) = 1, 2, \ldots., r$ with $p \neq q$:

y_1 ~ unit pulse δ_0 applied to input u_q at $t = 0$ ($k = 0$); input $u_p = 0$; inputs $u_j = 0, j \neq p, q;\ j = 1, 2, \ldots, r$

y_2 ~ unit pulse δ_0 applied to input u_p at $t = 0, \Delta T, 2\,\Delta T, \ldots, (ncomp\ \text{-}1)\,\Delta T$; input $u_q = 0$; inputs $u_j = 0,\ j \neq p, q;\ j = 1, 2, \ldots, r$

y_{12} ~ unit pulses δ_0 applied to input u_p at $t = 0, \Delta T, 2\,\Delta T, \ldots, (ncomp\ \text{-}1)\,\Delta T$ and input u_q at $t = 0$; inputs $u_j = 0,\ j \neq p, q;\ j = 1, 2, \ldots, r$

Collecting the three sets of subsidiary pulse time histories corresponding to each input, the off-diagonal second-order kernels are computed (component-by-component) using the expression

$$k_2^{(p,q)} = 0.5\ (\mathrm{y}_{12}^{(p,q)} - \mathrm{y}_1^{(p,q)} - \mathrm{y}_2^{(p,q)})\ ; \qquad p, q = 1, 2, \ldots, r;\ \ p \neq q \tag{C7}$$

There are $r^2 - r$ second-order off-diagonal kernels.

As was found for the case of two inputs in Appendix B, the pulse loading patterns described above for computing the subsidiary responses needed for calculating the off-diagonal kernels are different. However, not all the subsidiary responses mentioned above actually need to be computed because the collection of all pulse loadings (and hence subsidiary responses) is not unique so that several sets of identities exist among them. Thus, as before, a considerable reduction in computational effort can be achieved if advantage is taken of such relationships.

A Comment on Indexing for the Off-Diagonal Kernels:

The rule for determining which off-diagonal kernel is being calculated that was established for the case of two inputs in Appendix B also holds for this case. Building on the results for the case of two inputs in Appendix B, the rule for determining which off-diagonal kernel is being calculated while cycling the indices p and q through off-diagonal values depends on which pulse inputs are nonzero when computing the subsidiary responses y_1 and y_2. Specifically, whichever input is nonzero when computing y_1 sets the second index q and whichever input is nonzero when computing y_2 sets the first index p.

References

1. Silva, W. A.: Application of Nonlinear Systems Theory to Transonic Unsteady Aerodynamic Responses. *Journal of Aircraft*, Vol. 30, No. 5, Sept.-Oct. 1993.

2. Silva, W. A.: *Identification of Linear and Nonlinear Aerodynamic impulse Responses Using Digital Filter Techniques*. NASA Technical memorandum 112872, August 1997.

3. Silva, W. A.: Discrete-Time Linear and Nonlinear Aerodynamic Impulse Responses for Efficient CFD Analyses. Ph.D. dissertation, College of William and Mary, October 1997.

4. Silva, W. A.; and Raveh, D. E.: Development of Unsteady Aerodynamic State-Space Models from CFD-Based Pulse Responses. Presented at the 42nd Structures, Structural Dynamics, and Materials Conference, Seattle, WA, April 16-19, 2001. Paper No. AIAA-2001-1213.

5. Silva, W. A.; and Bartels, R. E.: Development of Reduced-Order Models for Aeroelastic Analysis and Flutter Prediction Using the CFL3Dv6.0 Code. *Journal of Fluids and Structures*, Vol. 19, Issue 6, July 2004, pp. 729-745.

6. Silva, W. A.: Identification of Nonlinear Aeroelastic Systems based on the Volterra Theory: Progress and Opportunities. *Nonlinear Dynamics* (2005) 39: 25-62.

7. Silva, W. A.; Piatak, D. J.; and Scott, R. C.: Identification of Experimental Unsteady Aerodynamic Impulse Responses. *Journal of Aircraft*, Vol. 42, No. 6, Nov.-Dec. 2005.

8. Mohler, R. R.: *Bilinear Control Processes: with Applications to Engineering, Ecology, and Medicine*. Academic Press, New York, 1973.

9. Bruni, C.; Di Pillo, G.; and Koch, G.: Bilinear Systems: An Appealing Class of Nearly Linear Systems in Theory and Applications. *IEEE Transactions on Automatic Control*, Vol. AC-19, No. 4, August 1974, pp. 334-348.

10. Mohler, R. R.; and Kolodziej, W. J.: An Overview of Bilinear System Theory and Applications. IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-10, No. 10, October 1980, pp. 683-688.

11. Mohler, R. R.: Controls, Nonlinear Systems. *Encyclopedia of Science and Technology*, 1987, Vol. 3, pp. 659-674.

12. Mohler, R. R.: *Nonlinear Systems*: *Volume I, Dynamics and Control, Volume II, Applications to Bilinear Control*. Prentice Hall, Englewood Cliffs, NJ, 1991.

13. Bruni, C.; Di Pillo, G.; and Koch, G.: On the Mathematical Models of Bilinear Systems. *Ricerche di Automatica*, Vol. 2, No. 1, January 1971, pp. 11-26.

14. Isidori, A.: Direct Construction of Minimal Bilinear Realizations from Nonlinear Input-Output Maps. *IEEE Transactions on Automatic Control*, Vol. AC-18, No. 6, December 1973, pp. 626-631.

15. Ruberti, A.; Isidori, A.; and D'Alessandro, P.: Theory of Bilinear Dyanmical Systems. International Center for Mechanical Sciences, Courses and Lectures – No. 158, Udine, Italy, 1972, Springer, New York, 1973.

16. Tarn, T. J.; and Nonoyama, S.: Realization of Discrete-Time Internally Bilinear Systems. *Proc. IEEE Conf. Decision and Control*, Clearwater, FL, 1977, pp. 125-133.

17. Rodrigues, E. A.: Linear and Nonlinear Discrete-Time State-Space Modeling of Dynamic Systems for Control Applications. Ph.D. Dissertation, Purdue University, December 1993.

18. Verdult, V.; and Verhaegen, M.: Subspace-Based Identification of MIMO Bilinear Systems. *Proceedings of the European Control Conference*, Karlsruhe, Germany, September 1999.

19. Verdult, V.: Nonlinear System Identification: A State-Space Approach. Ph.D. Dissertation, University of Twente, The Netherlands, 2002.

20. Juang, J-N: *Continuous-Time Bilinear System Identification*. NASA/TM-2003-212646, September 2003. Also available in *Nonlinear Dynamics* (2005) 39: 79-94.

21. Glad, S. T.: General Systems or Extending Linear Theory to Nonlinear Systems. Department of Electrical Engineering, Linköping University, Linköping, Sweden, November 2004.

22. Pearson, R. K.: *Discrete-Time Dynamic Models*. Oxford University Press, New York, 1999.

23. Elliott, D. L.: Bilinear Systems, in *Encyclopedia of Electrical Engineering*, Vol. II. John Webster (ed.), John Wiley and Sons, New York, 1999, pp. 308-323.

24. Rugh, W. J.: *Nonlinear System Theory: The Volterra/Wiener Approach.* Johns Hopkins University Press, Baltimore, 1981.

25. Schetzen, M.: *The Volterra and Weiner Theories of Nonlinear Systems*. Krieger Publishing Company, Malabar, FL, 1989.

26. Maner, B. R.: Nonlinear Model Predictive Control with Second-Order Volterra Models. M.S. Thesis, Purdue University, 1993.

27. Bendat, J. S.: Nonlinear System Techniques and Applications. John Wiley & Sons, Inc., New York, 1998.

28. Doyle III, F. J.; Pearson, R. K.; and Ogunnaike, B. A.: *Identification and Control Using Volterra Models*. Springer, 2002.

29. MATLAB Reference Guide. The MathWorks, Inc., Natick, MA, August 1992.

30. Juang, J-N.: *Applied System Identification*. Prentice-Hall, Englewood Cliffs, NJ, 1994.

31. Carnahan, B.; Luther, H. A.; and Wilkes, J. O.: *Applied Numerical Analysis*. John Wiley & Sons, Inc., New York, 1969.

32. Collatz, L.: *The Numerical Treatment of Differential Equations*. Springer-Verlag, New York, 1966.

33. Forsythe, G. E.; Malcolm, M. A.; and Moler, C. B.: *Computer Methods for Mathematical Computations*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1977.

34. Hildebrand, F. B.: *Finite-Difference Equations and Simulations*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1968.

35. Ralston, A.: *A First Course in Numerical Analysis*. McGraw-Hill Book Co., New York, 1965.

36. Hoffmann, K. A.; and Chiang, S. T.: *Computational Fluid Dynamics for Engineers. Volume I.* Engineering Educational System, Wichita, KS, 1993.

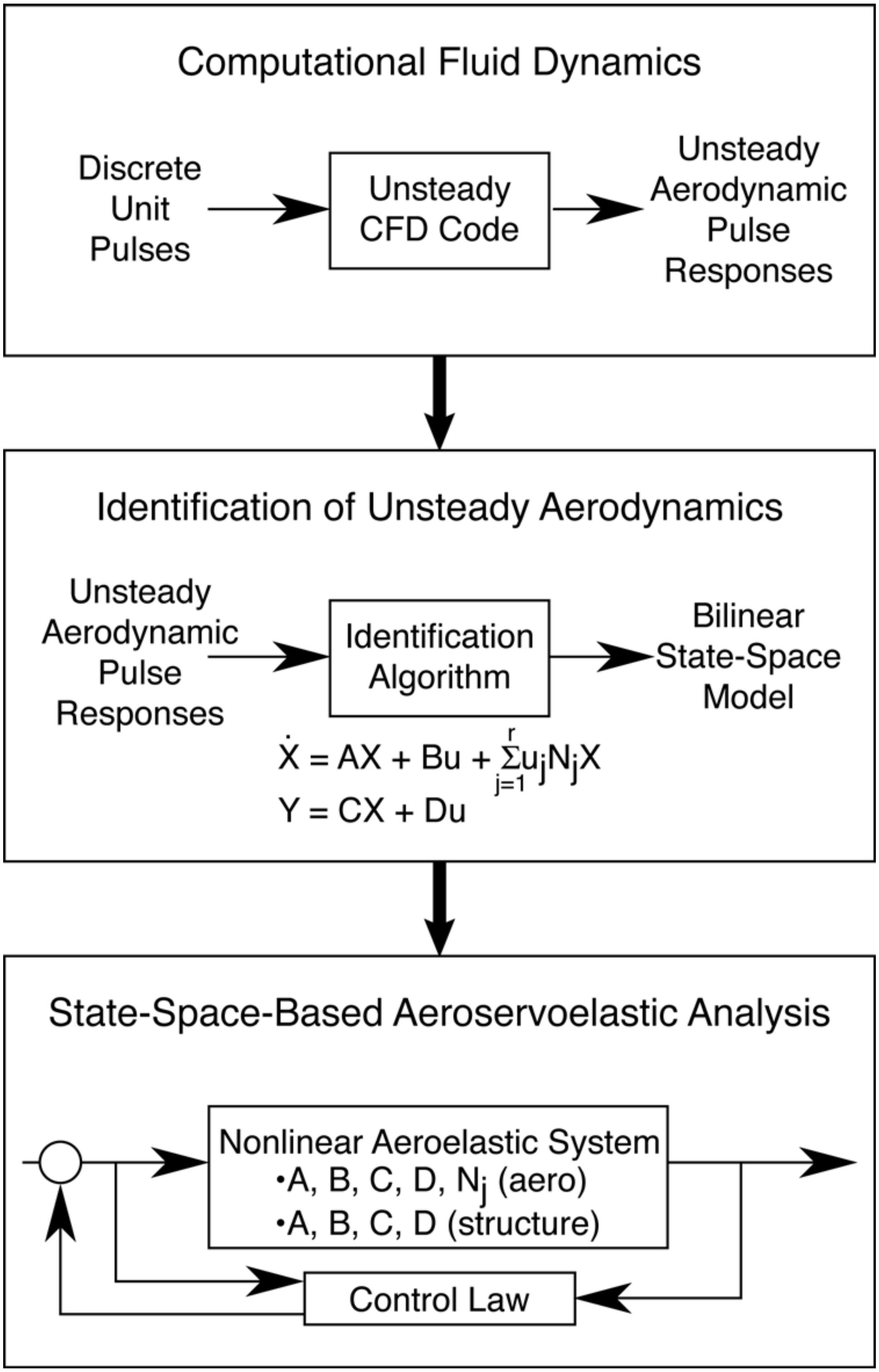

Computational Fluid Dynamics
Discrete Unit Pulses
Unsteady CFD Code
Unsteady Aerodynamic Pulse Responses
Identification of Unsteady Aerodynamics
Unsteady Aerodynamic Pulse Responses
Identification Algorithm
Bilinear State-Space Model
$\dot{X} = AX + Bu + \sum_{j=1}^{r} u_j N_j X$
$Y = CX + Du$
State-Space-Based Aeroservoelastic Analysis
Nonlinear Aeroelastic System
•A, B, C, D, N_j (aero)
•A, B, C, D (structure)
Control Law

Figure 1.- Notional block diagram of CFD-based reduced-order modeling approach for nonlinear aeroservoelastic analysis.

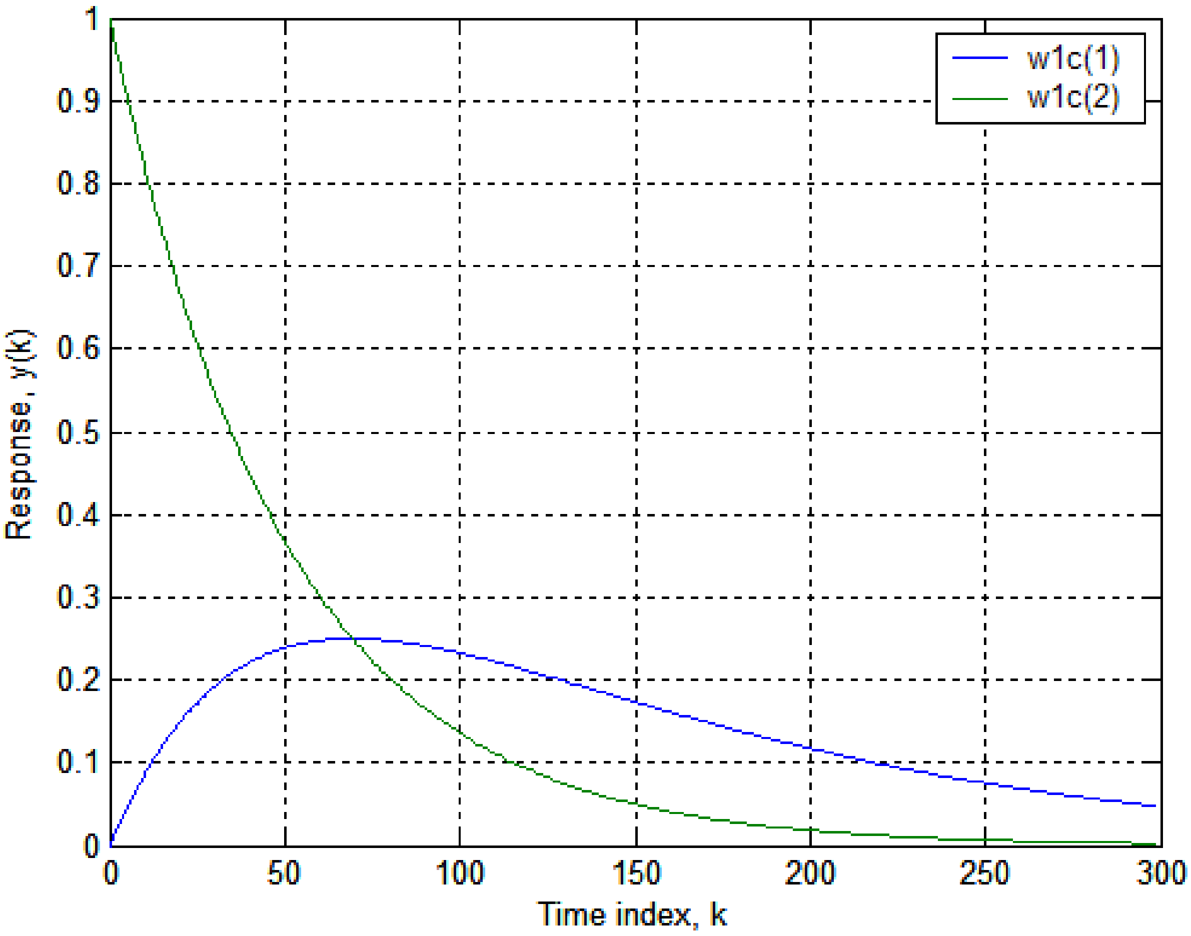


Figure 2.- Continuous first-order kernels of ref. 13 (*N* = 300, *dt* = 0.01 sec).

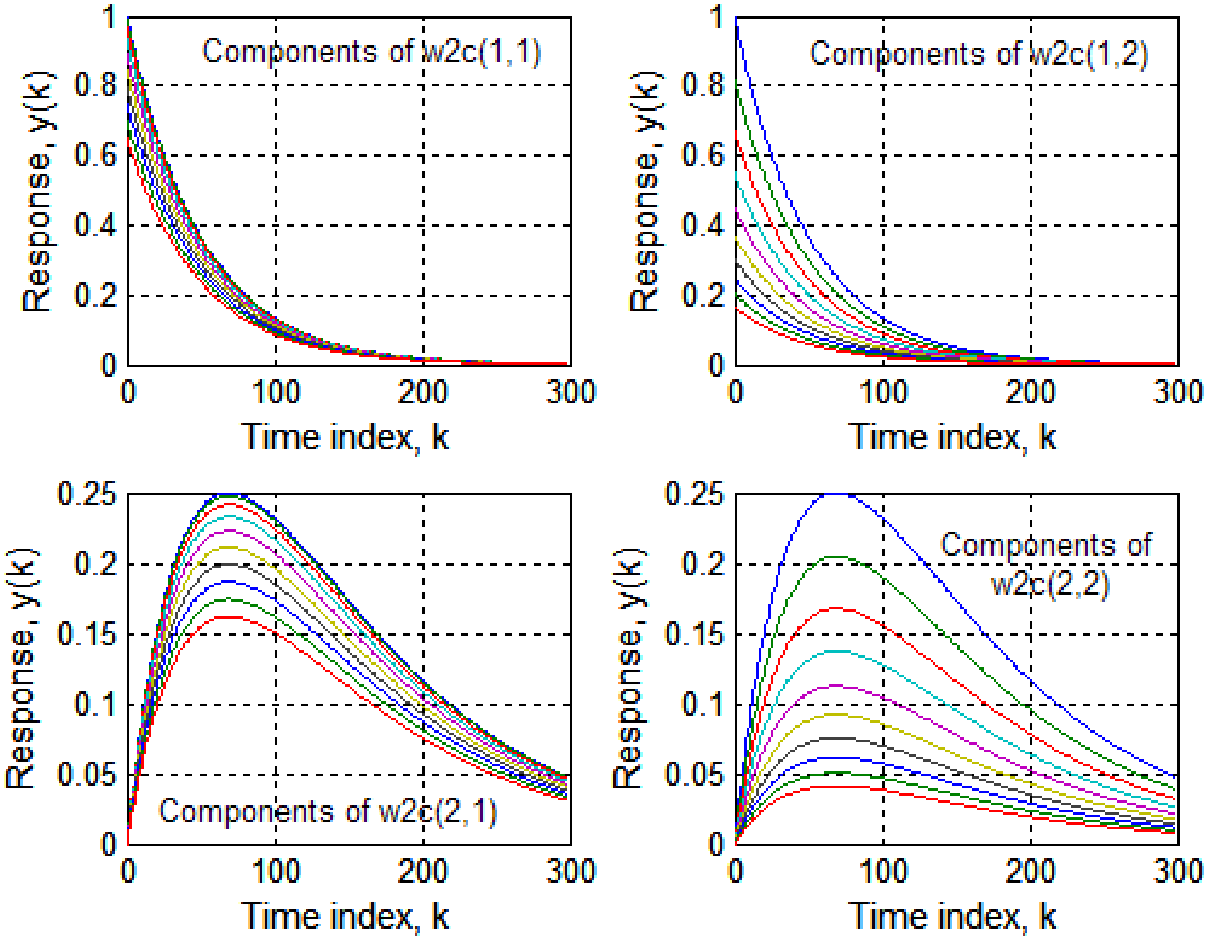


Figure 3.- Continuous second-order kernels of ref. 13 at 10 values of time lag *T* (*N* = 300, *dt* = 0.01 sec, *dT* = 0.1 sec, and *ncomp* = 10).

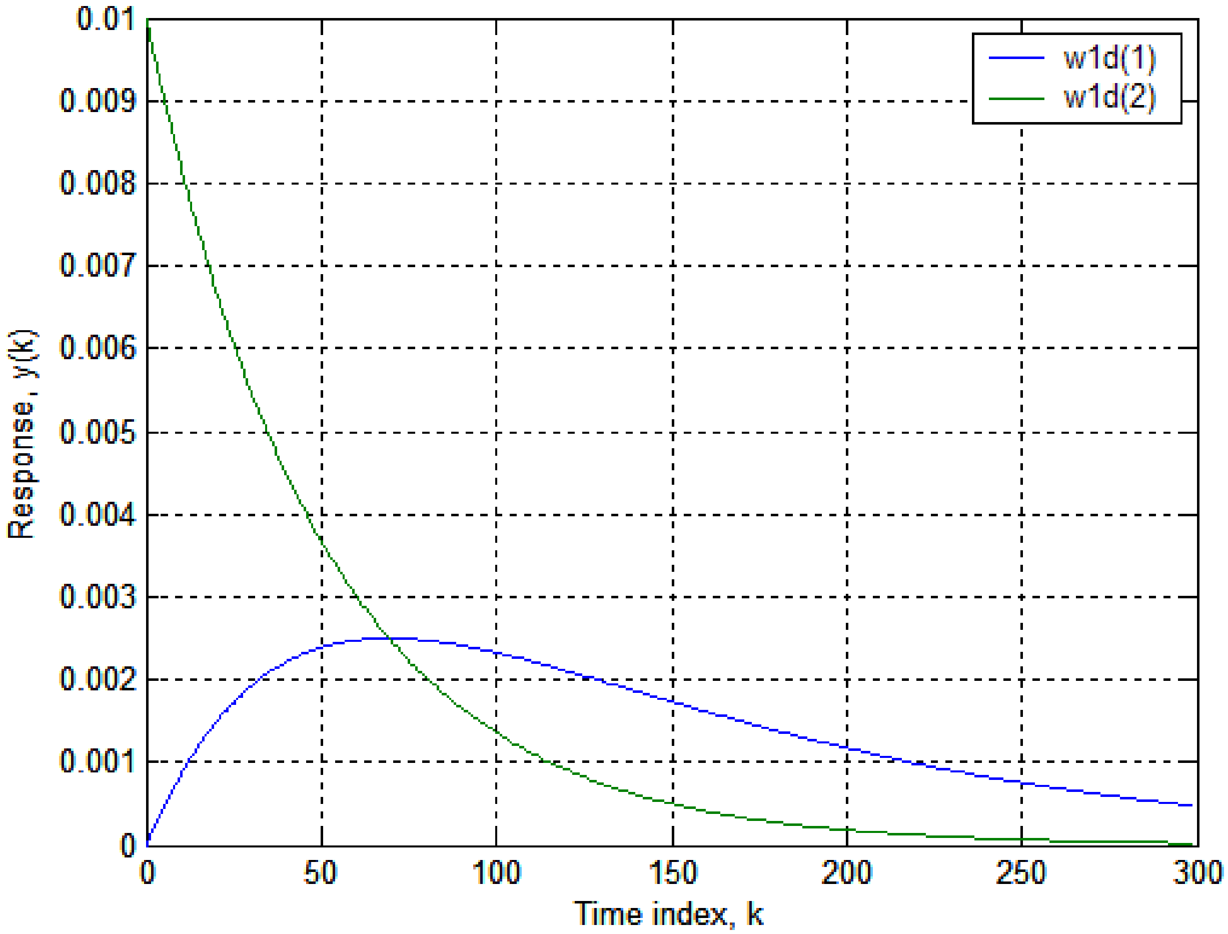


Figure 4.- Discretized first-order kernels of ref. 13 ($N = 300$, $\Delta t = 0.01$ sec).

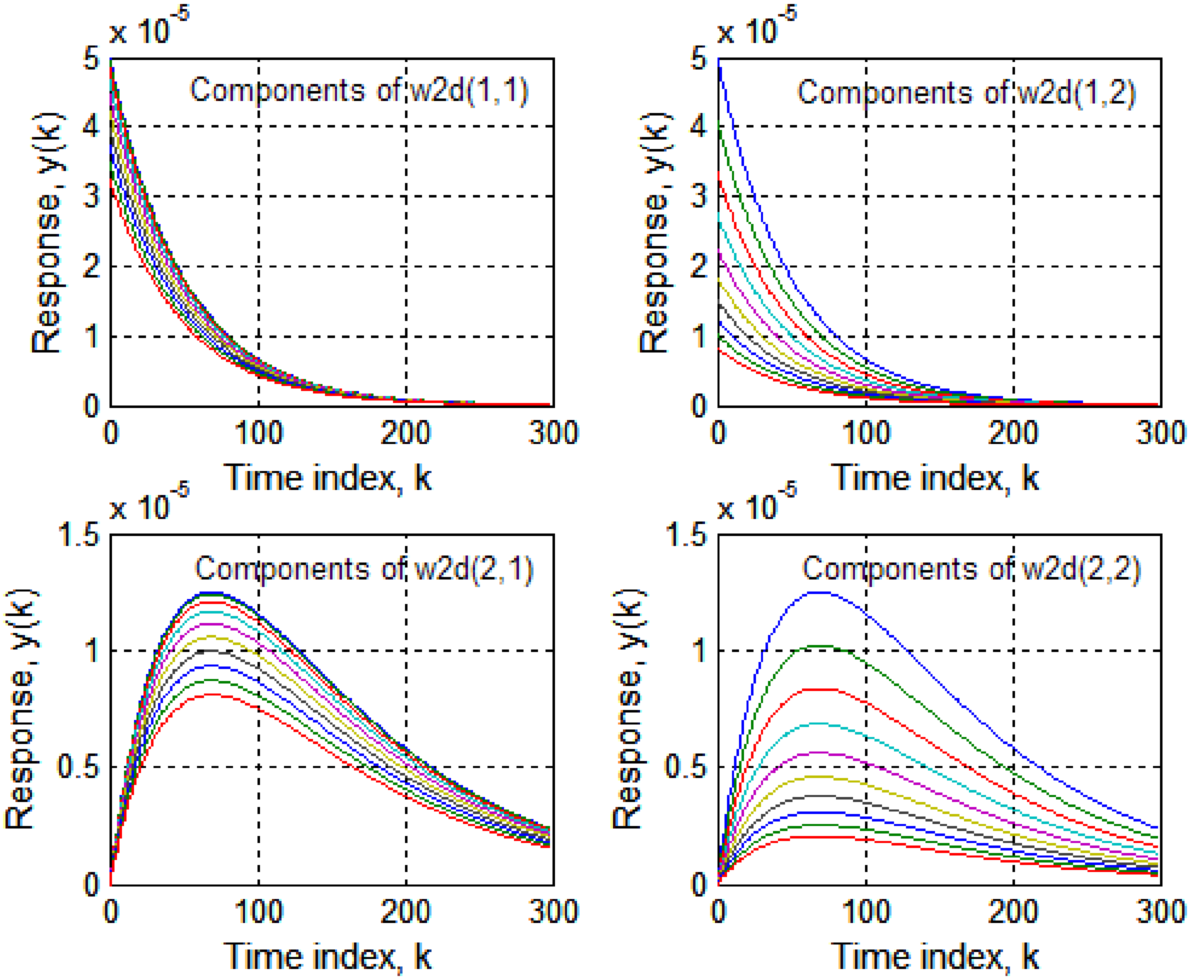


Figure 5.- Discretized second-order kernels of ref. 13 at 10 values of time lag *T* (N = 300, Δt = 0.01 sec, ΔT = 0.1 sec, and *ncomp* = 10).

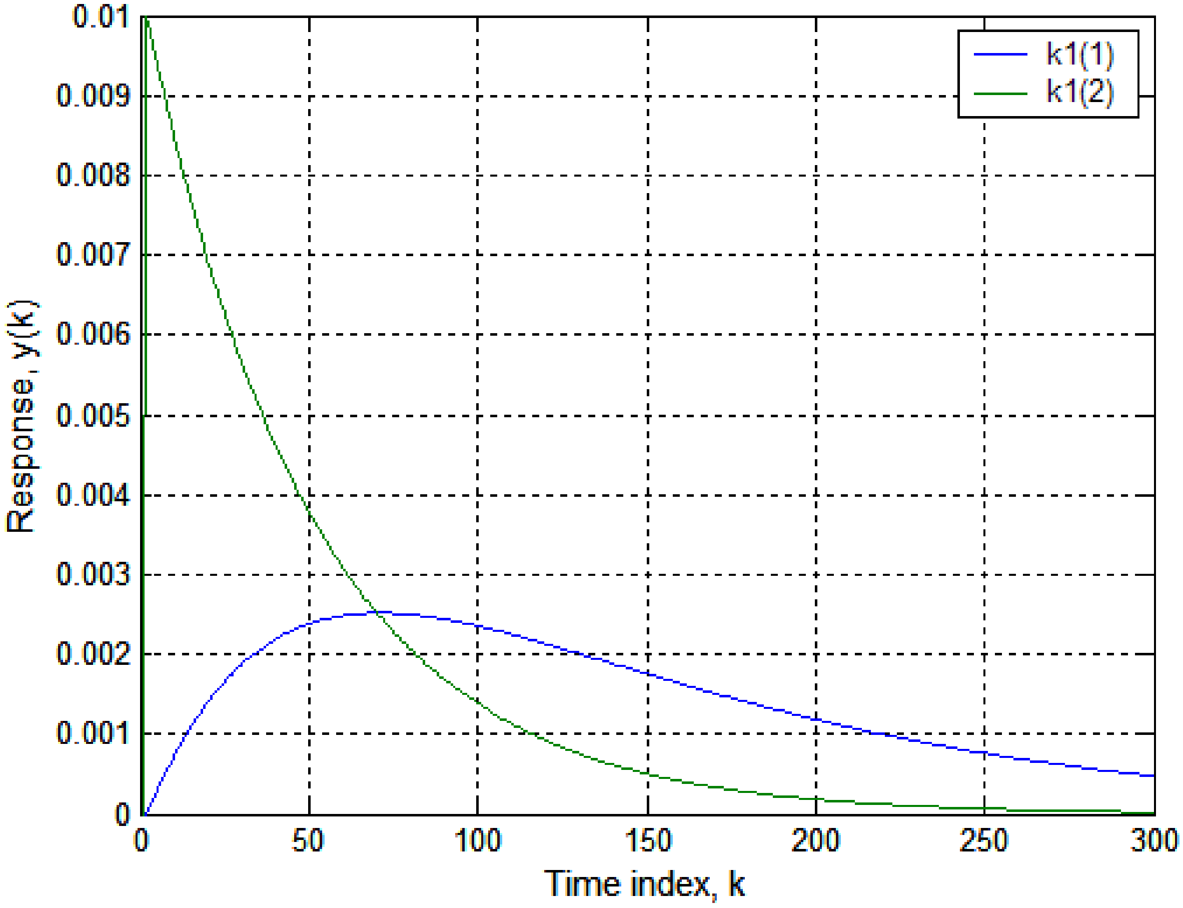


Figure 6.- Discrete first-order kernels calculated using method of present report ($N = 300$, $\Delta t = 0.01$ sec).

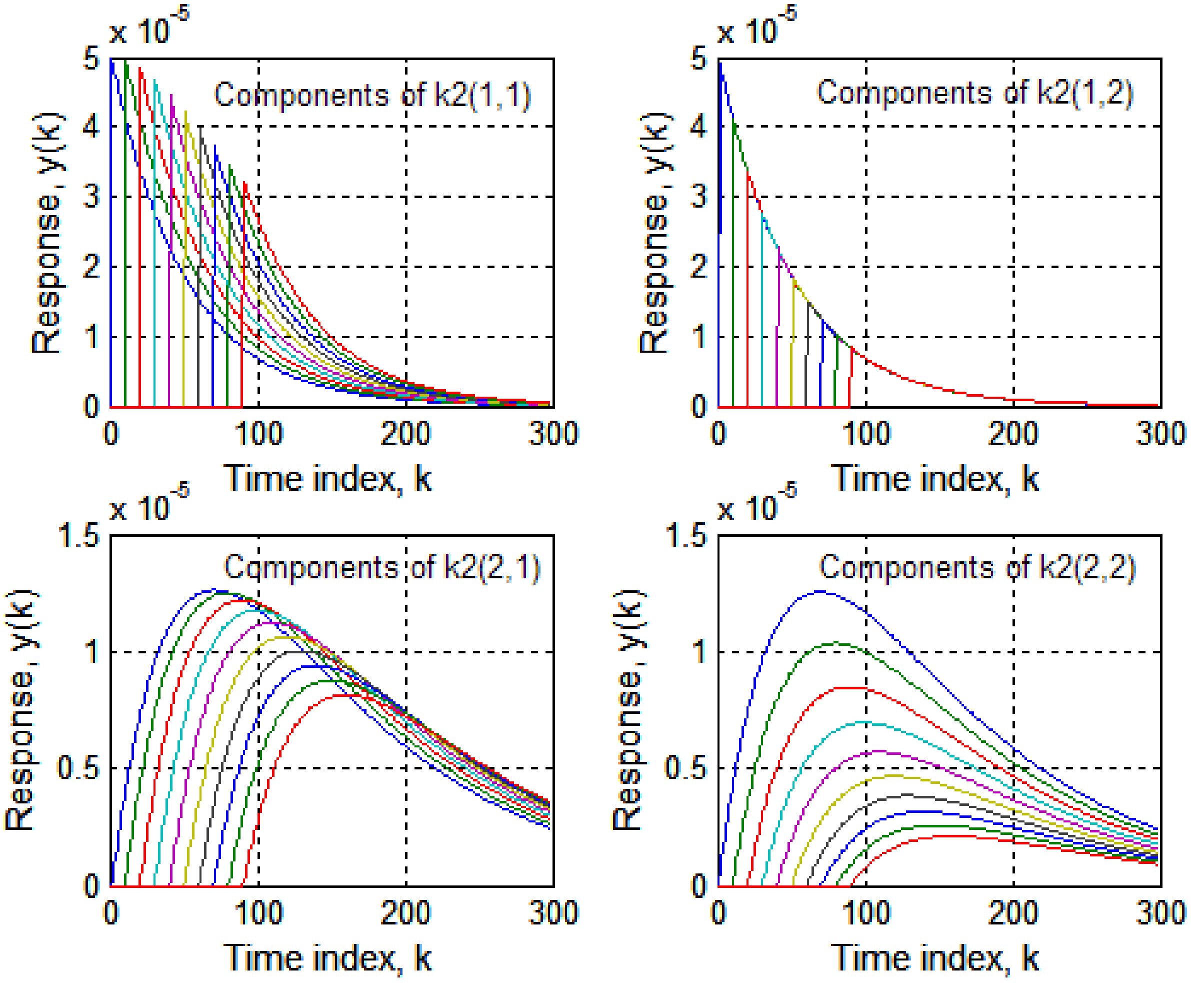


Figure 7.- Discrete second-order kernels calculated using method of present report at 10 values of time lag T ($N = 300$, $\Delta t = 0.01$ sec, $\Delta T = 0.1$ sec, and *ncomp* = 10).

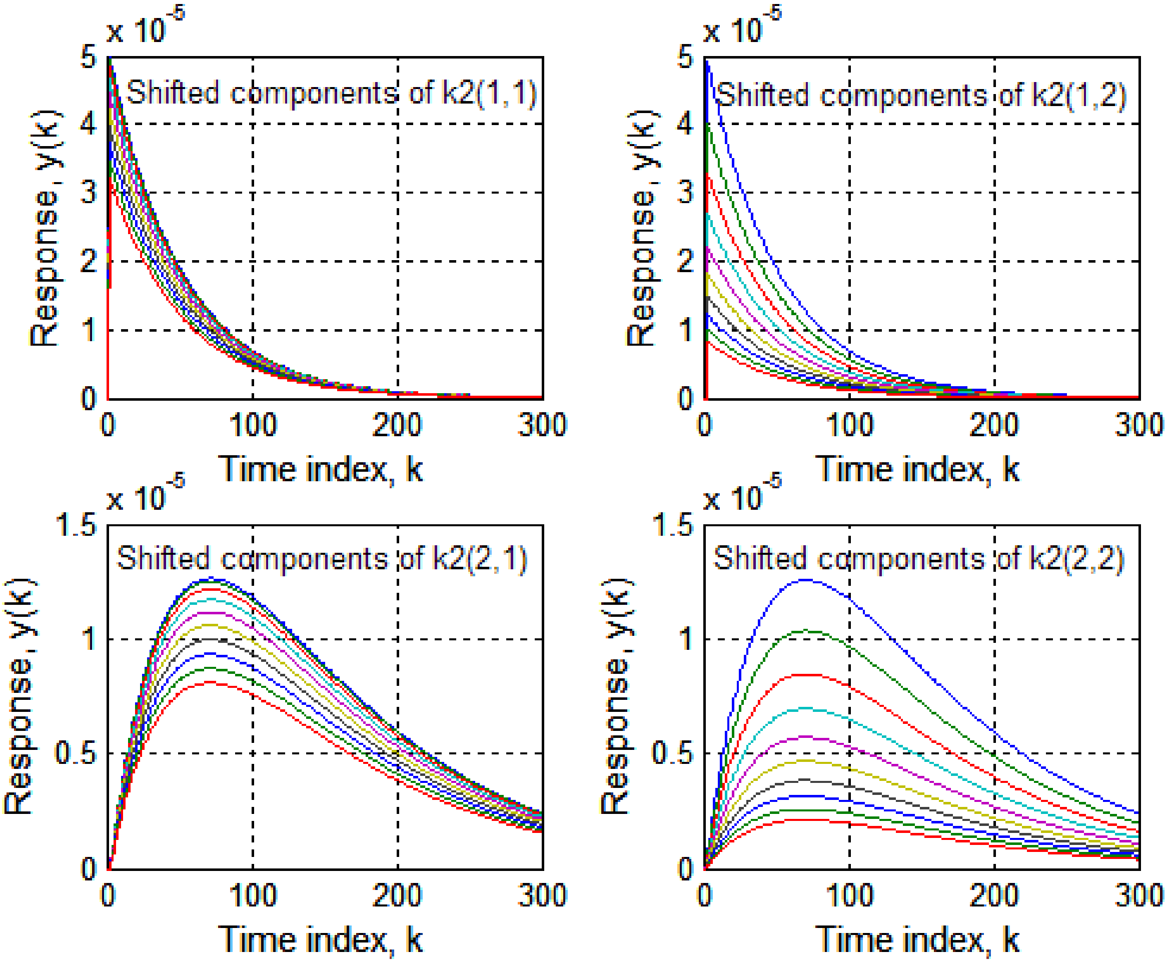


Figure 8.- Components of discrete second-order kernels shifted to time index zero for comparison to the discretized kernels of reference 13 as shown in figure 5 of present report.

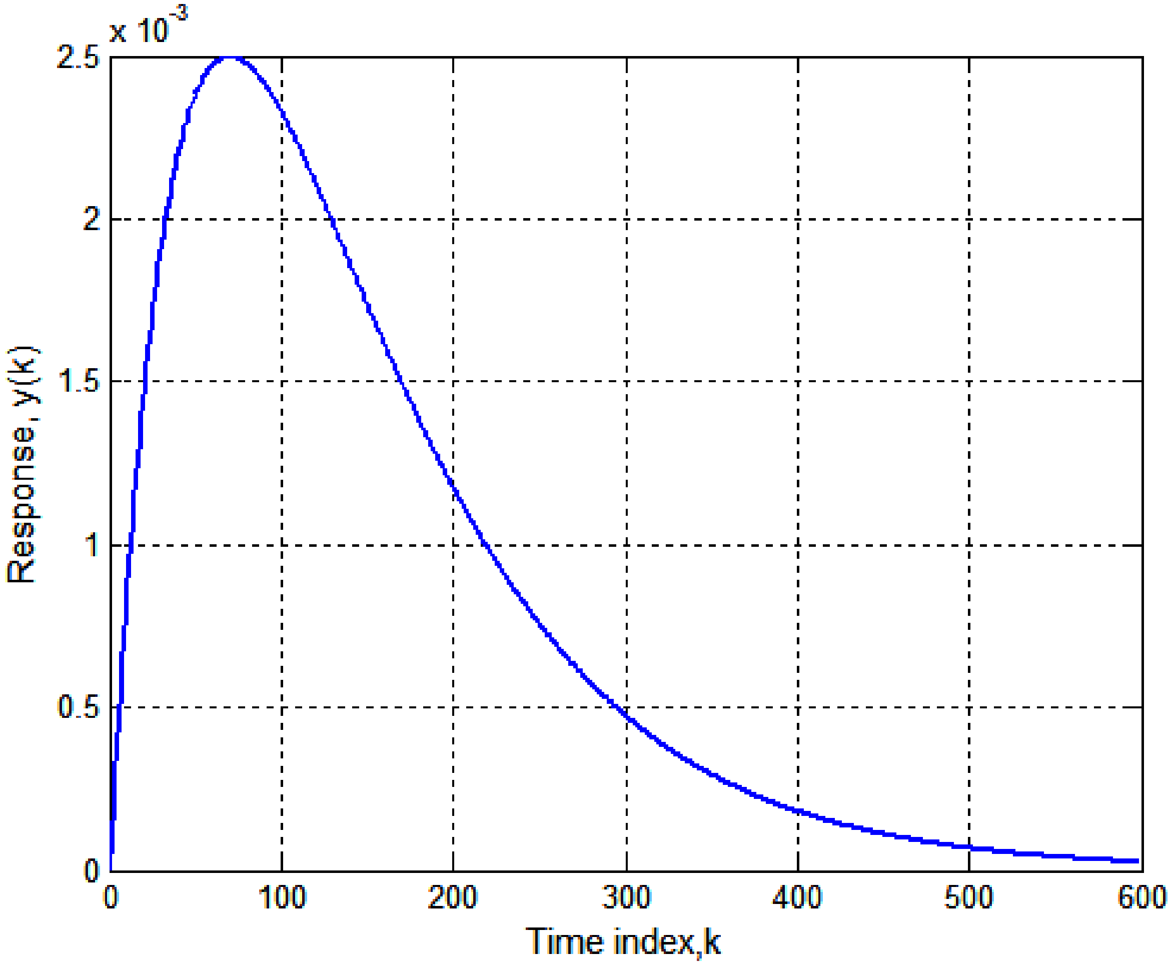


Figure 9.- Discrete first-order kernel calculated using method of present report for bilinear system of Example 1 ($N = 600$, $\Delta t = 0.01$ sec).

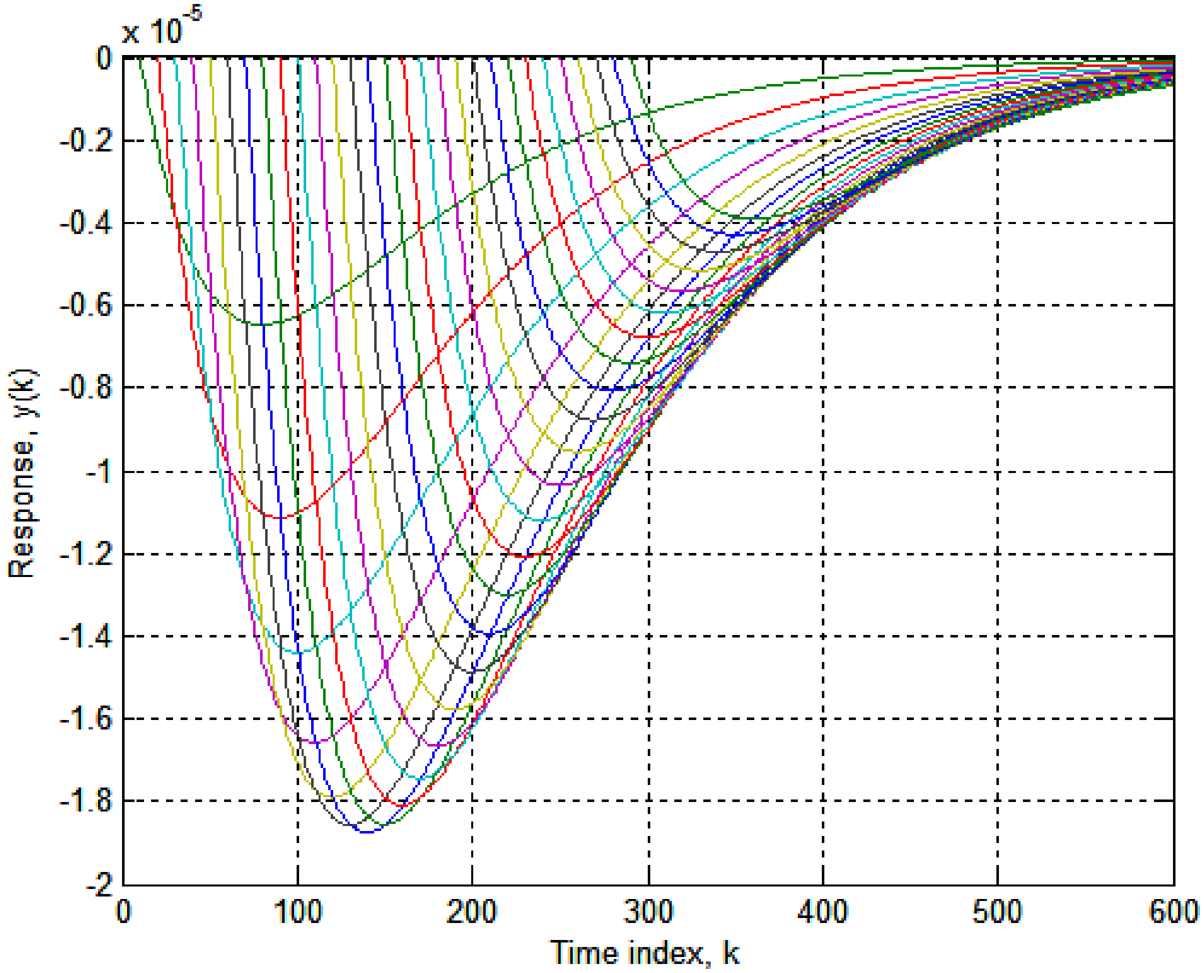


Figure 10.- Discrete second-order kernel calculated using method of present report at 30 values of time lag T for bilinear system of Example 1 ($N = 600$, $\Delta t = 0.01$ sec, $\Delta T = 0.1$ sec, and $ncomp = 30$).

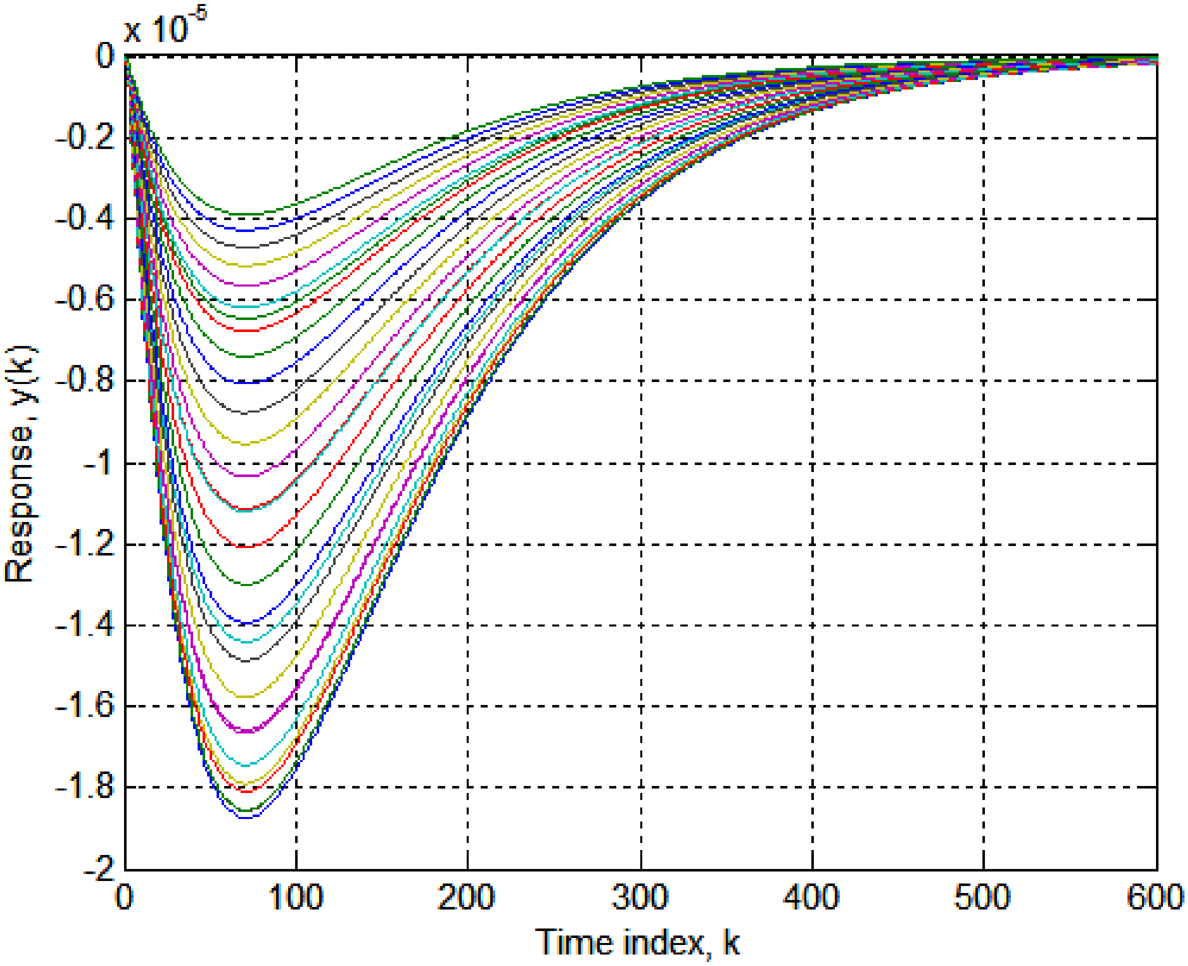


Figure 11.- Components of discrete second-order kernel shifted to time index zero for bilinear system of Example 1 ($N = 600$, $\Delta t = 0.01$, $\Delta T = 0.1$, and *ncomp* = 30).

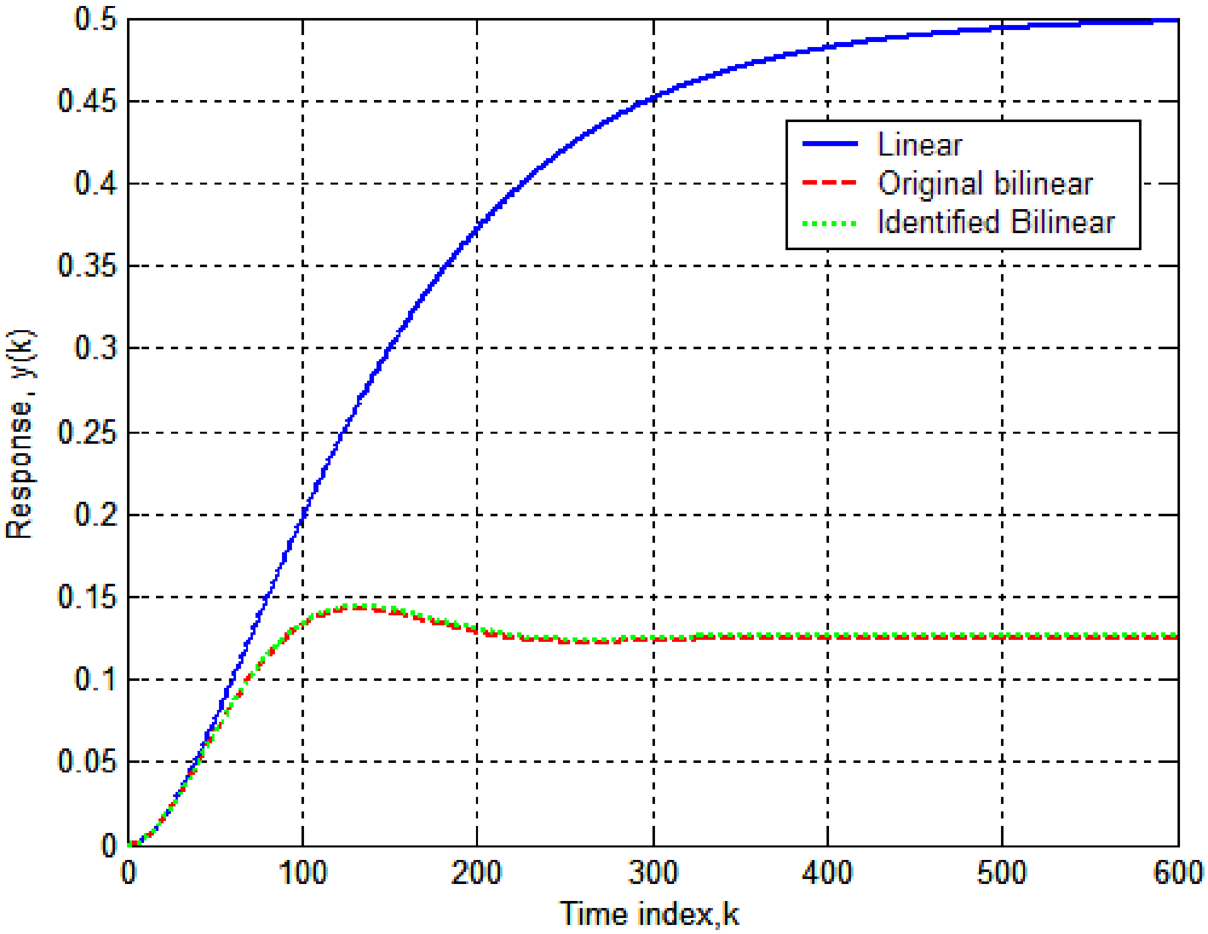


Figure 12.- Response time histories of linear, original bilinear, and identified bilinear equations to unit step input for bilinear system of Example 1.

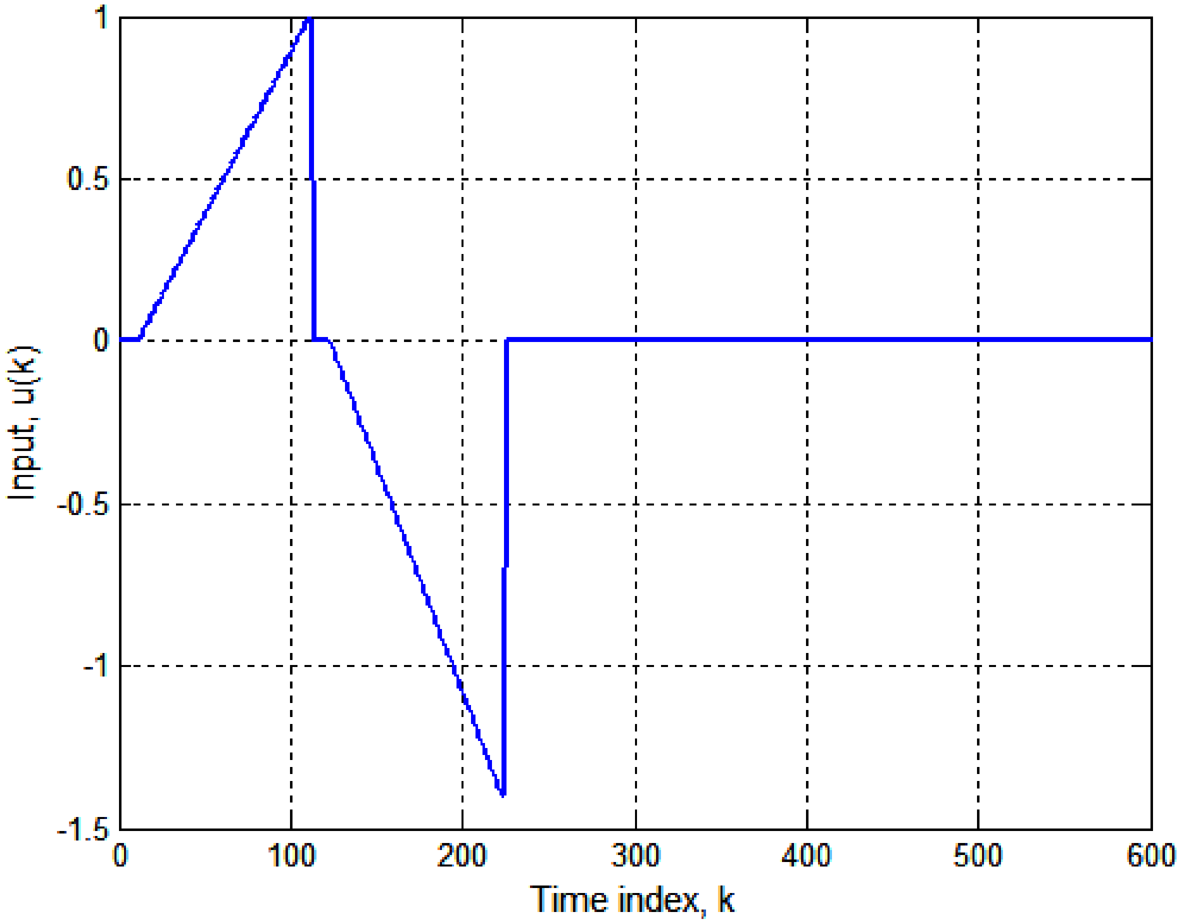


Figure 13.- Double sawtooth input applied to bilinear system of Example 1.

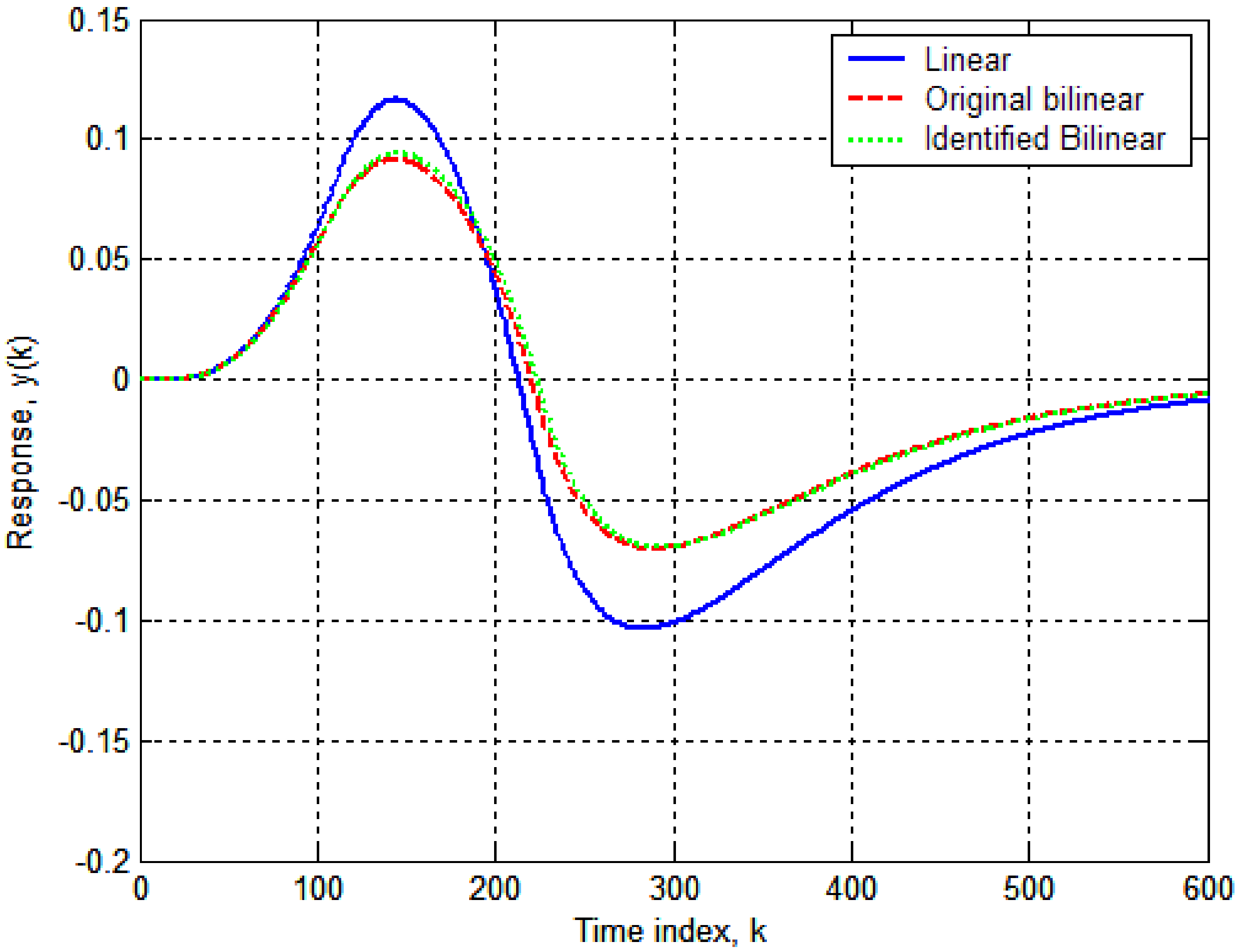


Figure 14.- Response time histories of linear, original bilinear, and identified bilinear equations to double sawtooth input of figure 13 for bilinear system of Example 1.

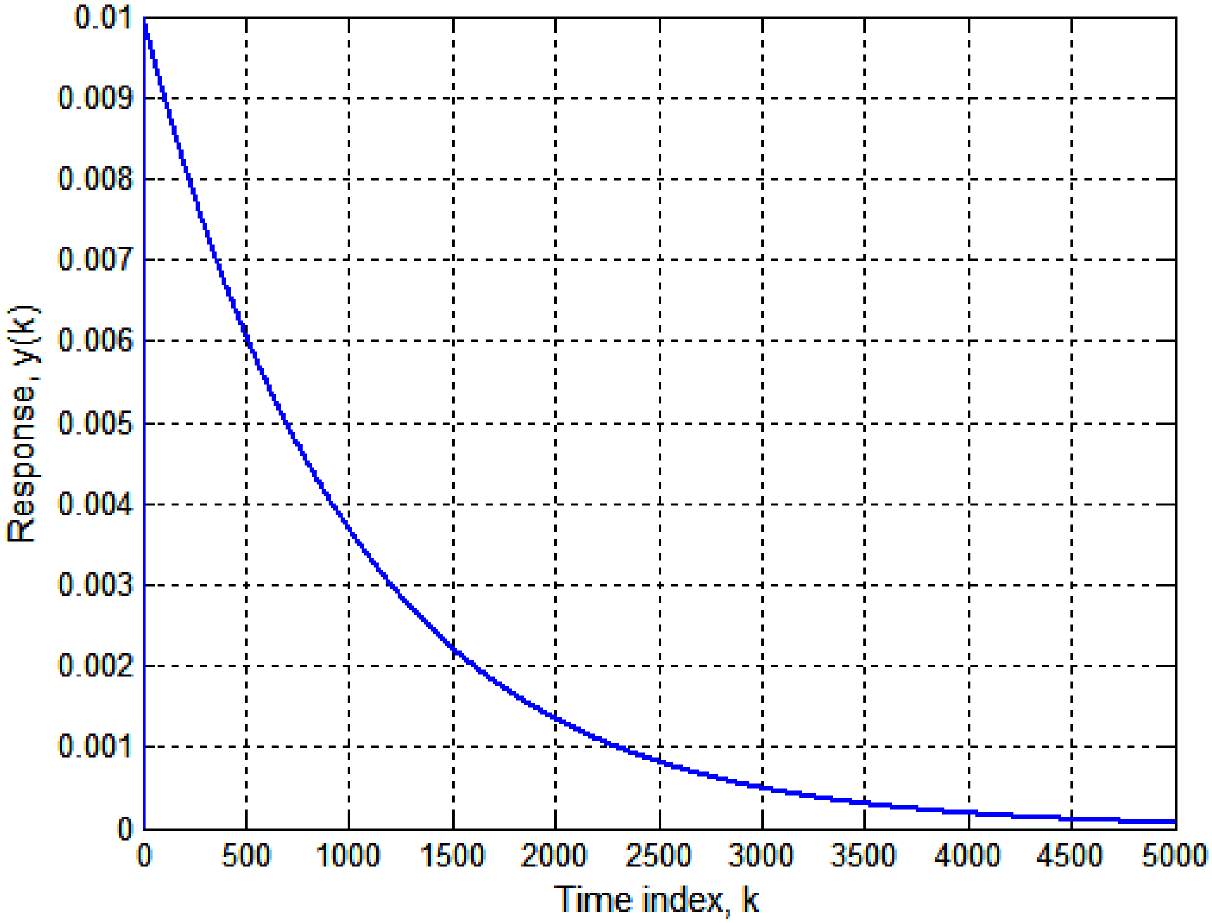


Figure 15.- Discrete first-order kernel calculated for nonlinear Riccati equation of Example 2 ($N = 5000$, $\Delta t = 0.01$ sec).

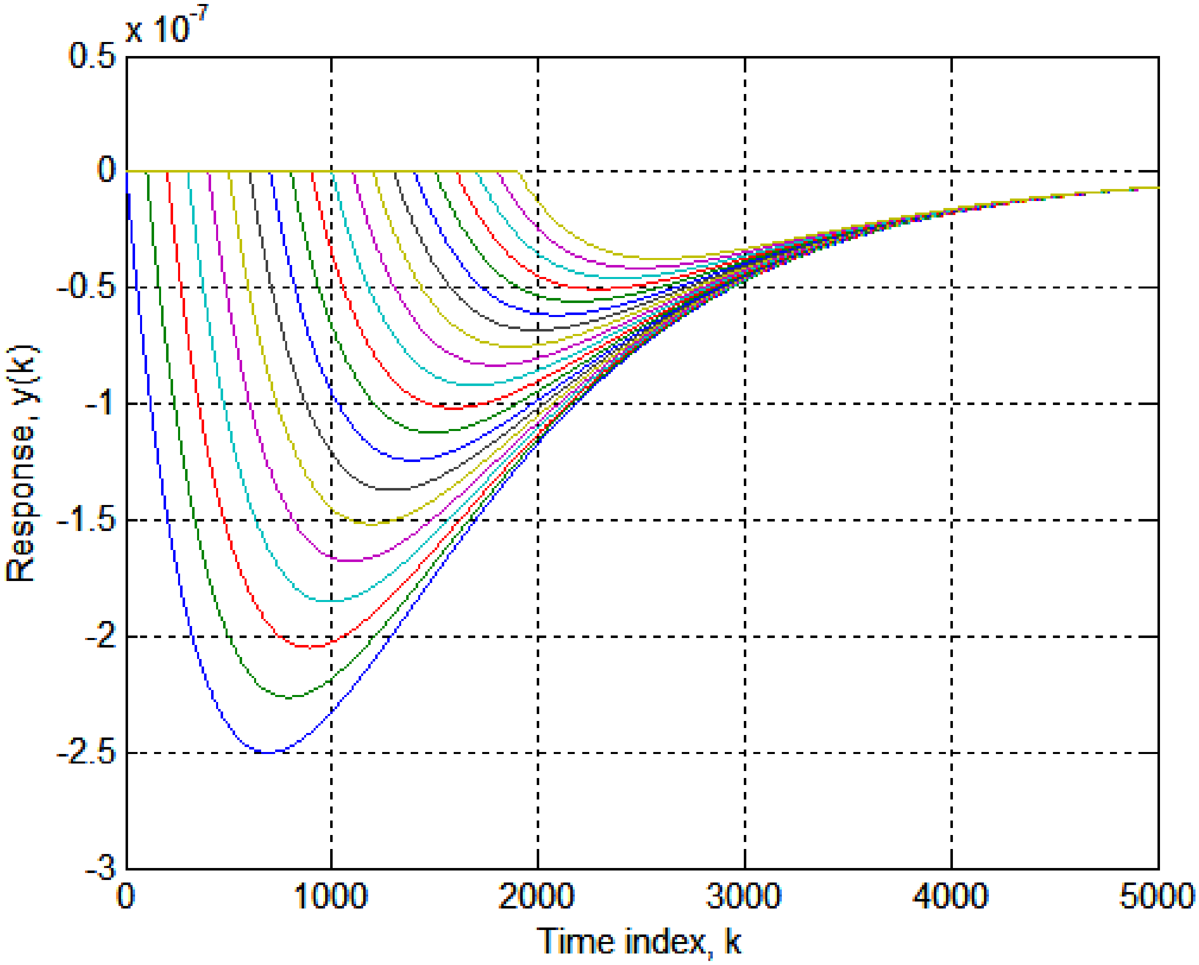


Figure 16.- Discrete second-order kernel calculated at 20 values of time lag *T* for nonlinear Riccati equation of Example 2 ($N = 5000$, $\Delta t = 0.01$ sec, $\Delta T = 0.1$ sec, and $ncomp = 20$).

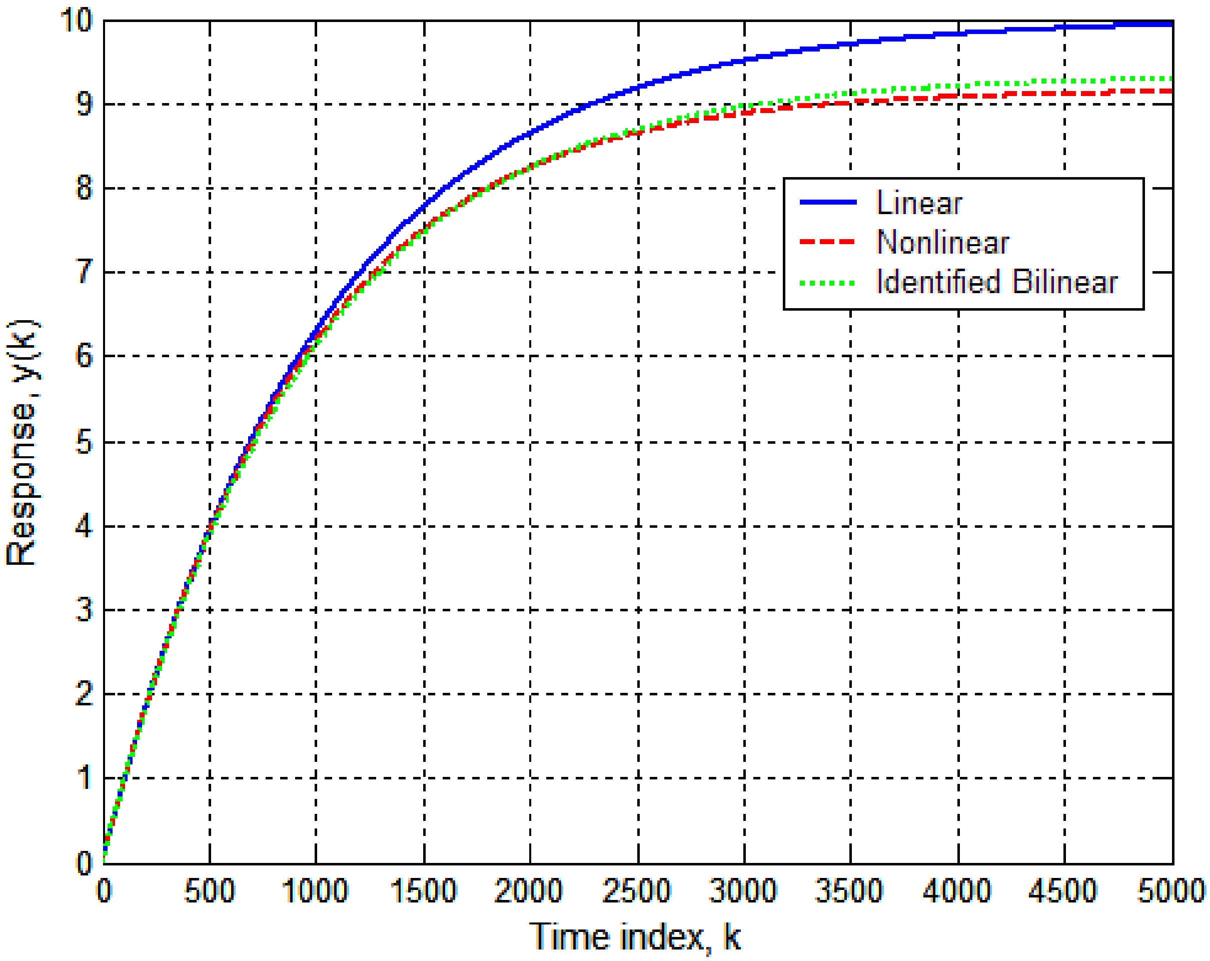


Figure 17.- Computed response time histories for unit step excitation of linearized Riccati equation, nonlinear Riccati equation, and identified bilinear equation of Example 2.

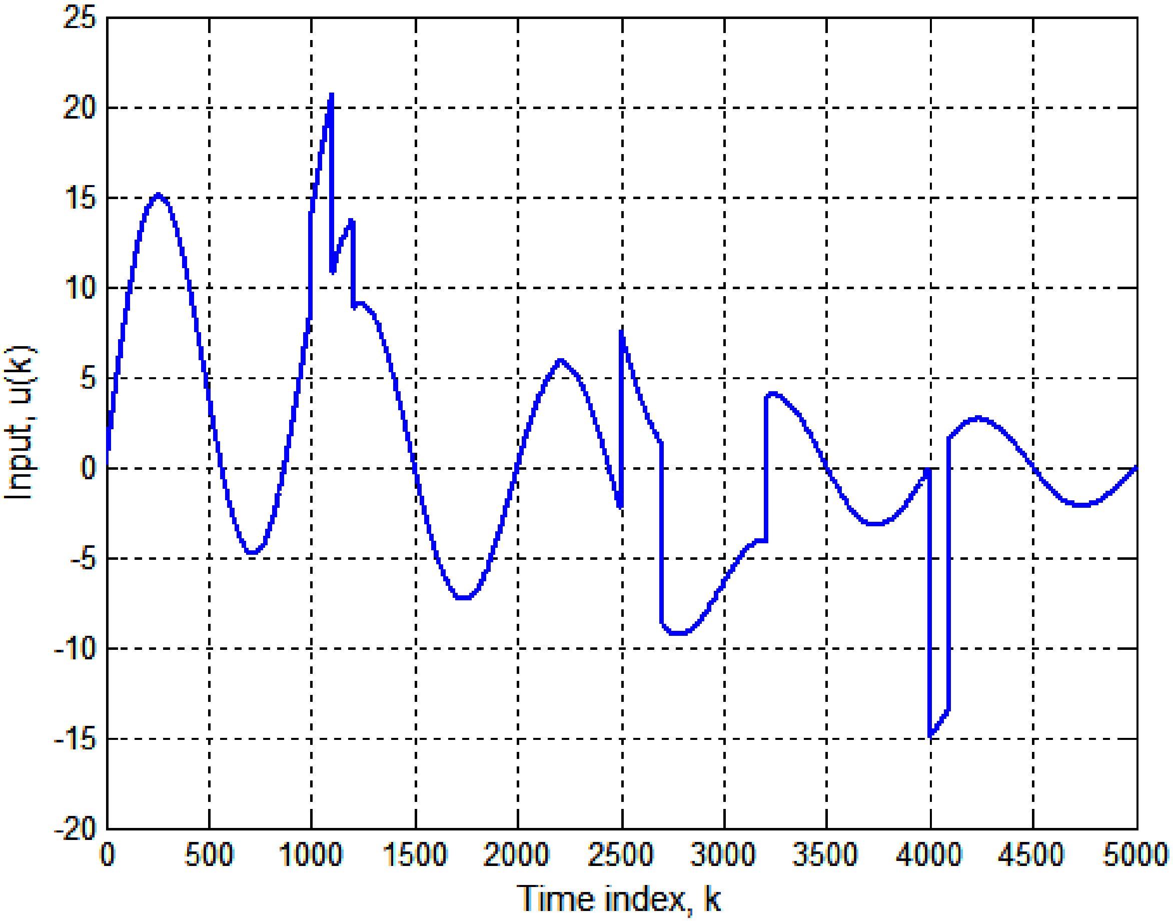


Figure 18.- Composite input applied to Riccati equation of Example 2.

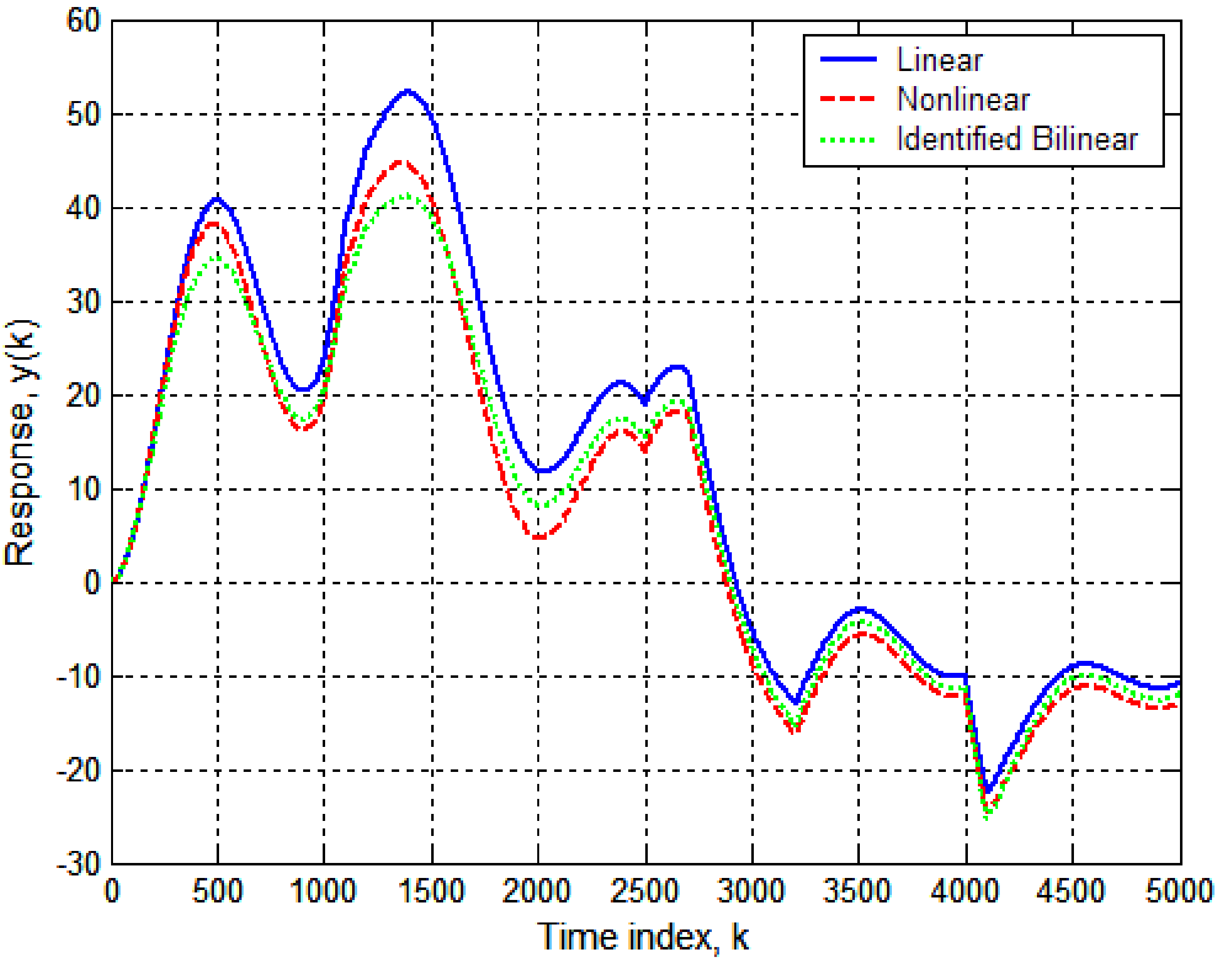


Figure 19.- Response time histories of linearized Riccati equation, nonlinear Riccati equation, and identified bilinear equation for Example 2 to composite input of figure 18.

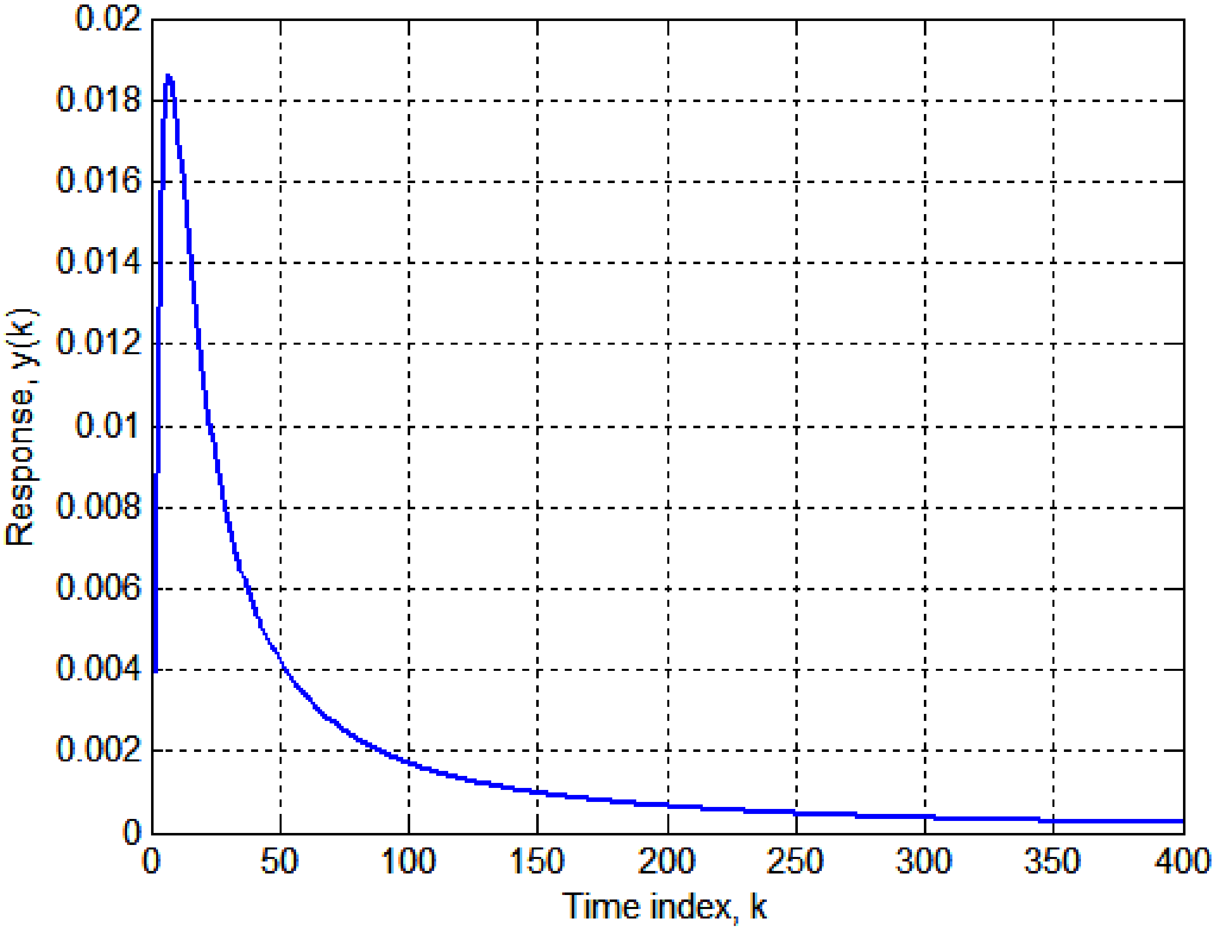


Figure 20.- Discrete first-order kernel calculated for nonlinear Burgers equation of Example 3 ($N = 400$, $\Delta t = 0.01$ sec).

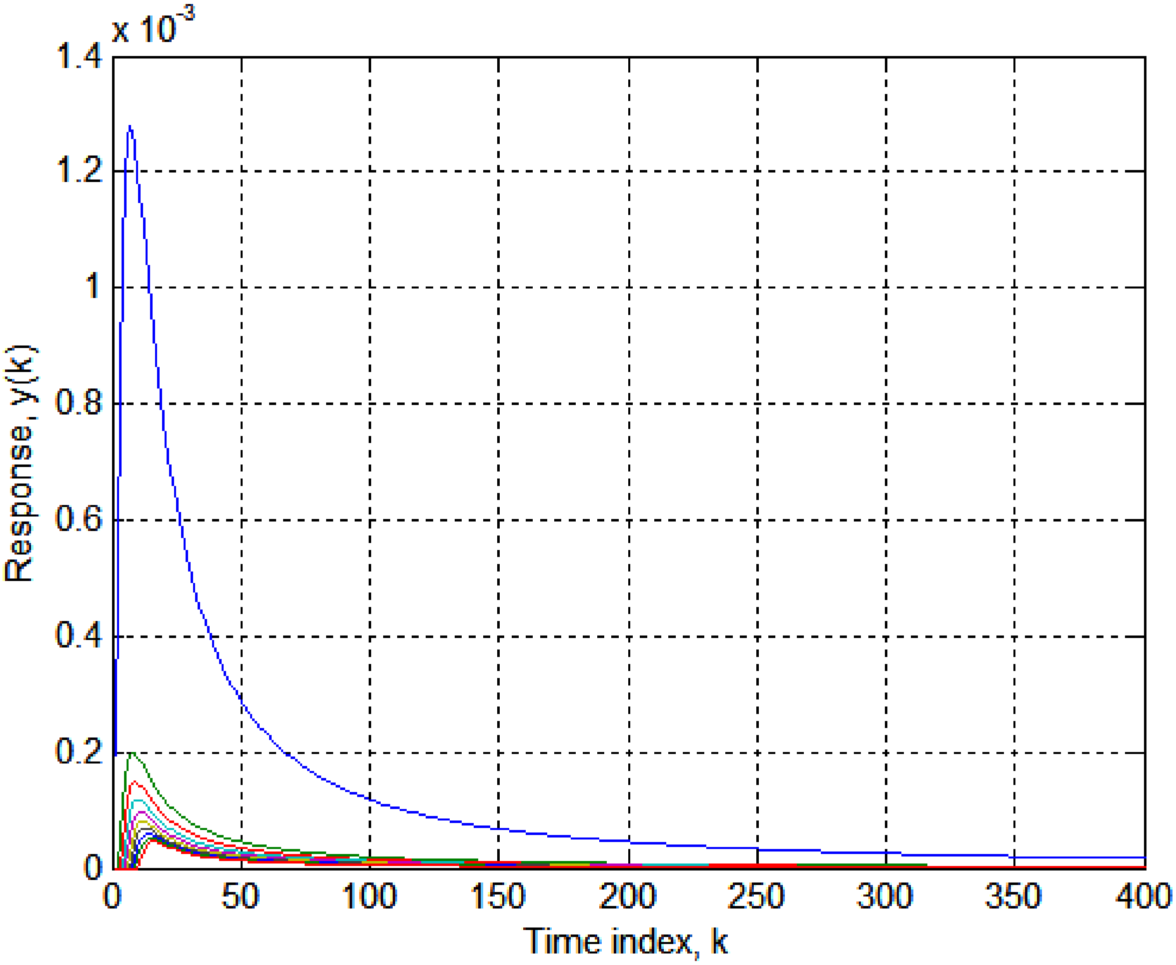


Figure 21.- Components of discrete second-order kernel calculated at 10 values of time lag *T* for nonlinear Burgers equation of Example 3 ($N = 400$, $\Delta x = 0.2$, $\Delta t = 0.01$ sec, $\Delta T = 0.01$ sec, and *ncomp* = 10).

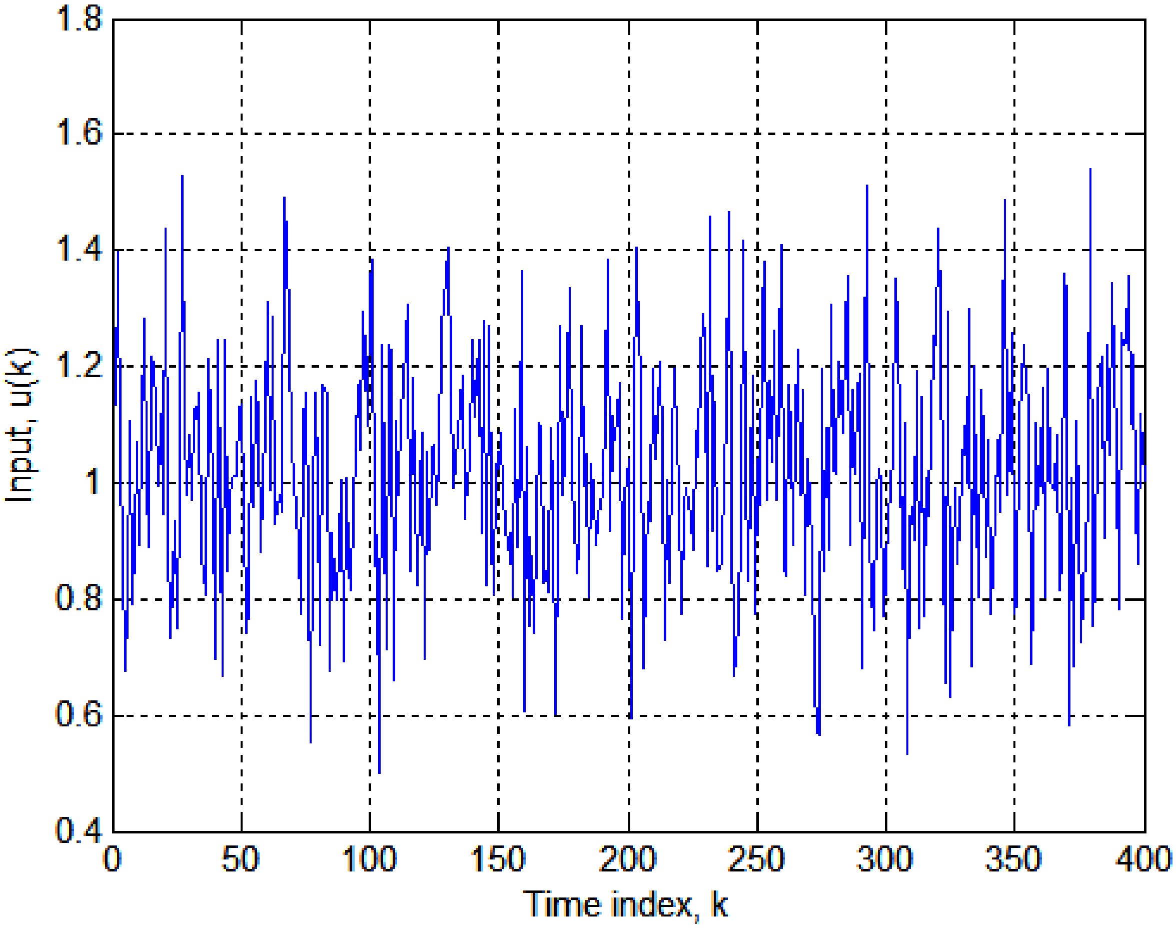


Figure 22.- Noisy unit step input applied to nonlinear Burgers equation of Example 3.

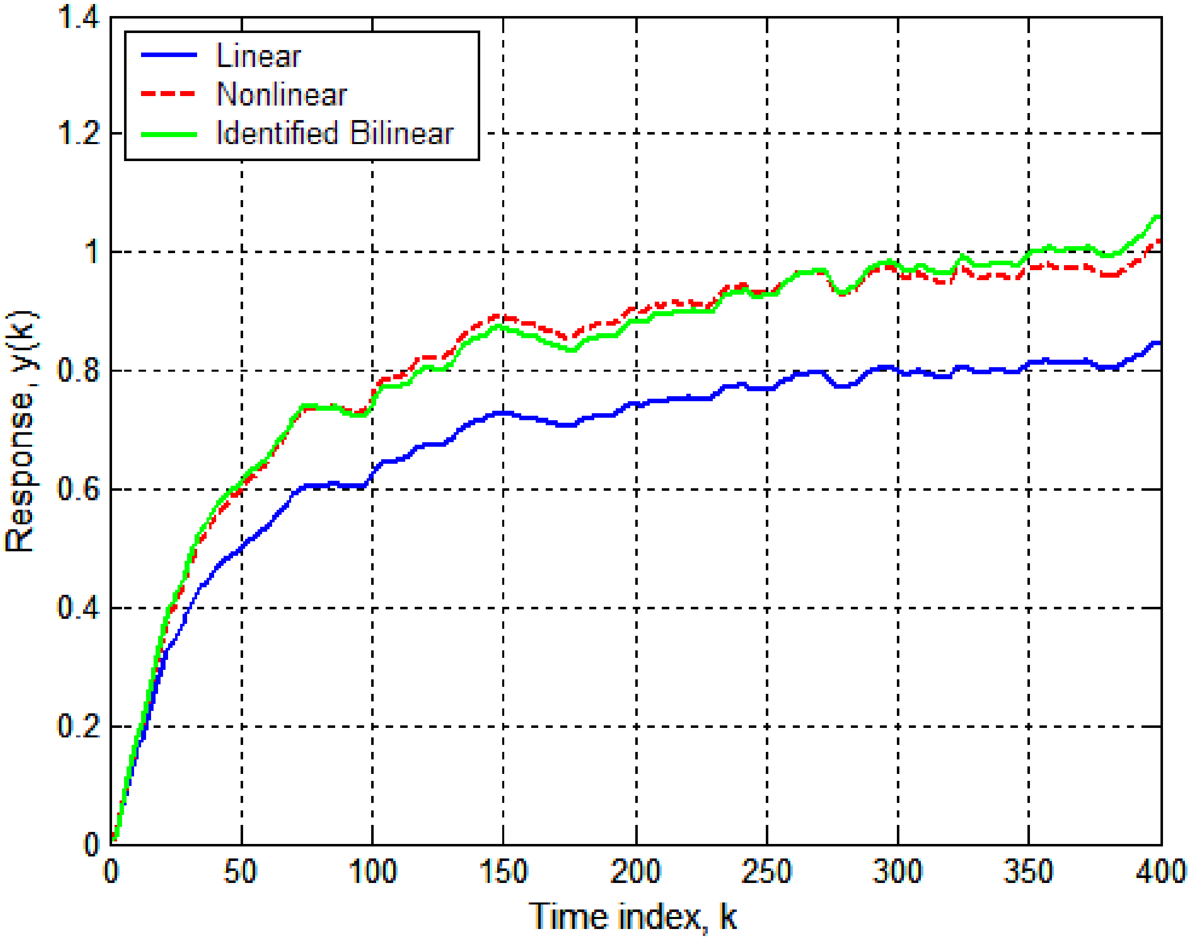


Figure 23.- Linear, nonlinear, and identified bilinear responses of Burgers equation to noisy unit step input of figure 22.

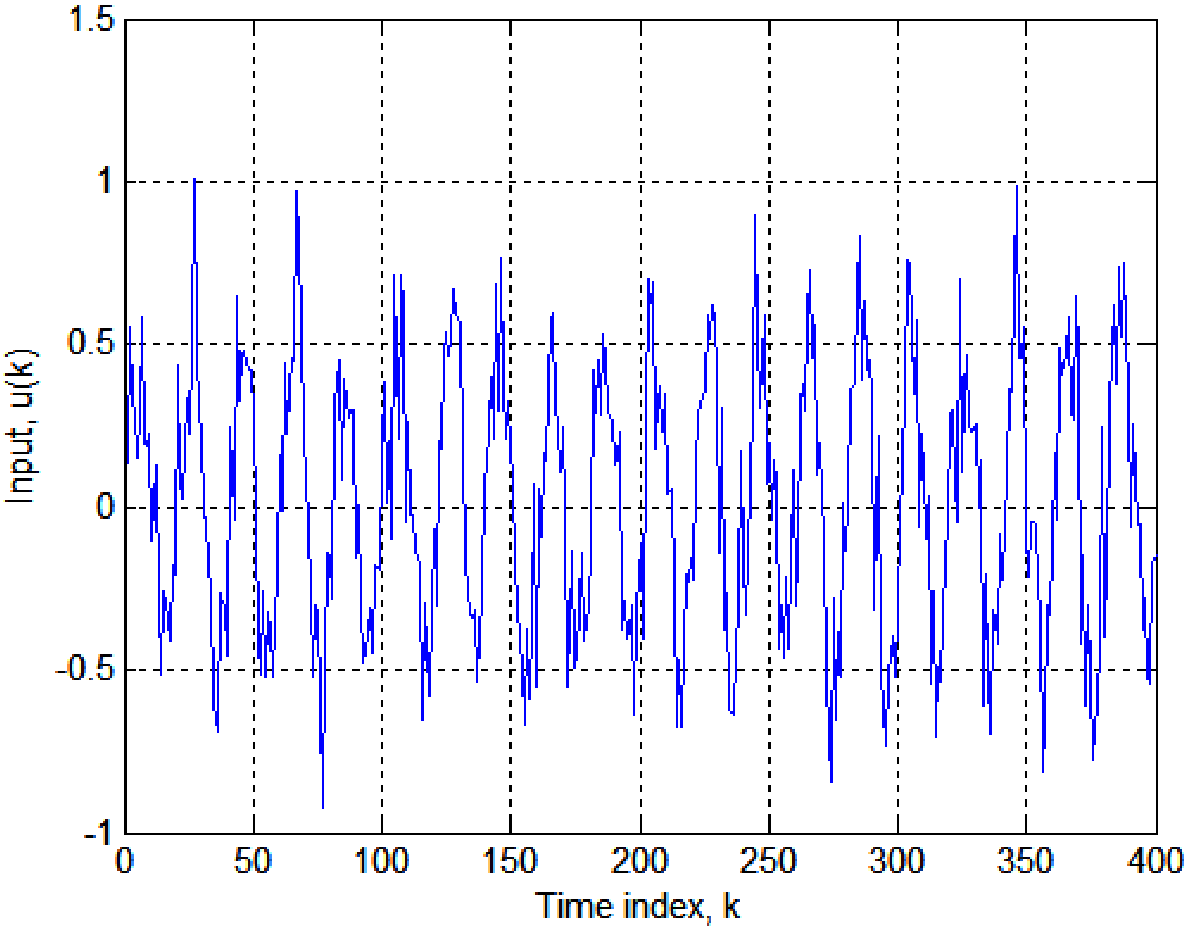


Figure 24.- Noisy five-hertz sine wave applied to nonlinear Burgers equation of Example 3.

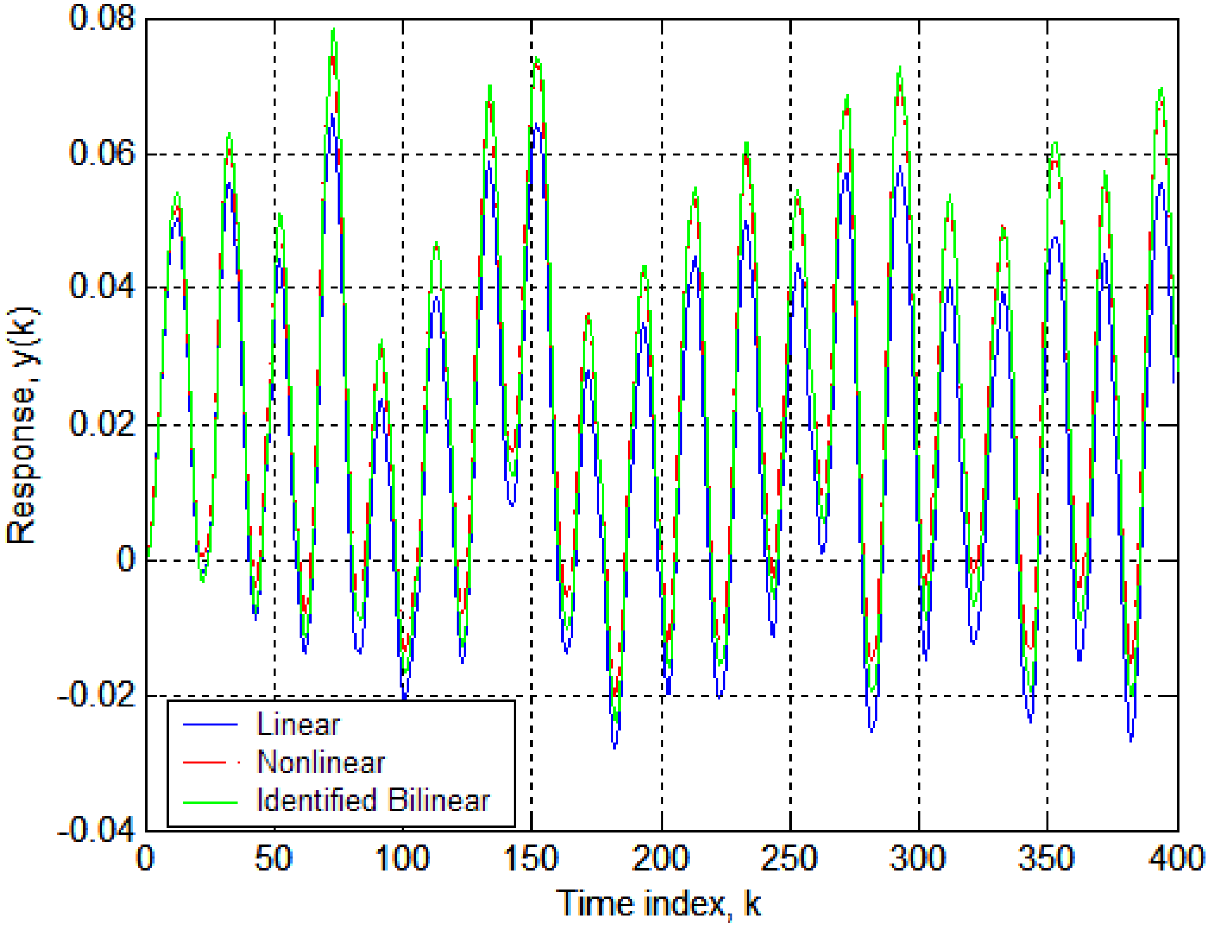


Figure 25.- Linear, nonlinear, and identified bilinear equation responses to noisy sine wave of figure 24.

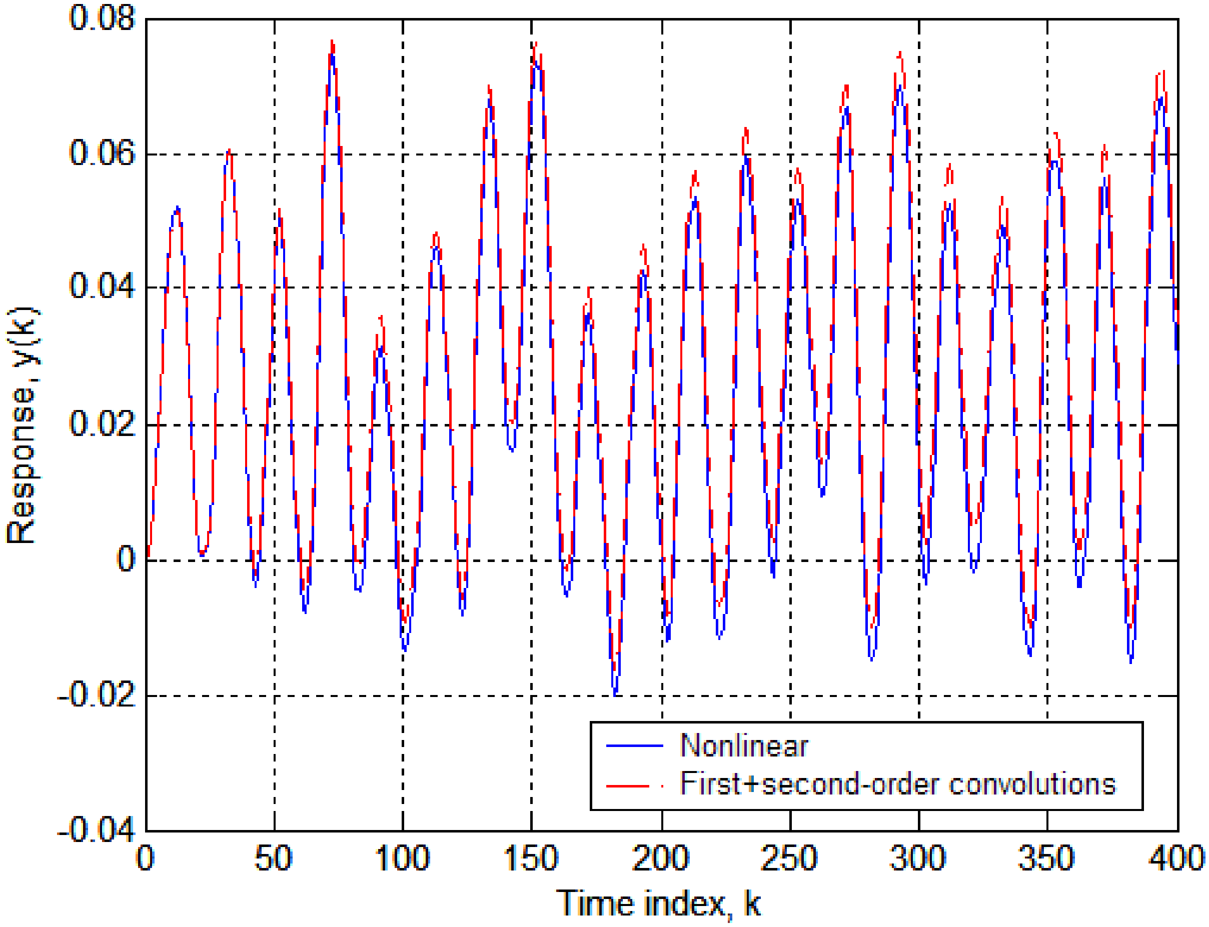


Figure 26.- Nonlinear and first- plus second-order convolution responses of Burgers equation to noisy sine wave of figure 24.

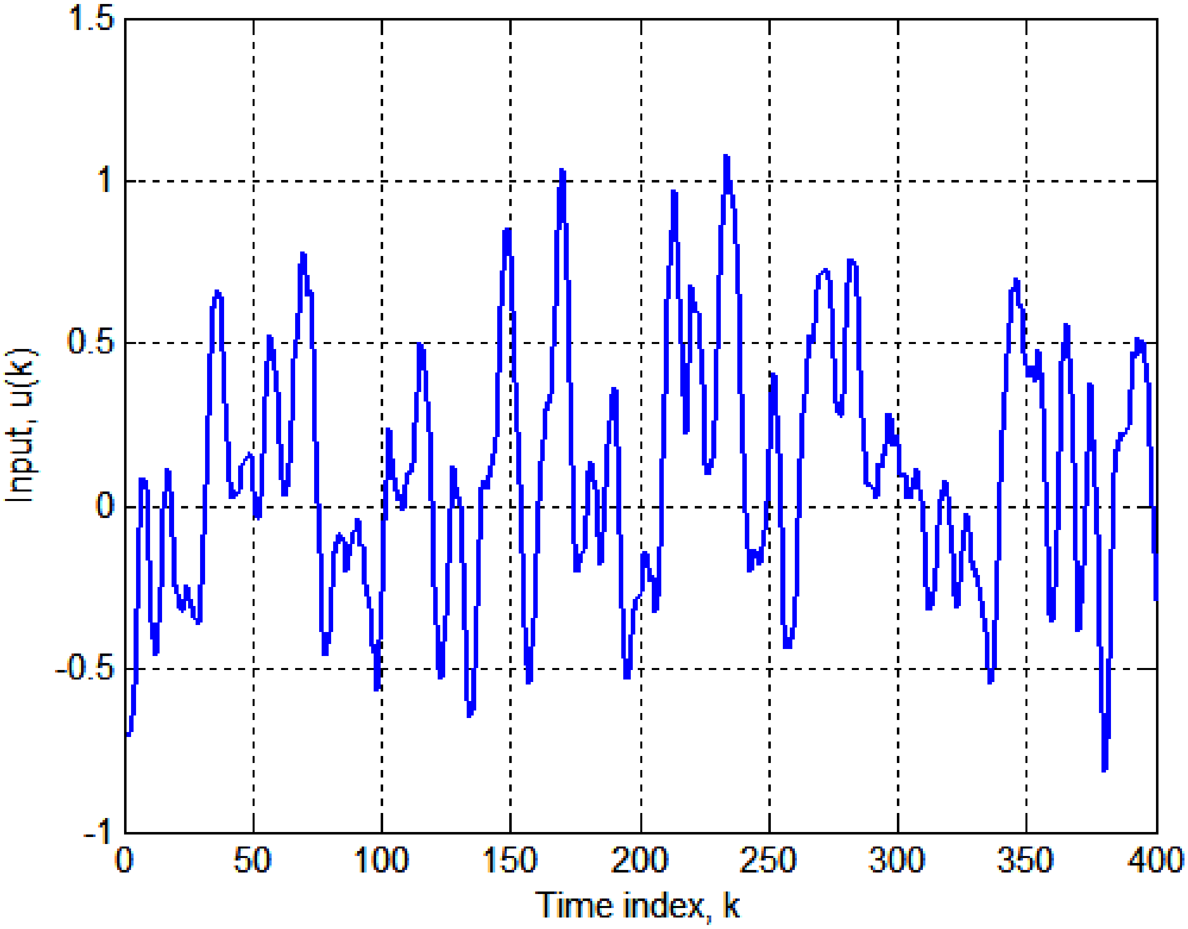


Figure 27.- Random input applied to Burgers equation of Example 3.

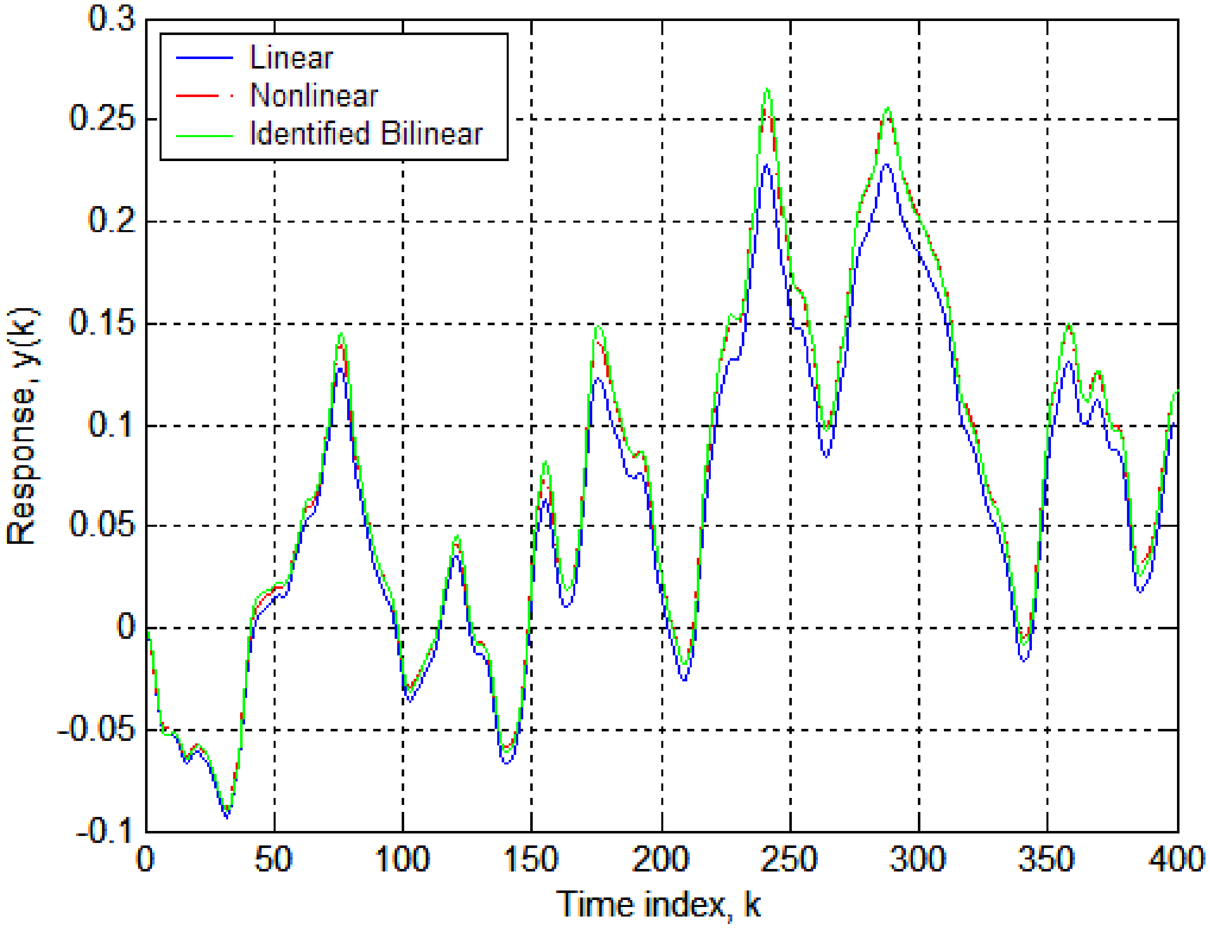


Figure 28.- Linear, nonlinear, and identified bilinear responses of Burgers equation to random input of figure 27.

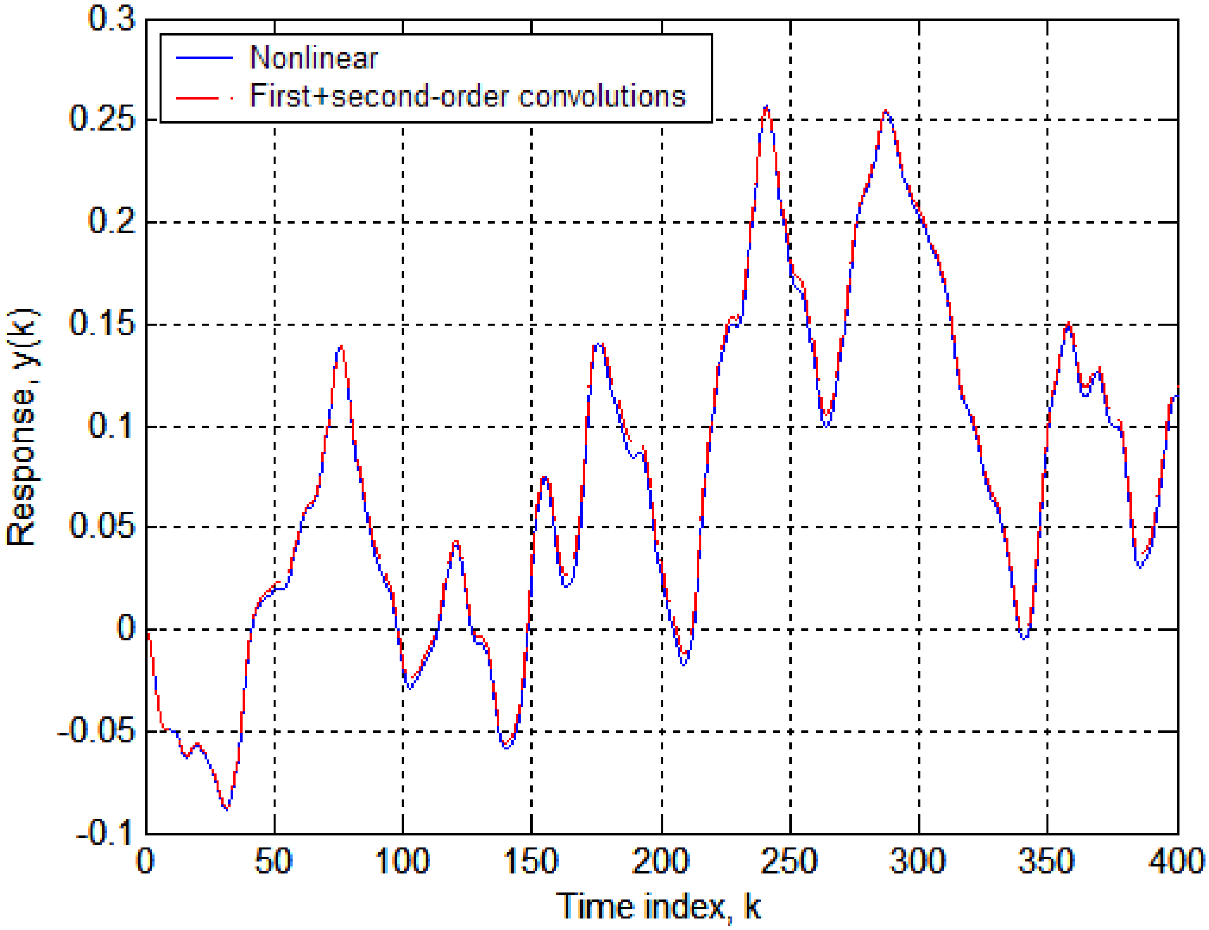


Figure 29.- Nonlinear and first- plus second-order convolution responses of Burgers equation to random input of figure 27.

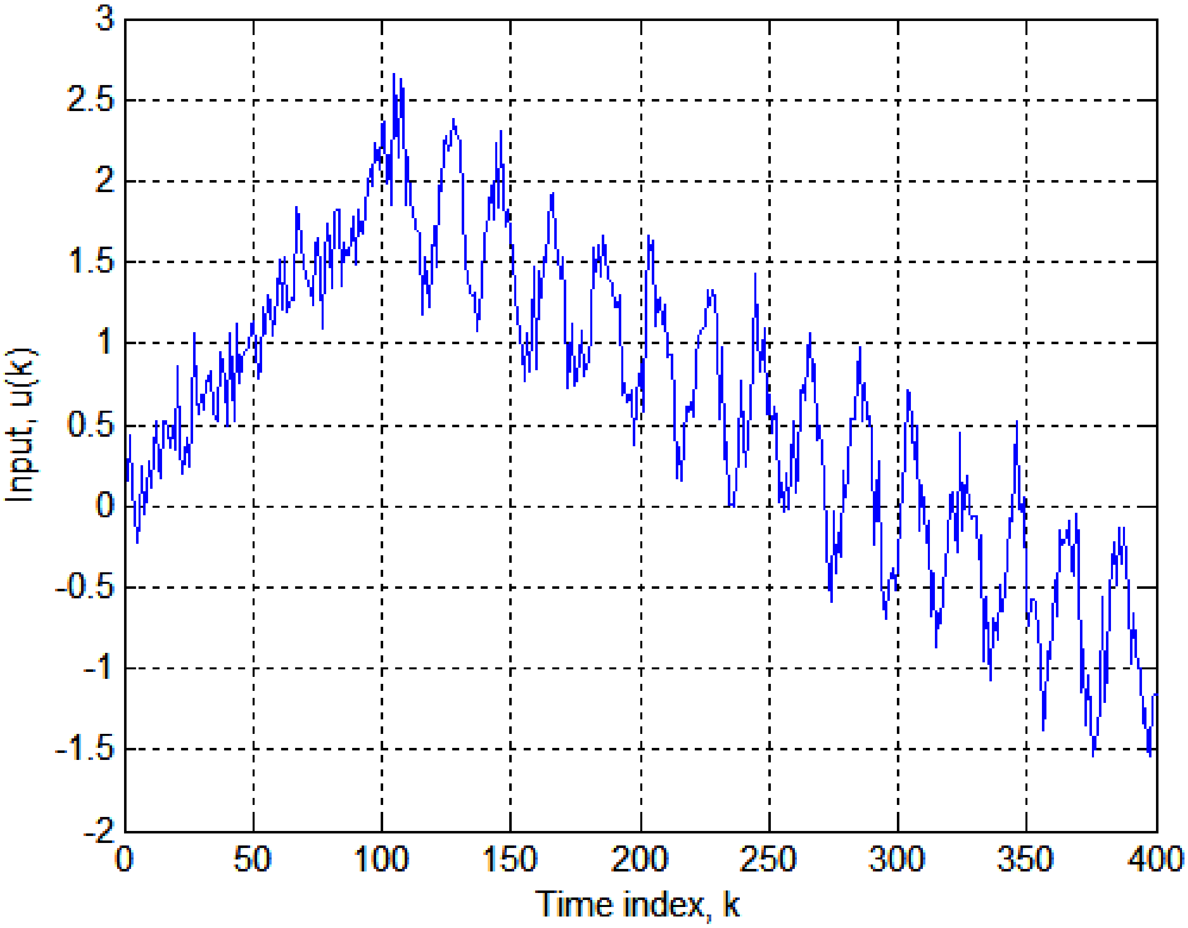


Figure 30.- Noisy/sinusoidal ramp excitations applied to Burgers equation of Example 3.

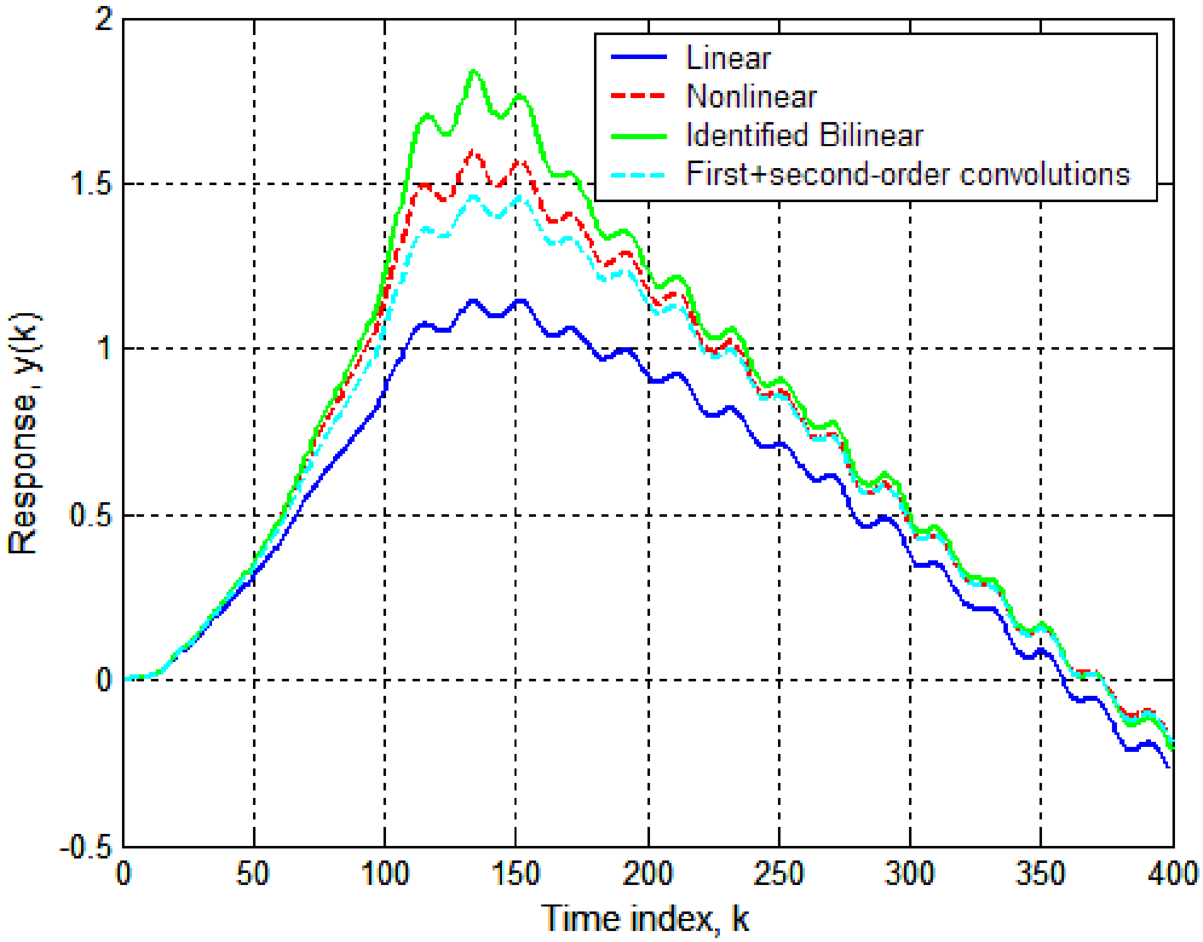


Figure 31.- Linear, nonlinear, identified bilinear, and convolution responses of Burgers equation to noisy/sinusoidal ramp input of figure 30.

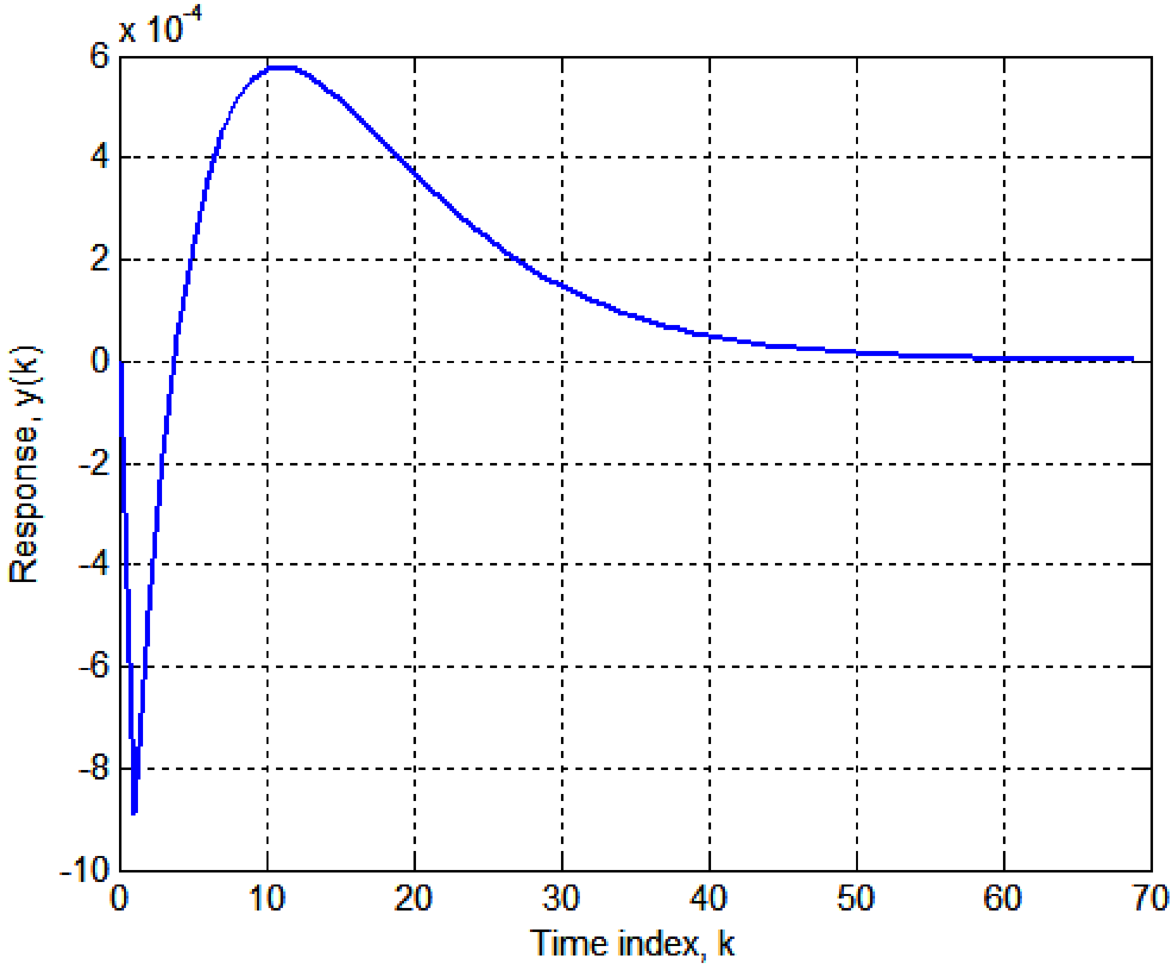


Figure 32.- Discrete first-order kernel for Continuous Stirred Tank Reactor (CSTR) problem of Example 4 ($N = 250$, $\Delta t = 0.001$ hr; first 70 time points plotted).

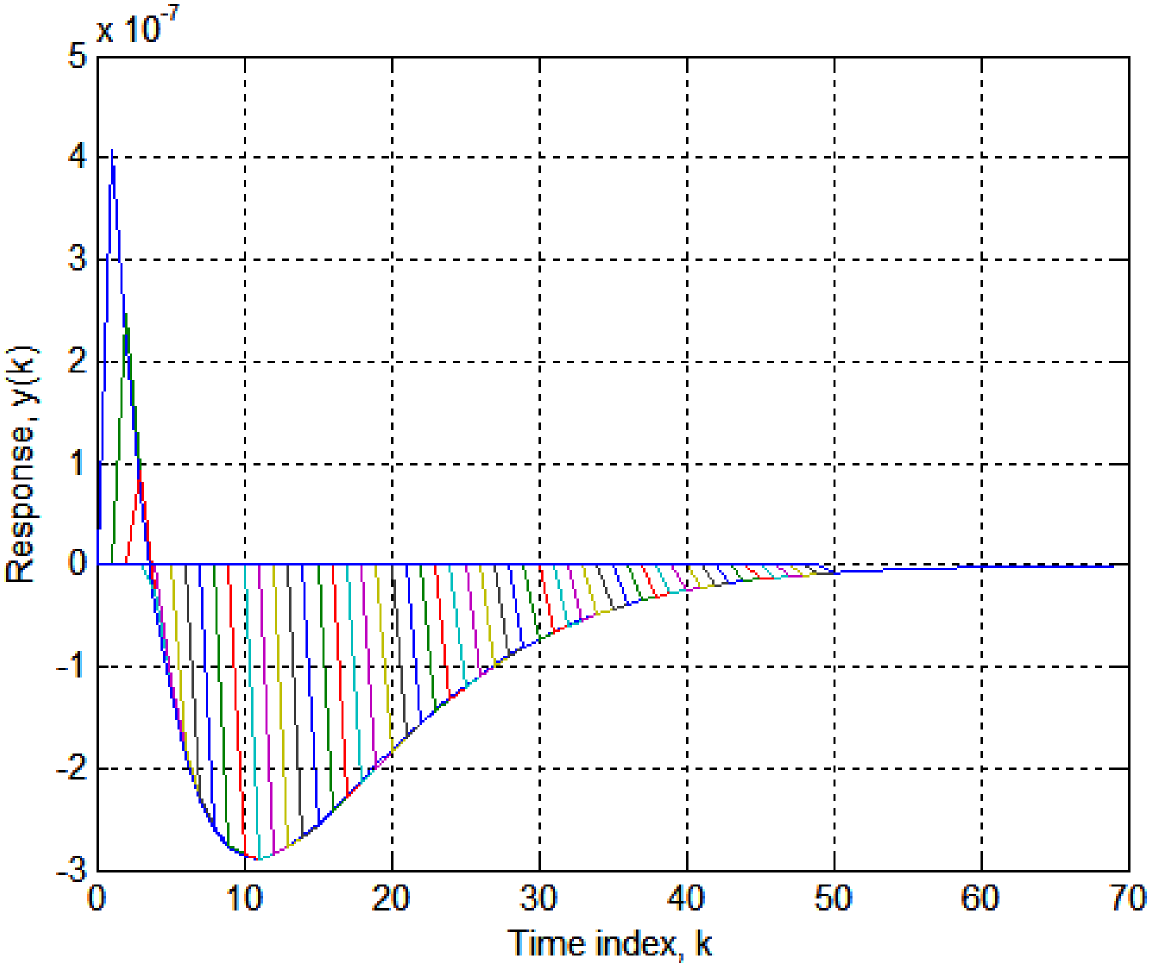


Figure 33.- Components of discrete second-order kernel calculated at 50 values of time lag *T* for CSTR problem of Example 4 ($N = 250$, $\Delta t = 0.001$ hr, $\Delta T = 0.001$ hr, and *ncomp* = 50; first 70 time points plotted).

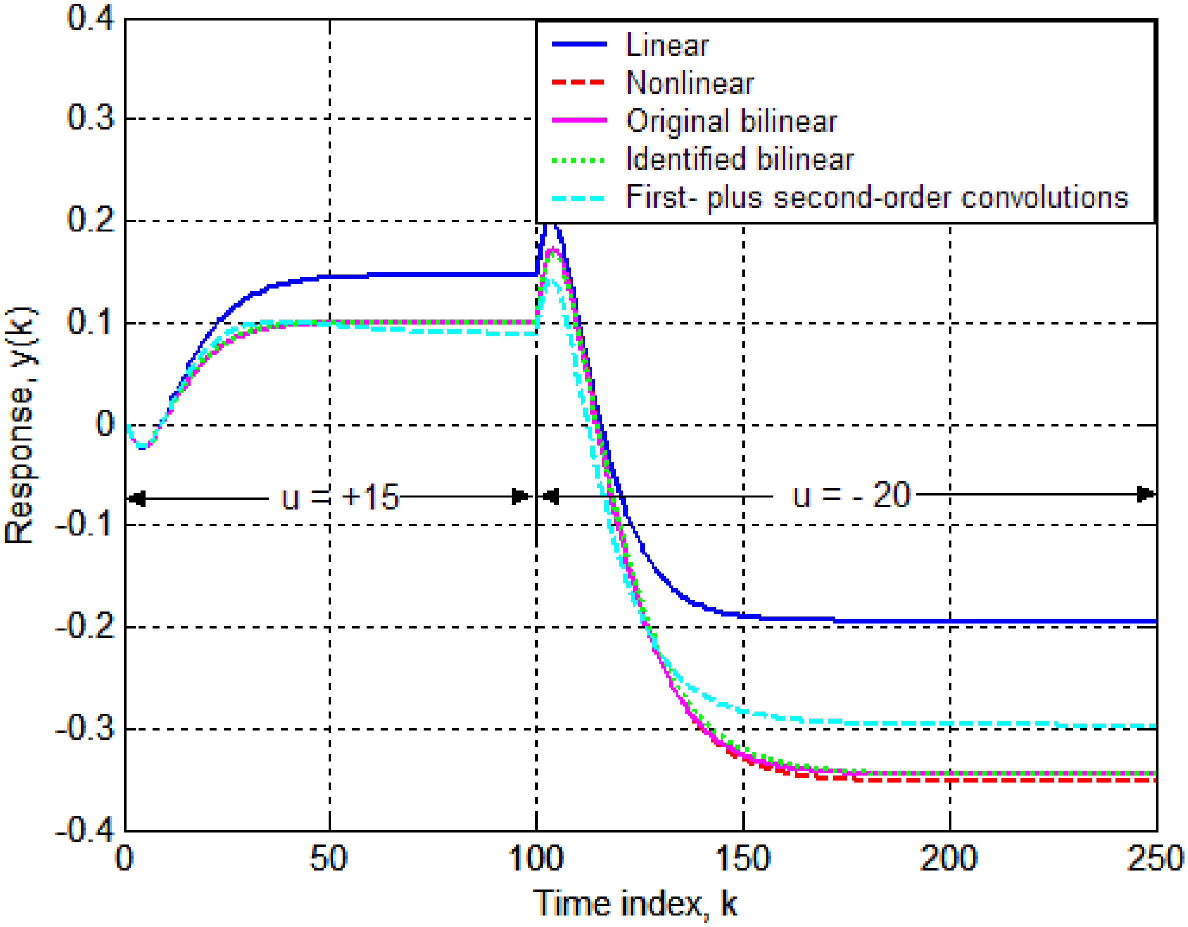


Figure 34.- Responses of linear, nonlinear, original bilinear, identified bilinear, and convolution models for CSTR of Example 4 to piecewise step excitations.

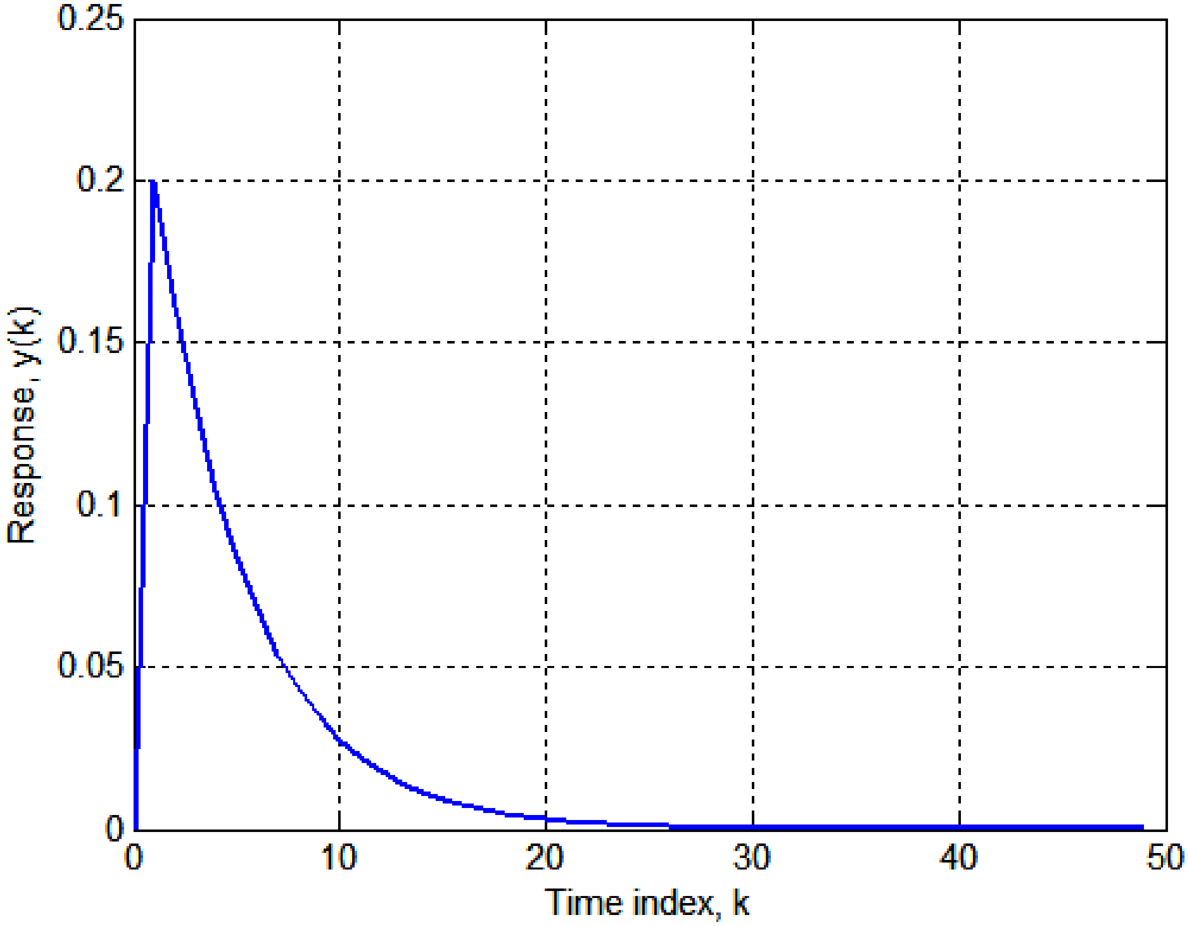


Figure 35.- First-order kernel for nonlinear difference equation of Example 5
(N = 250, Δt = 1 sec; first 50 time points plotted).

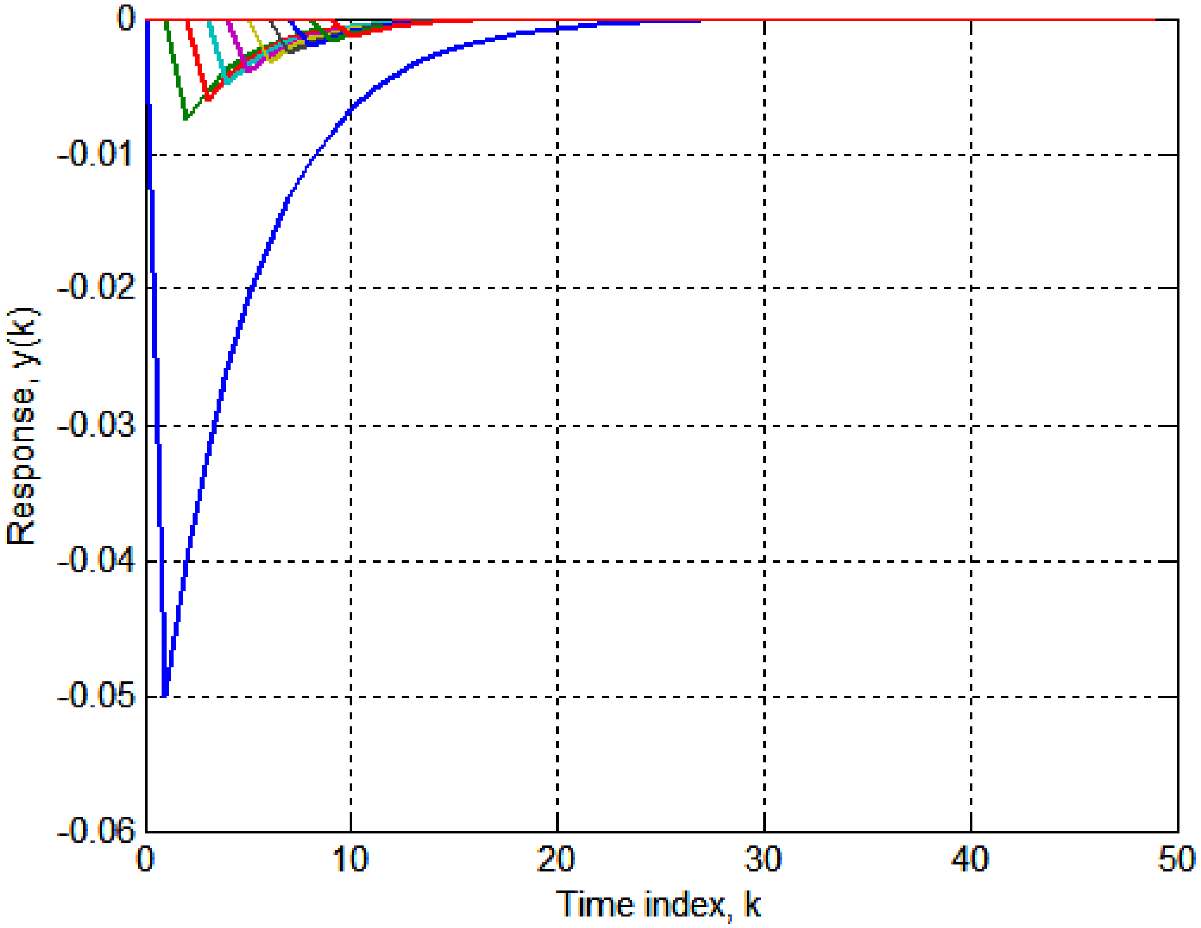


Figure 36.- Components of discrete second-order kernel computed at 10 values of time lag *T* for nonlinear difference equation of Example 5 (*N* = 250, Δ*t* = 1 sec, Δ*T* = 1 sec, and *ncomp* = 10; first 50 time points plotted).

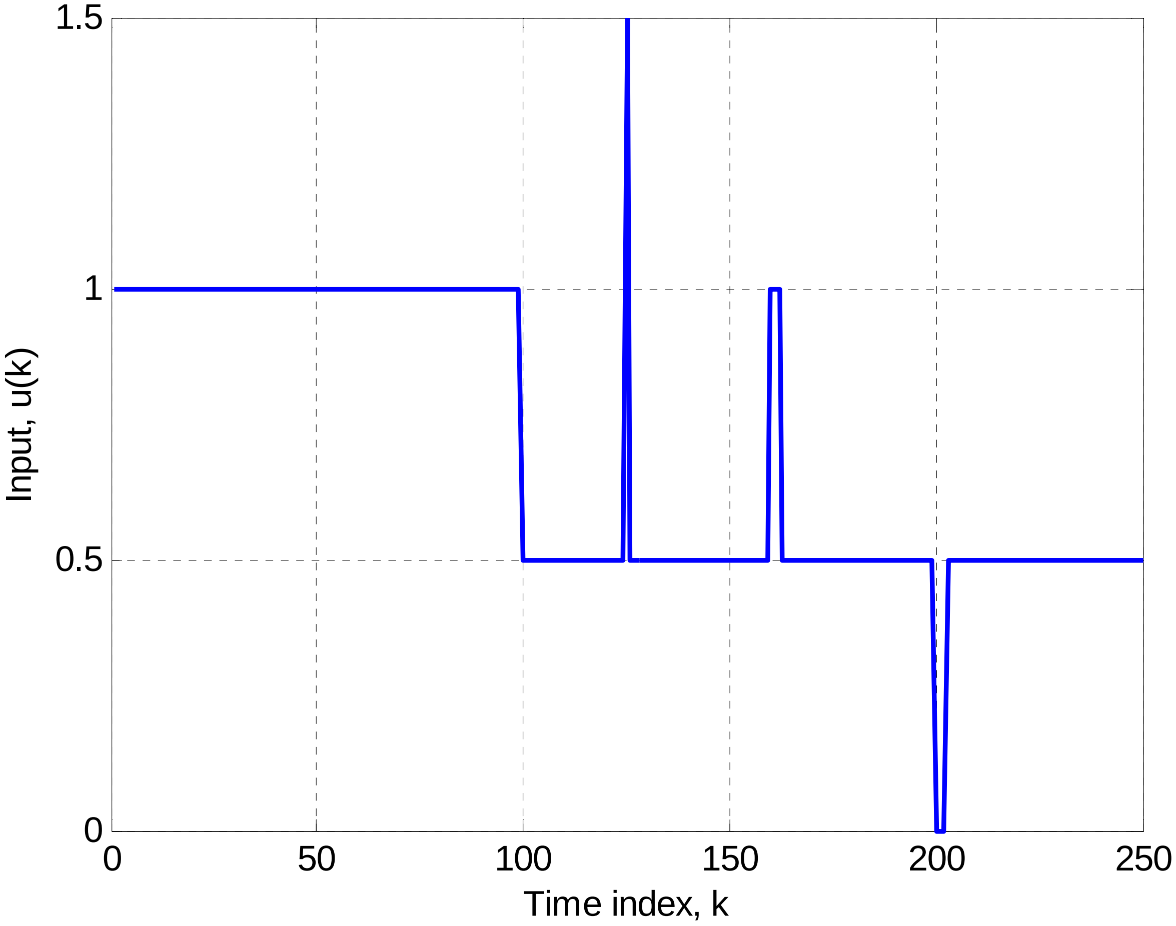


Figure 37.- Time history of step/pulse inputs applied to equation of Example 5.

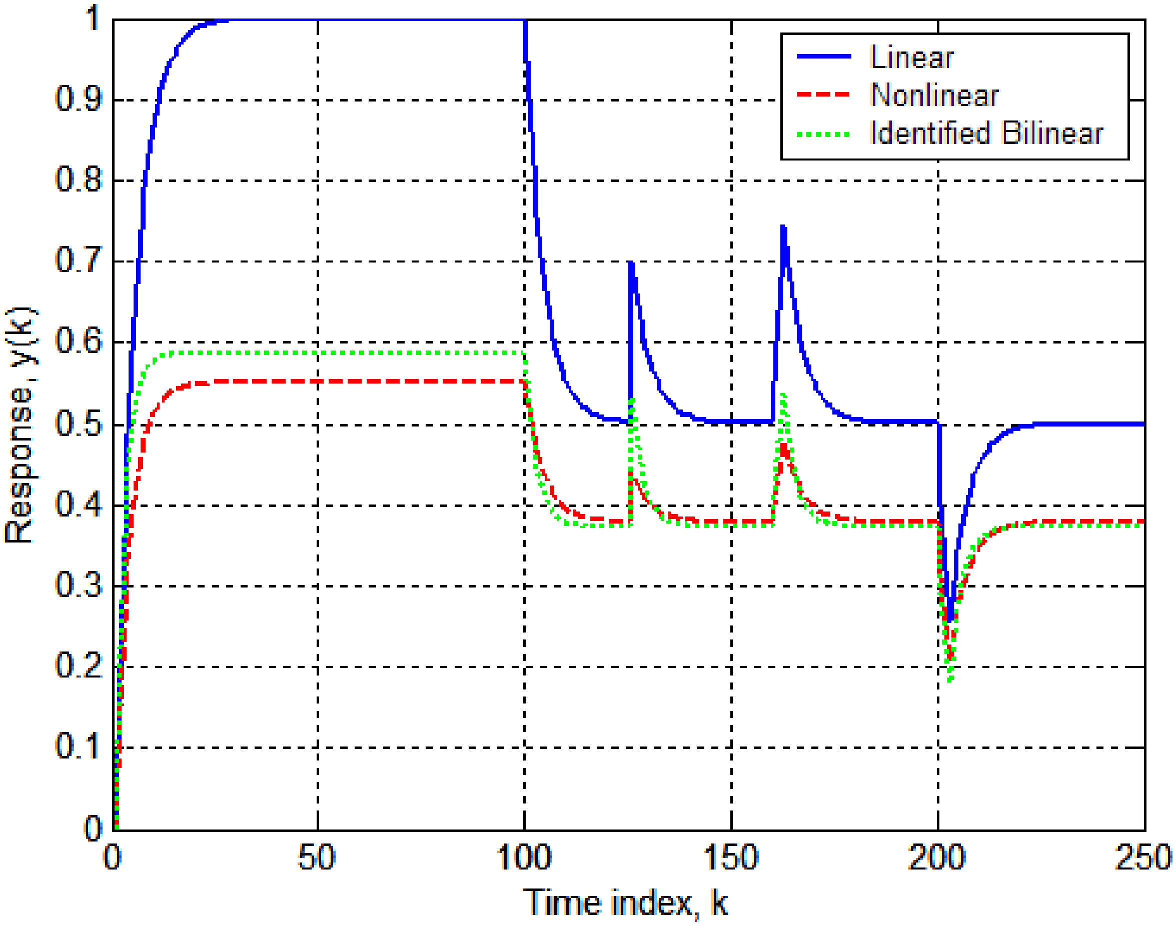


Figure 38.- Responses of linear, nonlinear, and identified bilinear equations for Example 5 to step/pulse inputs of figure 37.

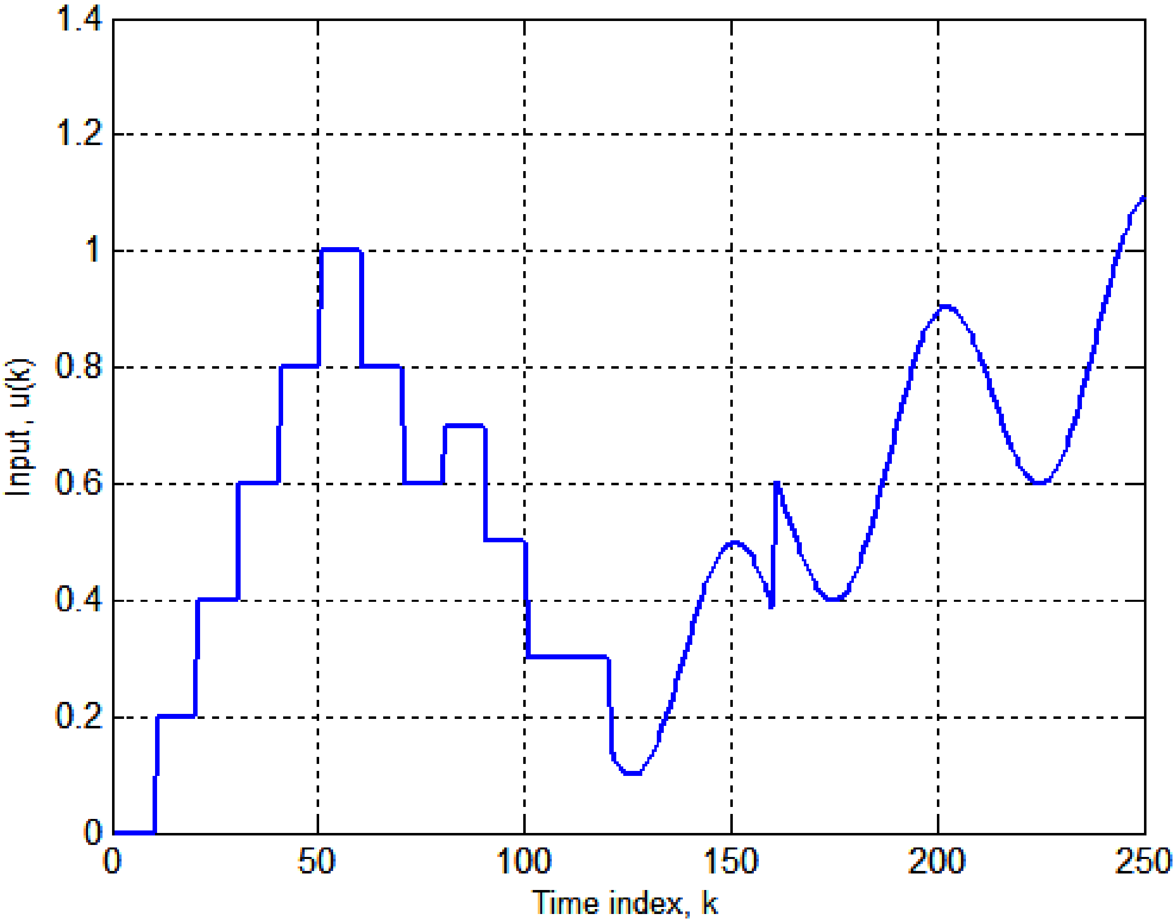


Figure 39.- Composite excitation applied to nonlinear difference model of Example 5.

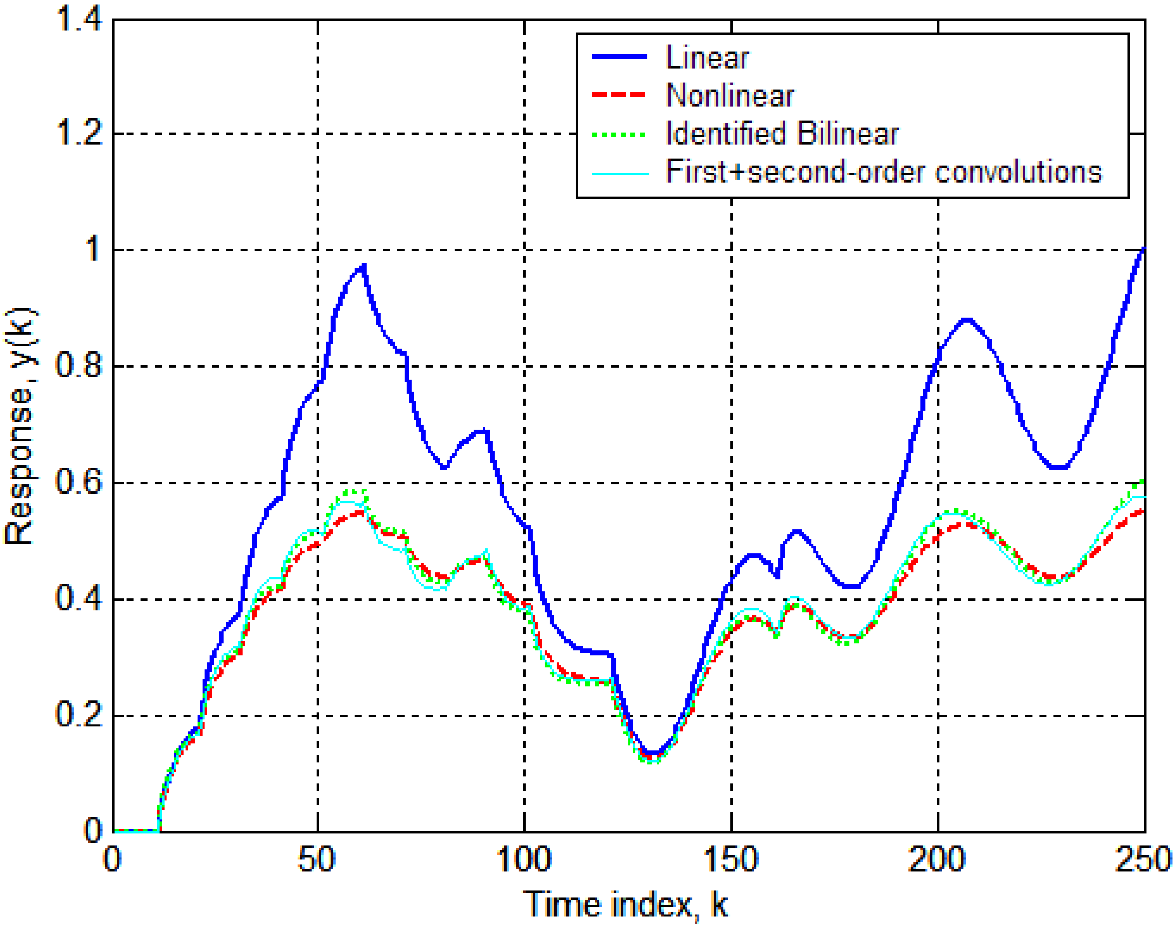


Figure 40.- Responses of linear, nonlinear, identified bilinear, and convolution models of nonlinear difference equation of Example 5 to composite input of figure 39.

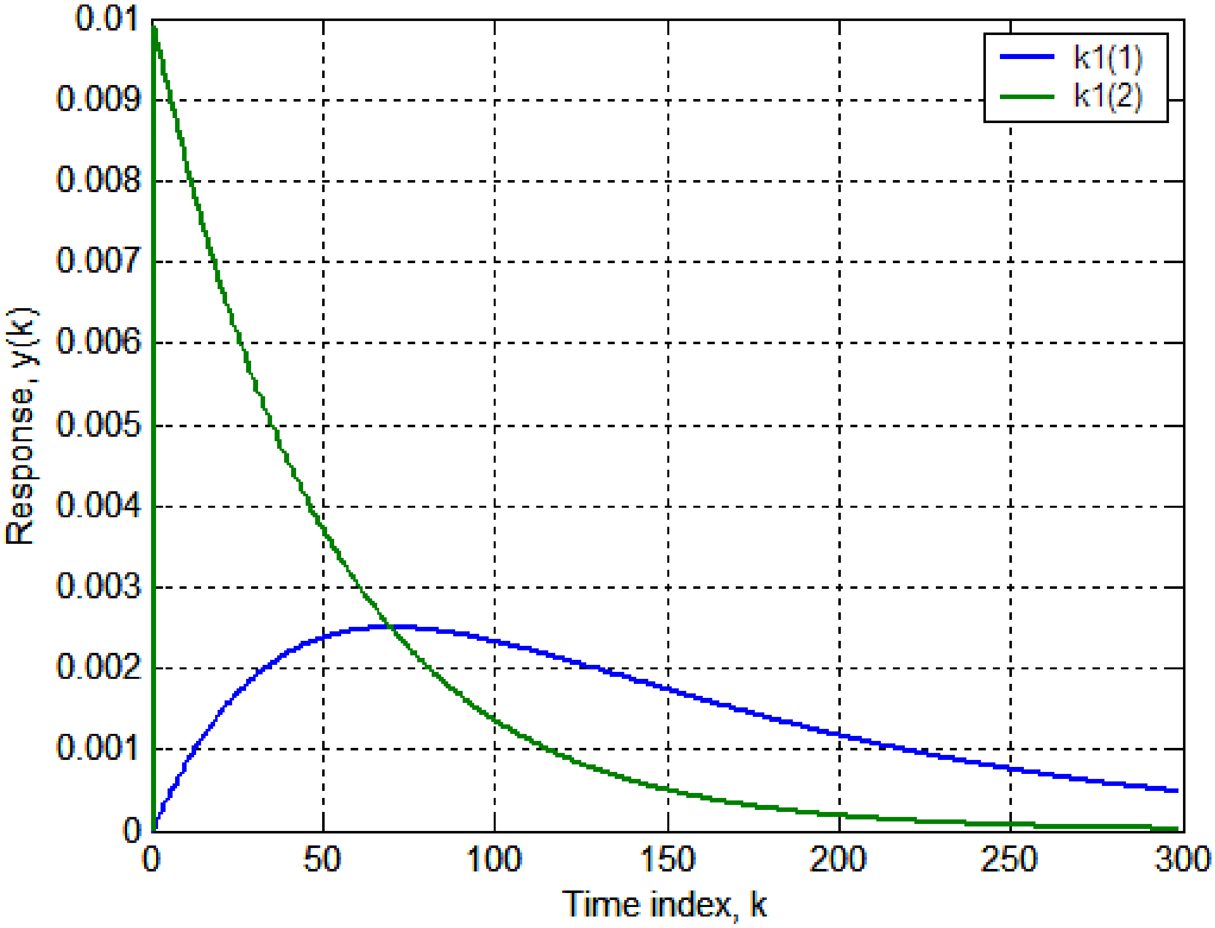


Figure 41.- Discrete first-order kernels of modified bilinear equation used for MISO examples ($N = 300$, $\Delta t = 0.01$ sec).

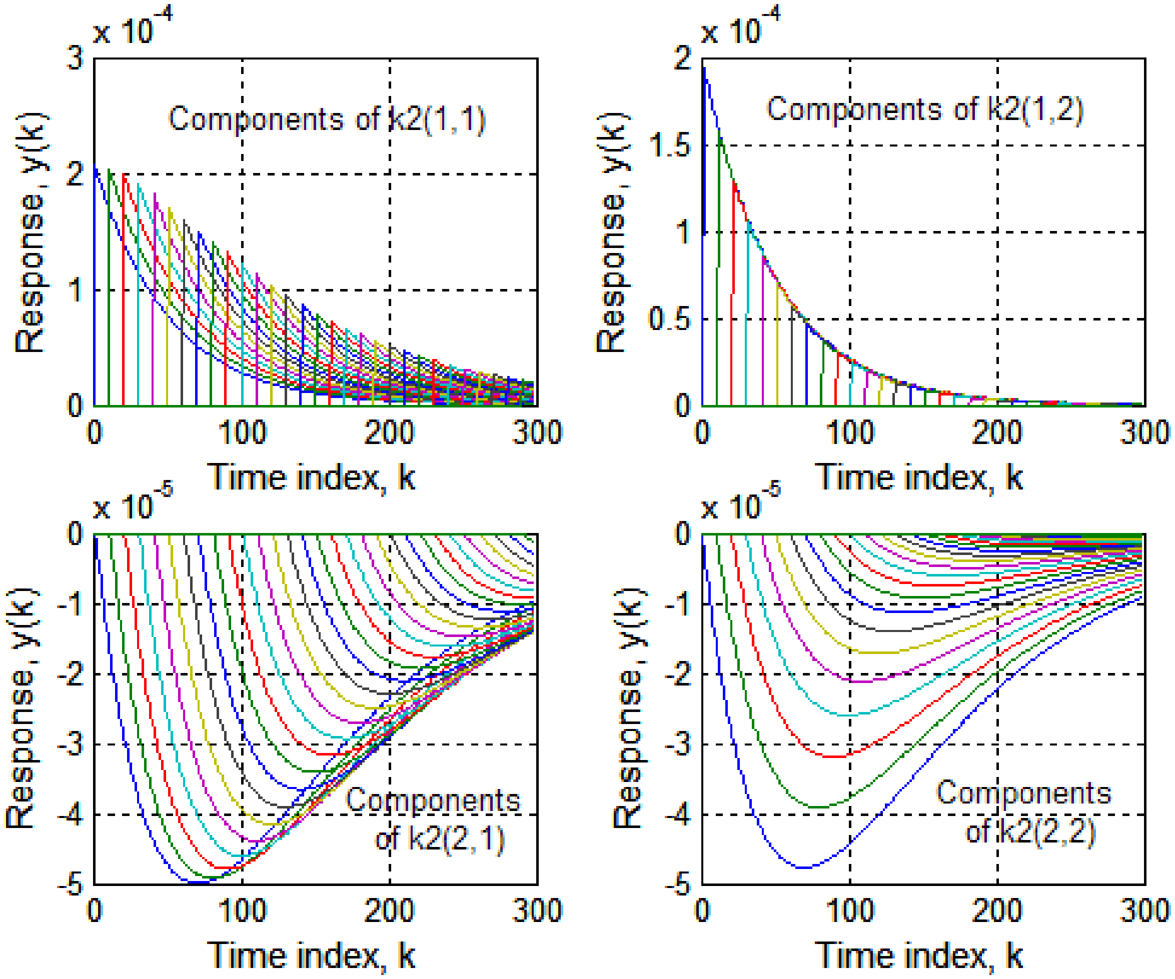


Figure 42.- Discrete second-order kernels of modified bilinear equation used for MISO examples at 30 values of time lag T ($N = 300$, $\Delta t = 0.01$ sec, $\Delta T = 0.1$ sec, and $ncomp = 30$).

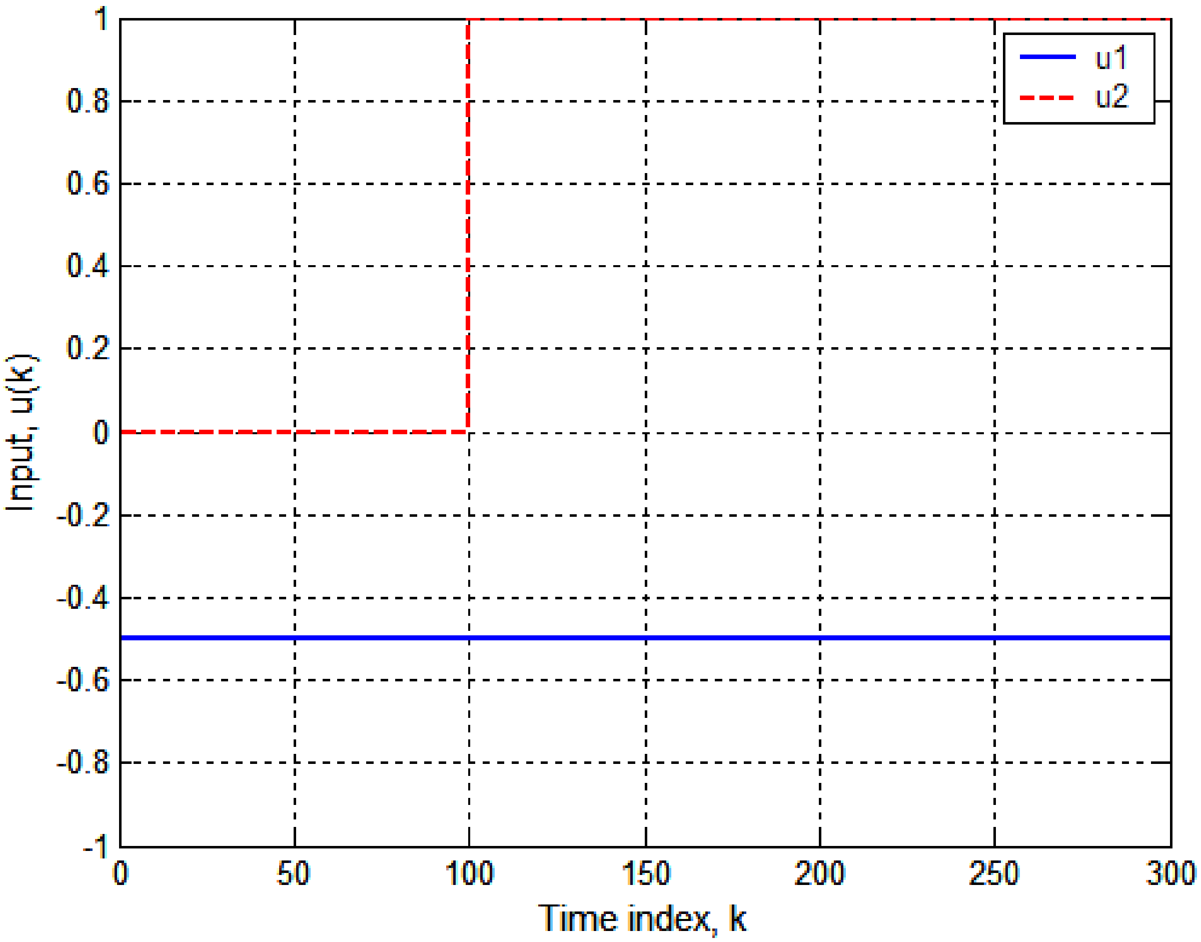


Figure 43.- Step inputs applied to MISO bilinear system example.

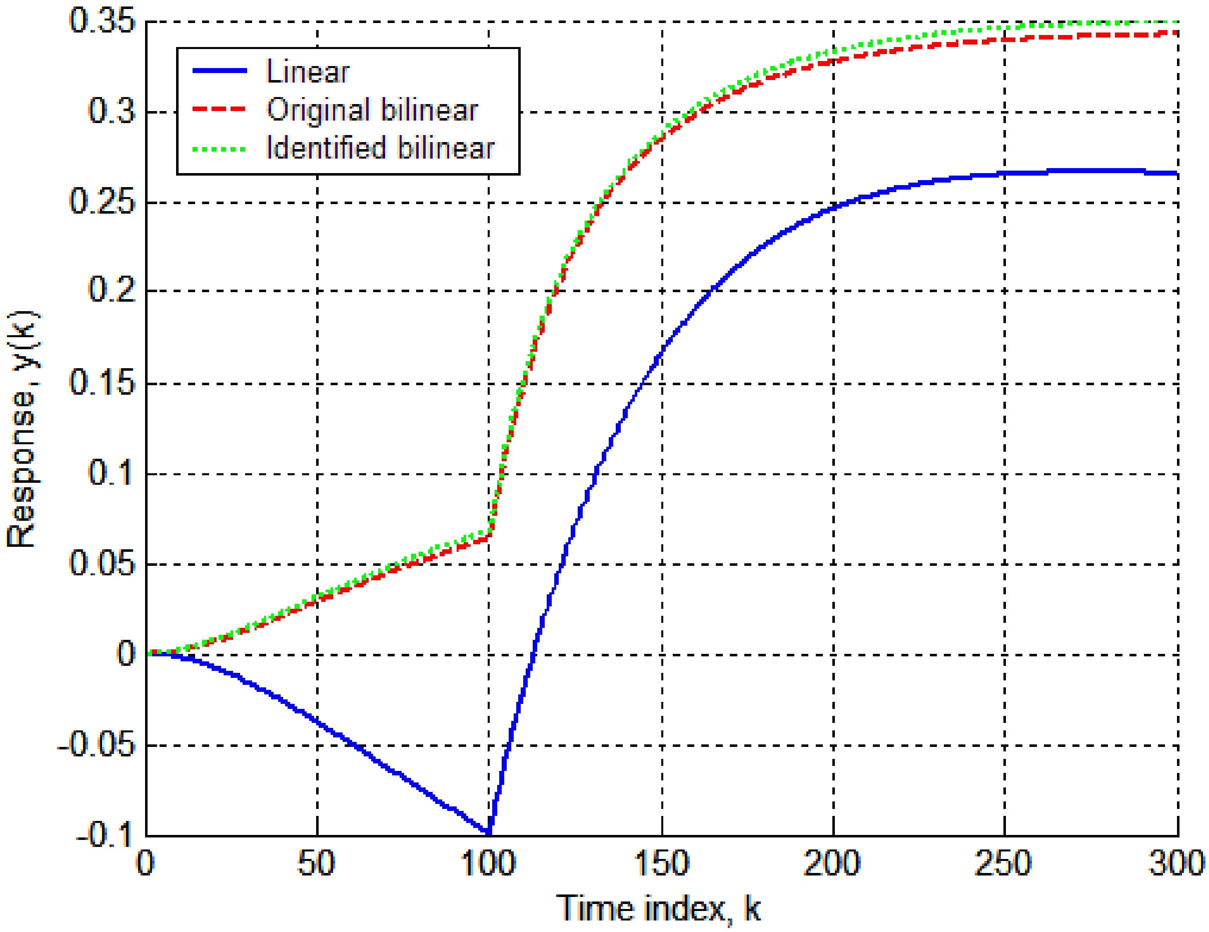


Figure 44.- Linear, original bilinear, and identified bilinear responses of MISO bilinear system example to step inputs of figure 43.

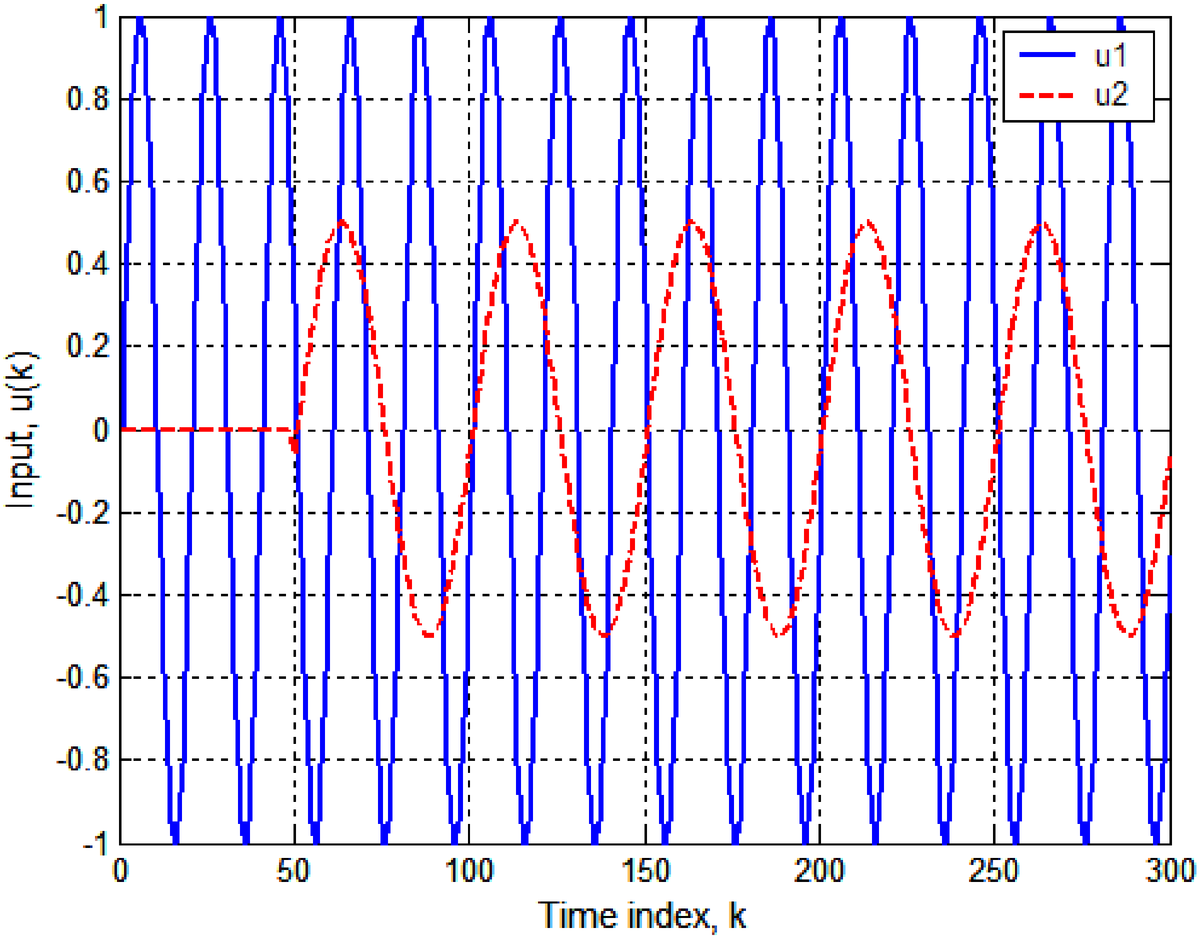


Figure 45.- First pattern of sinusoidal inputs applied to MISO bilinear equation (Input u_2 delayed 50 time points).

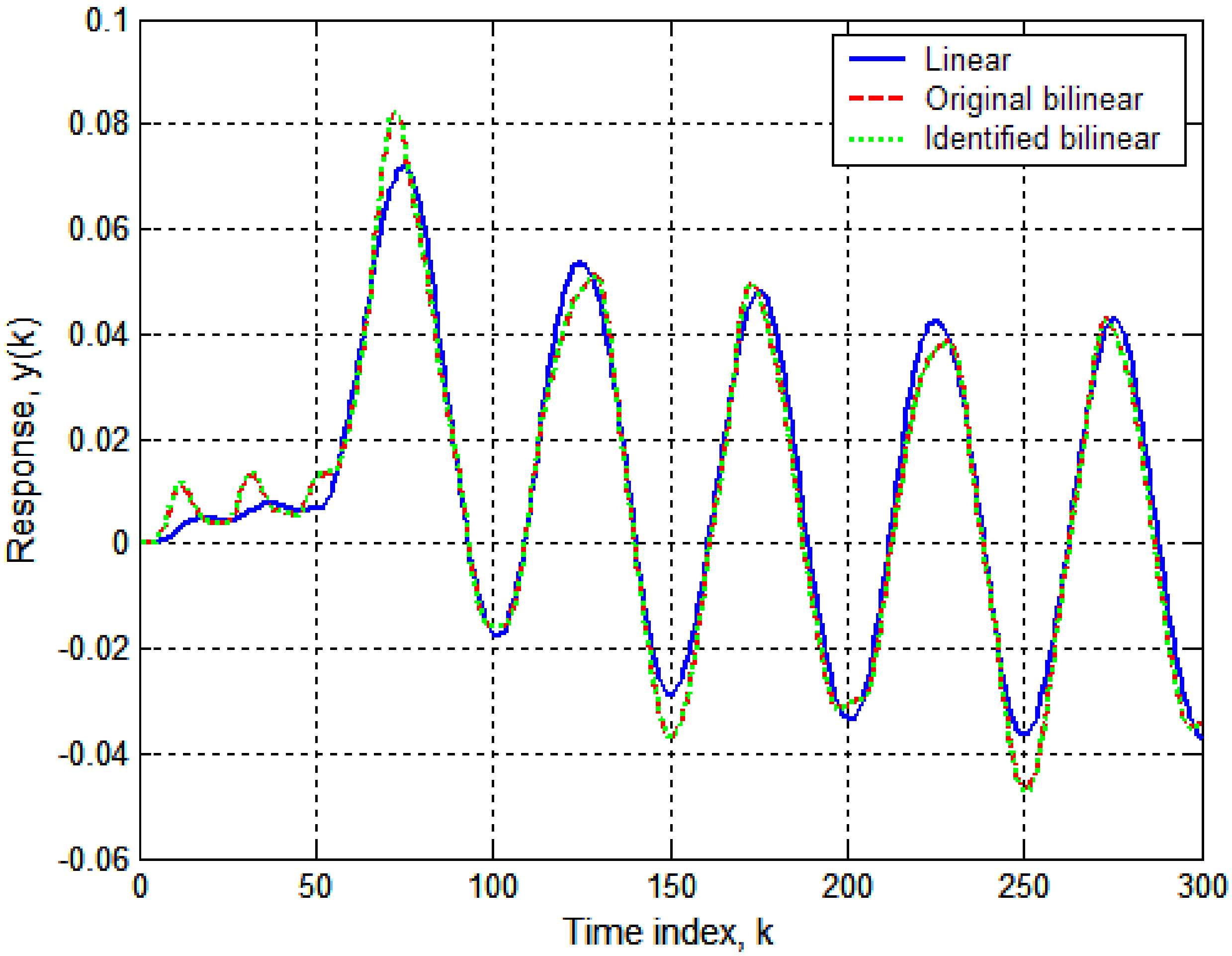


Figure 46.- Responses of linear, original bilinear, and identified bilinear equations to sinusoidal inputs of figure 45.

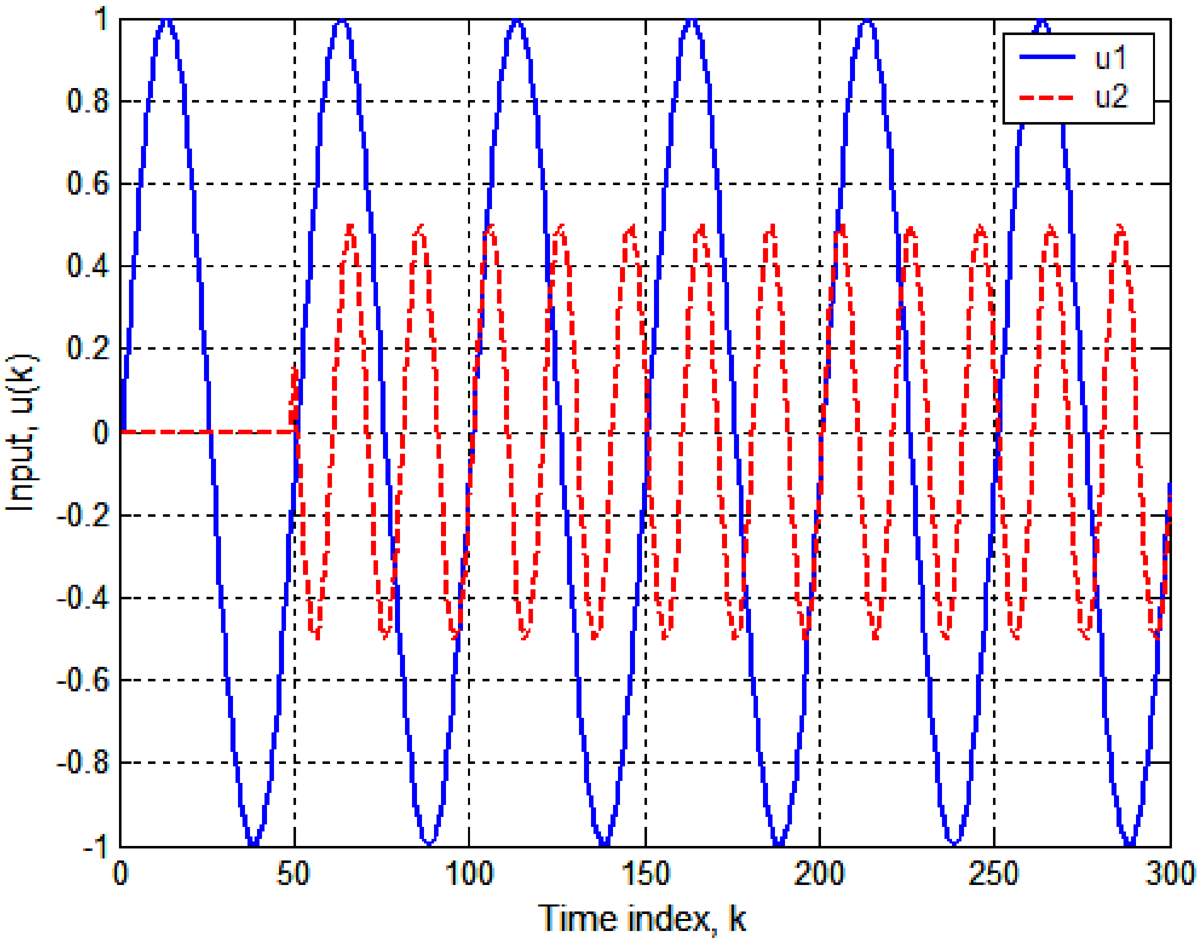


Figure 47.- Second pattern of sinusoidal inputs applied to MISO bilinear example equation (Inputs same as in figure 45 but with interchanged frequencies).

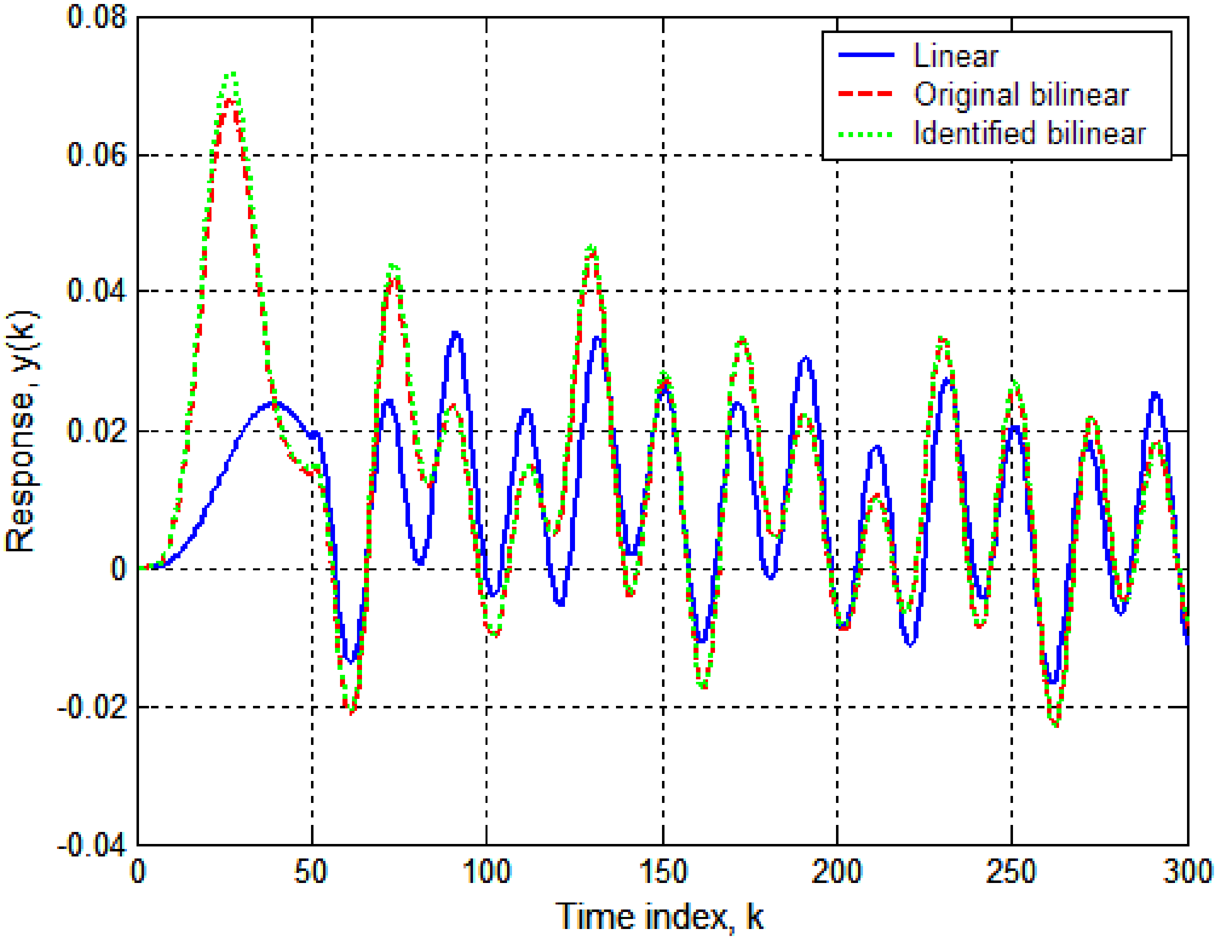


Figure 48.- Responses of linear, original bilinear, and identified bilinear equations to sinusoidal inputs of figure 47.

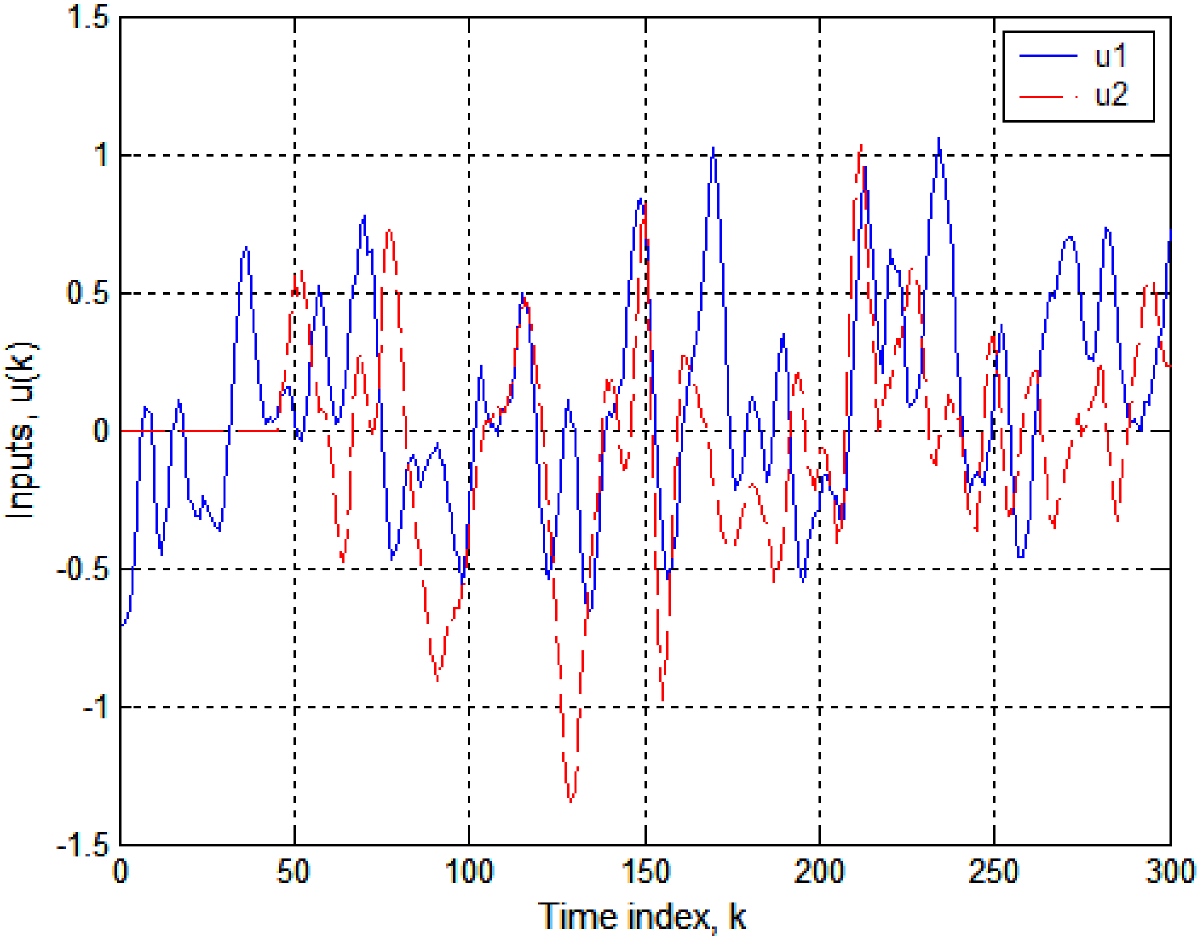


Figure 49.- Random inputs applied to both u_1 and u_2. Input u_2 delayed 45 time points.

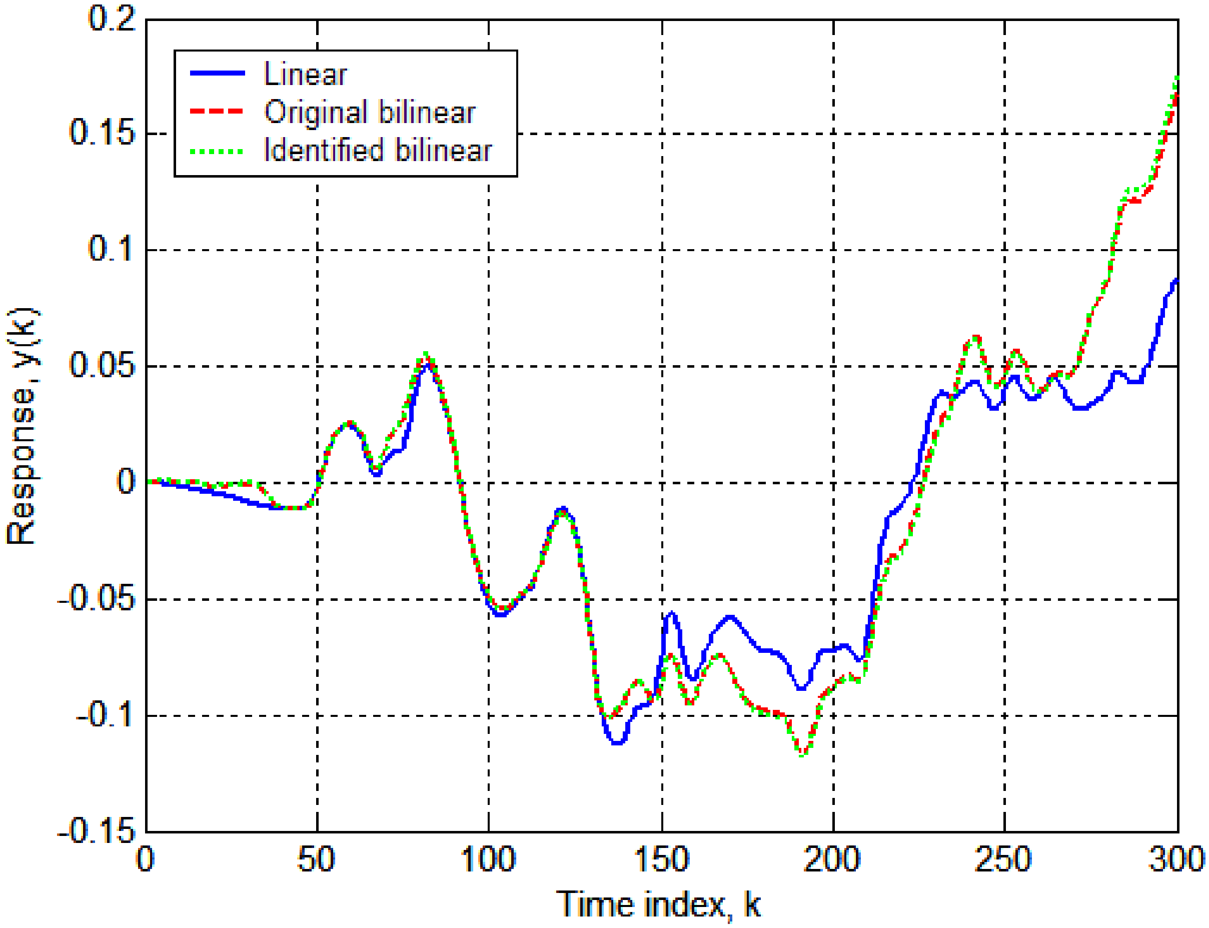


Figure 50.- Responses of linear, original bilinear, and identified bilinear equations to random inputs of figure 49.

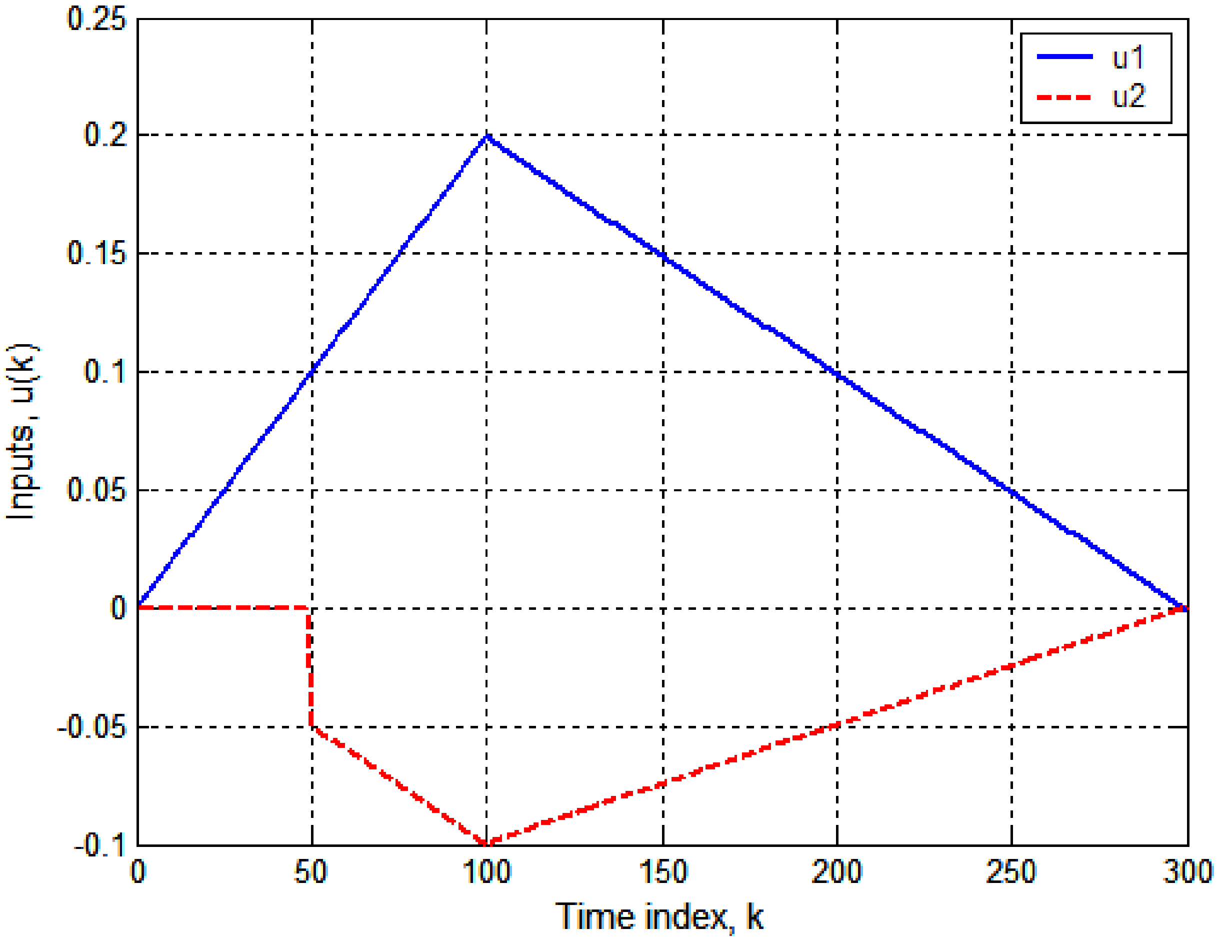


Figure 51.- Ramp inputs applied to u_1 and u_2. Input u_2 delayed 50 time points.

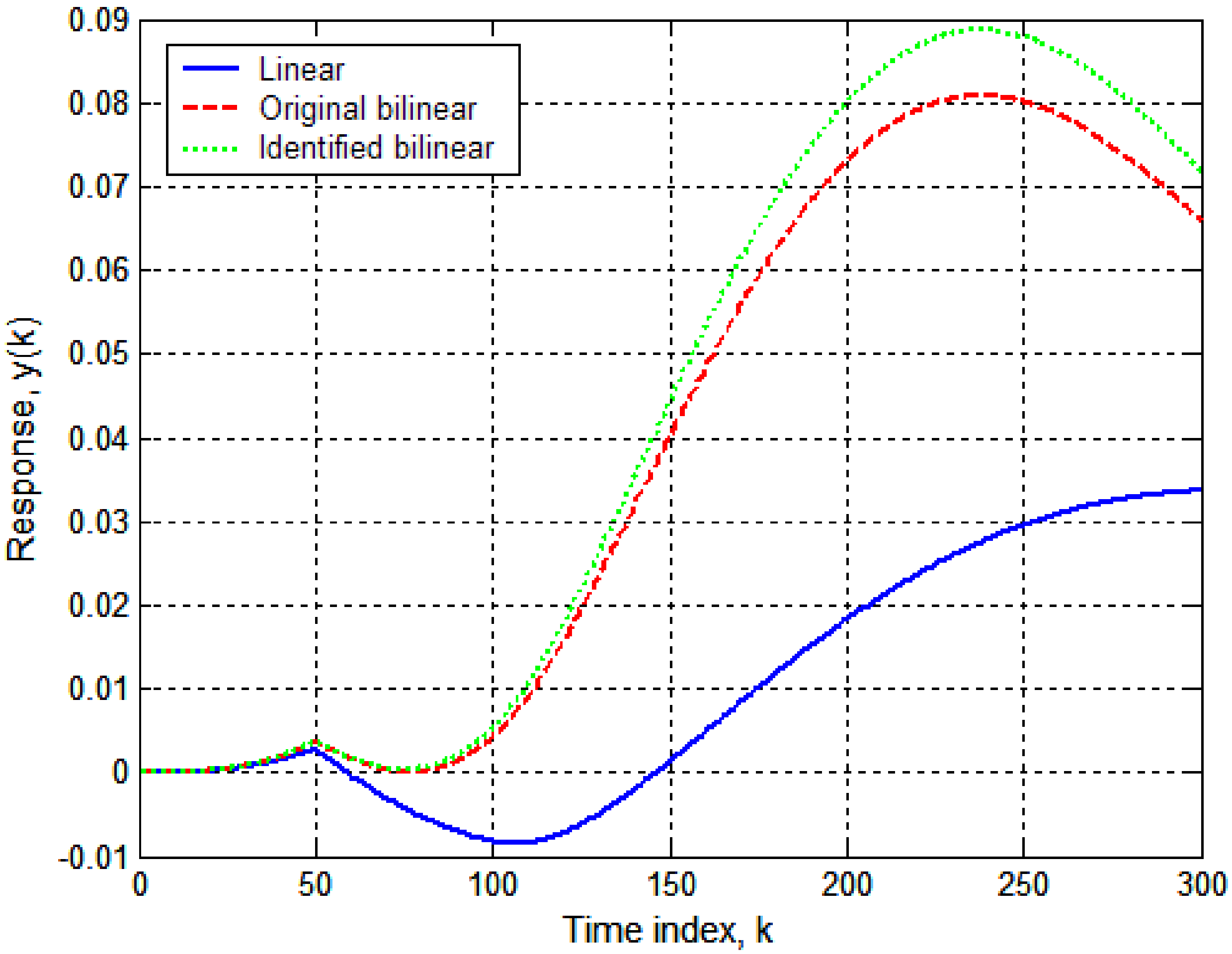


Figure 52.- Responses of linear, original bilinear, and identified bilinear equations to ramp inputs of figure 51.

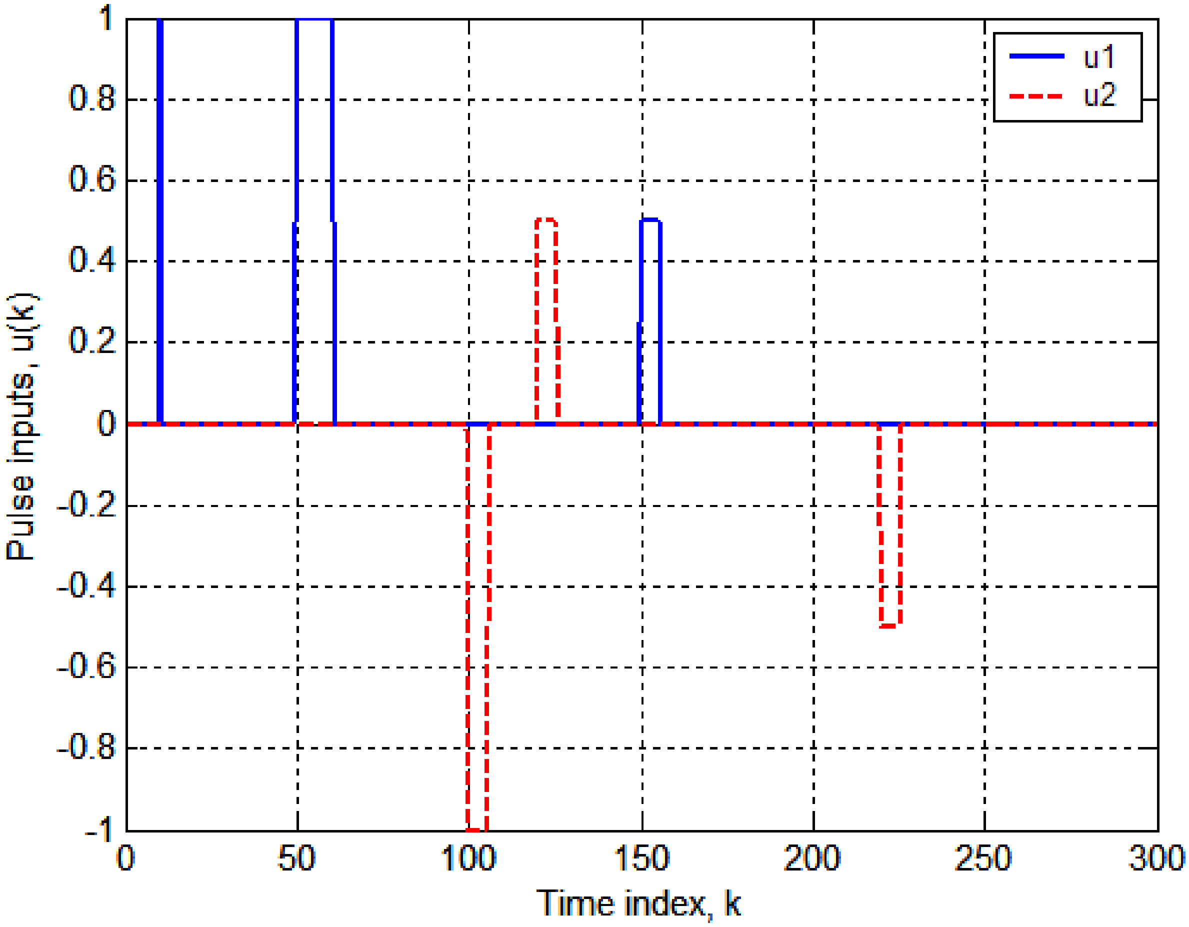


Figure 53.- Multi-pulse u_1 and u_2 loading patterns applied to MISO bilinear system.

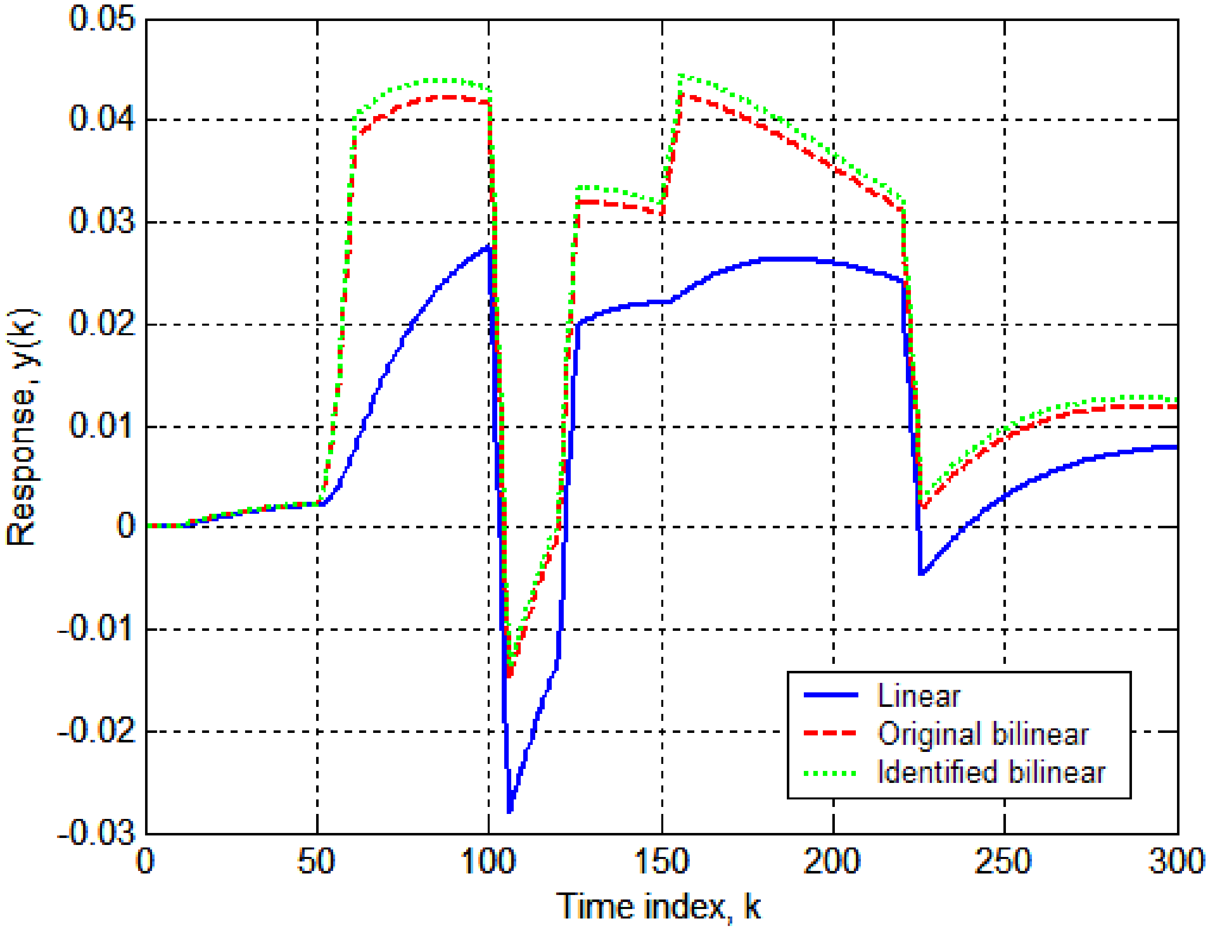


Figure 54.- Responses of linear, original bilinear, and identified bilinear equations to multi-pulse inputs of figure 53.

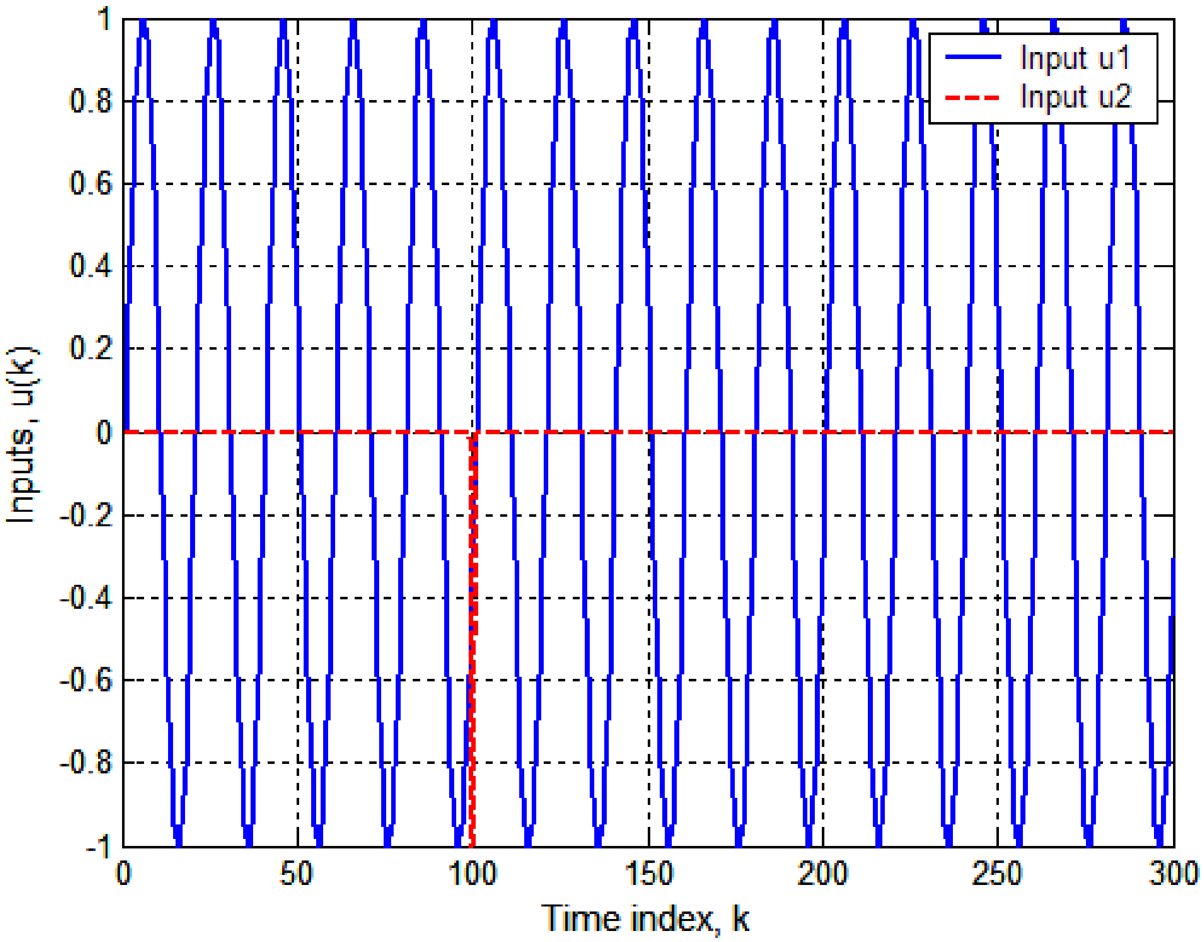


Figure 55.- Five-hertz sinusoid on u_1; single sharp pulse on u_2.

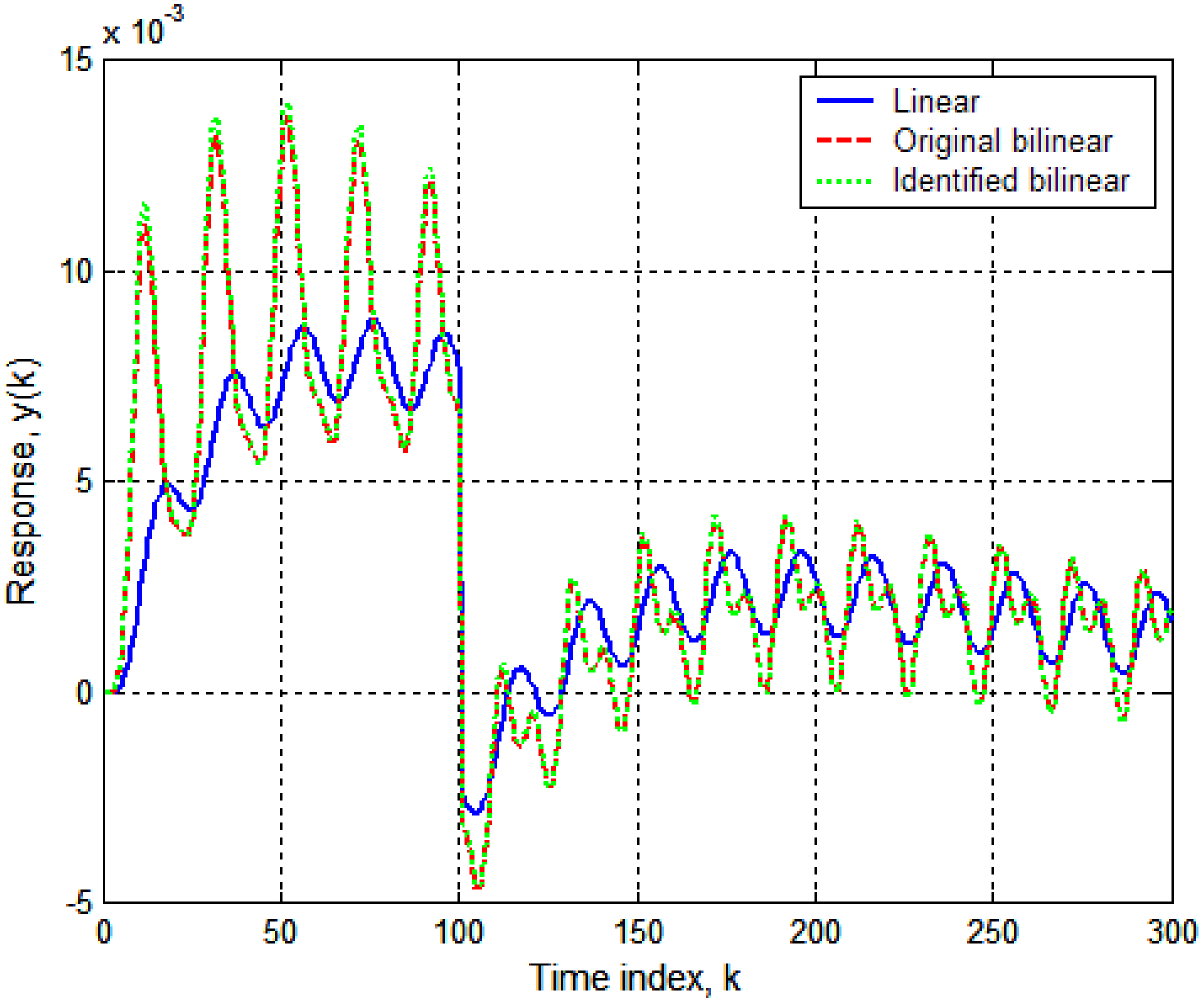


Figure 56.- Response time histories of linear, original bilinear, and identified bilinear equations to inputs of figure 55.

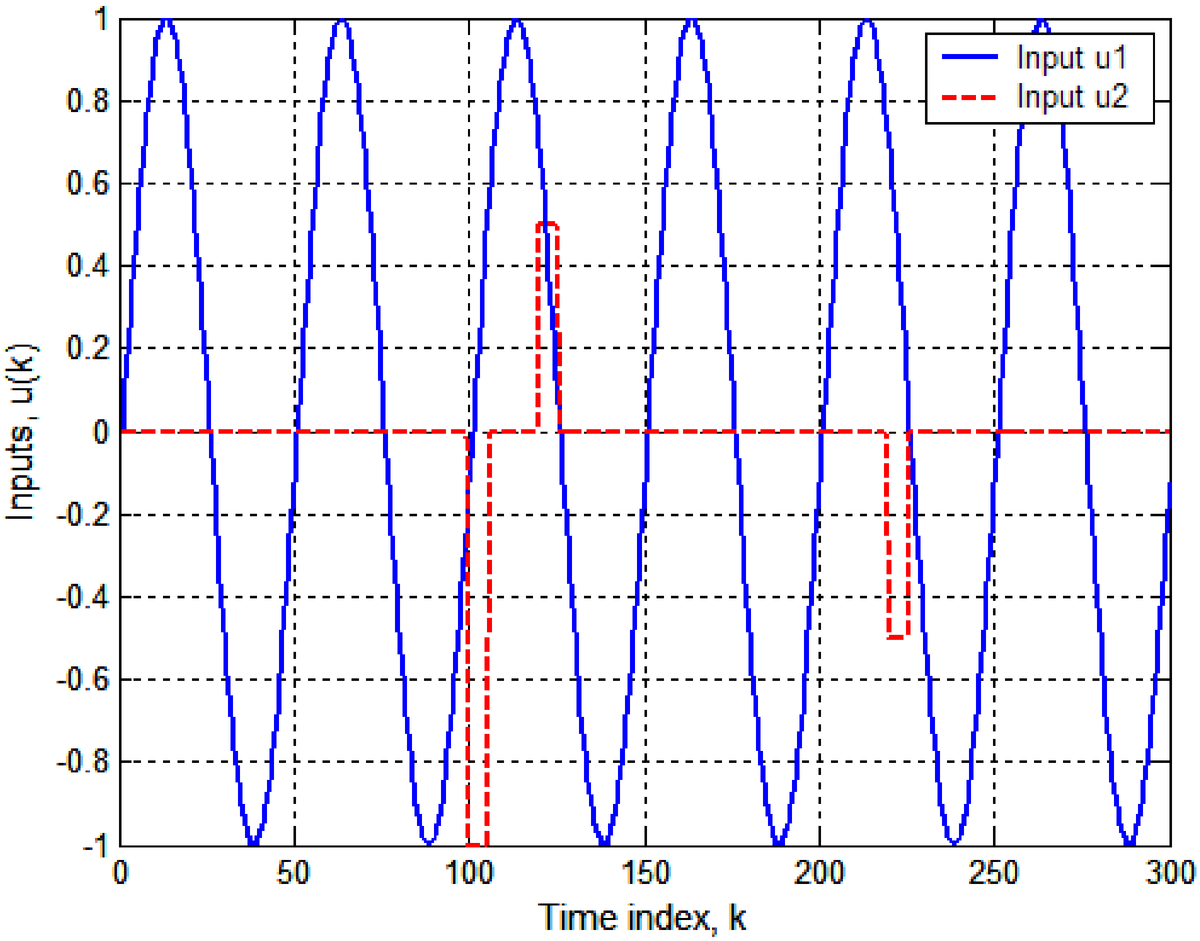


Figure 57.- Two-hertz sinusoid on u_1; sequence of three narrow pulses on u_2.

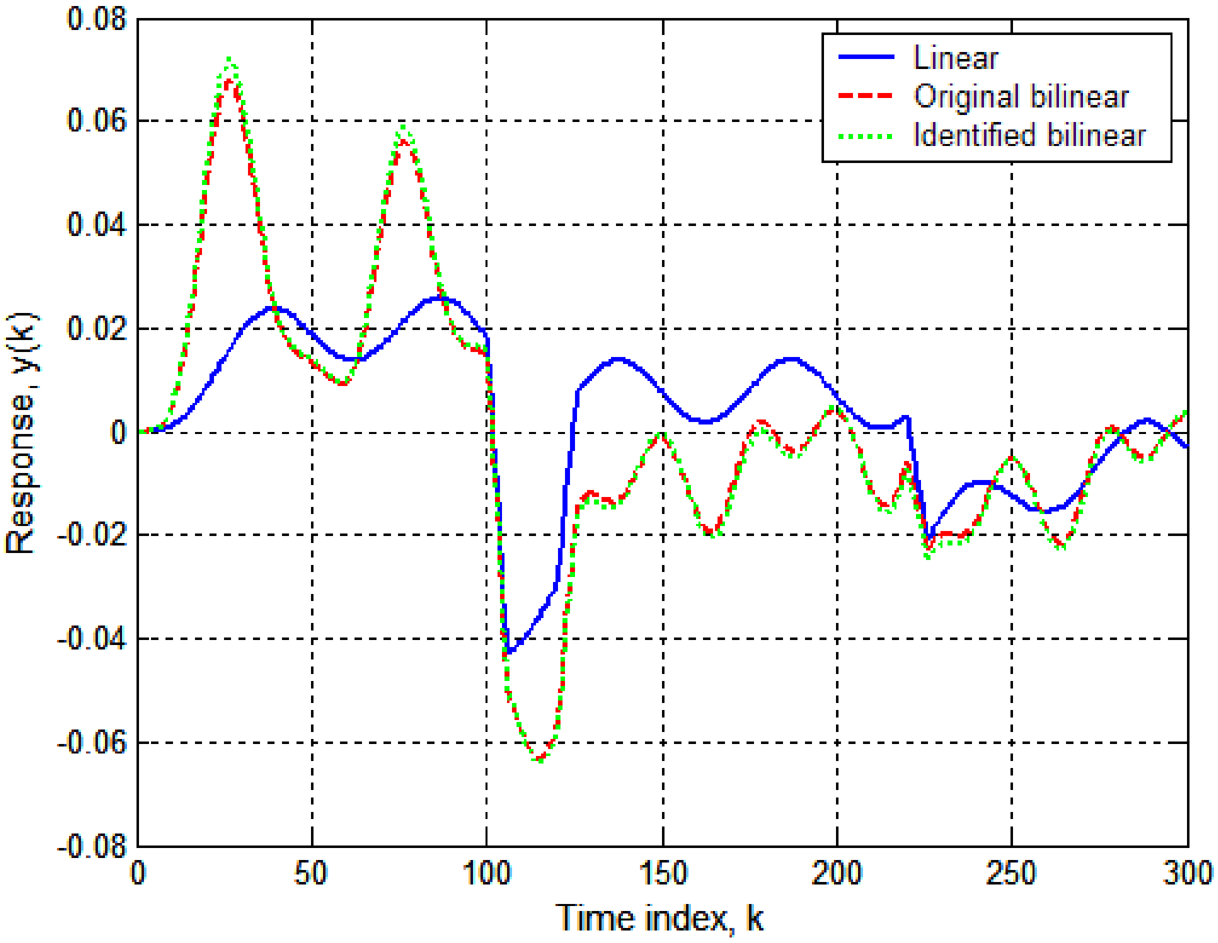


Figure 58.- Response time histories using linear, original bilinear, and identified bilinear equations to sine and pulse-train inputs of figure 57.

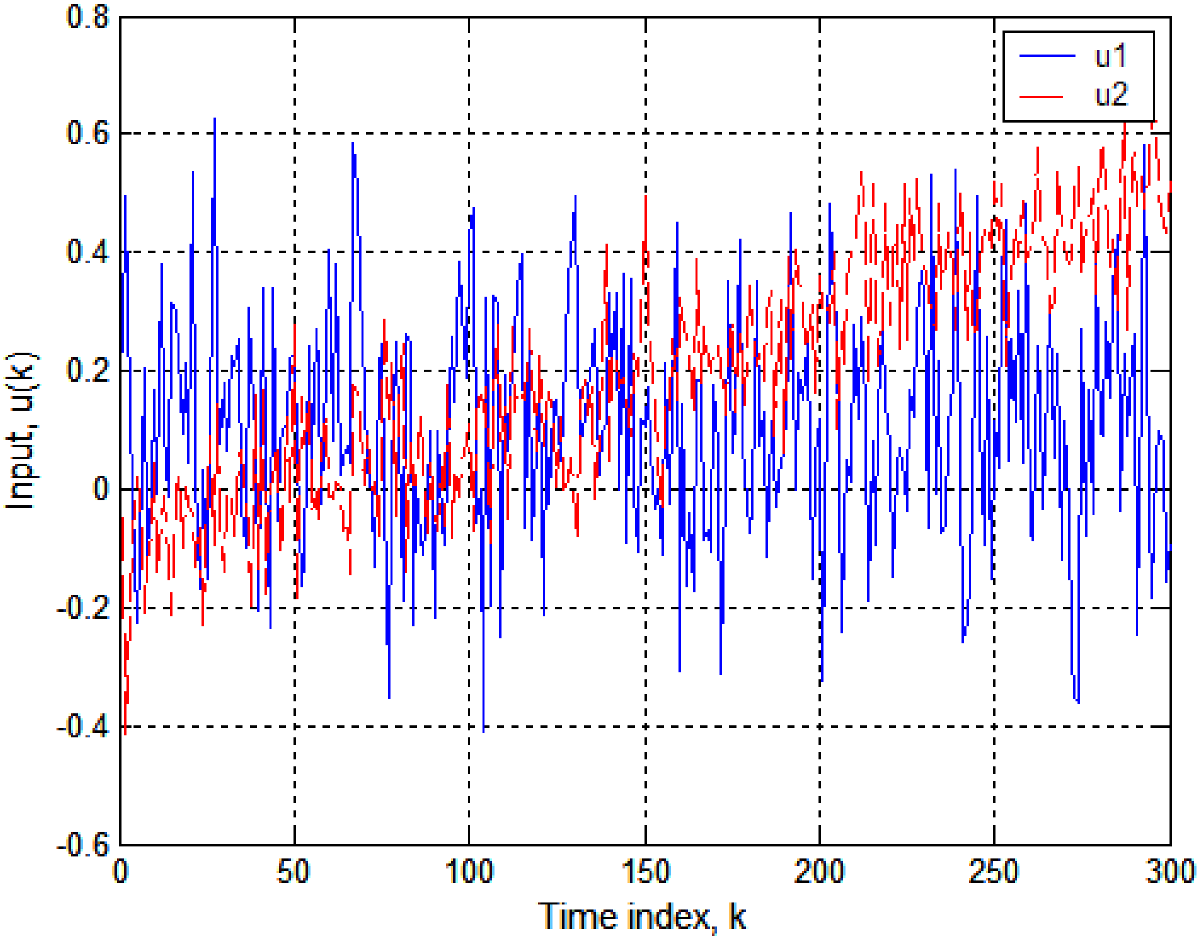


Figure 59.- Noisy u_1 and u_2 step inputs with a constant 0.002 per step drift in u_2.

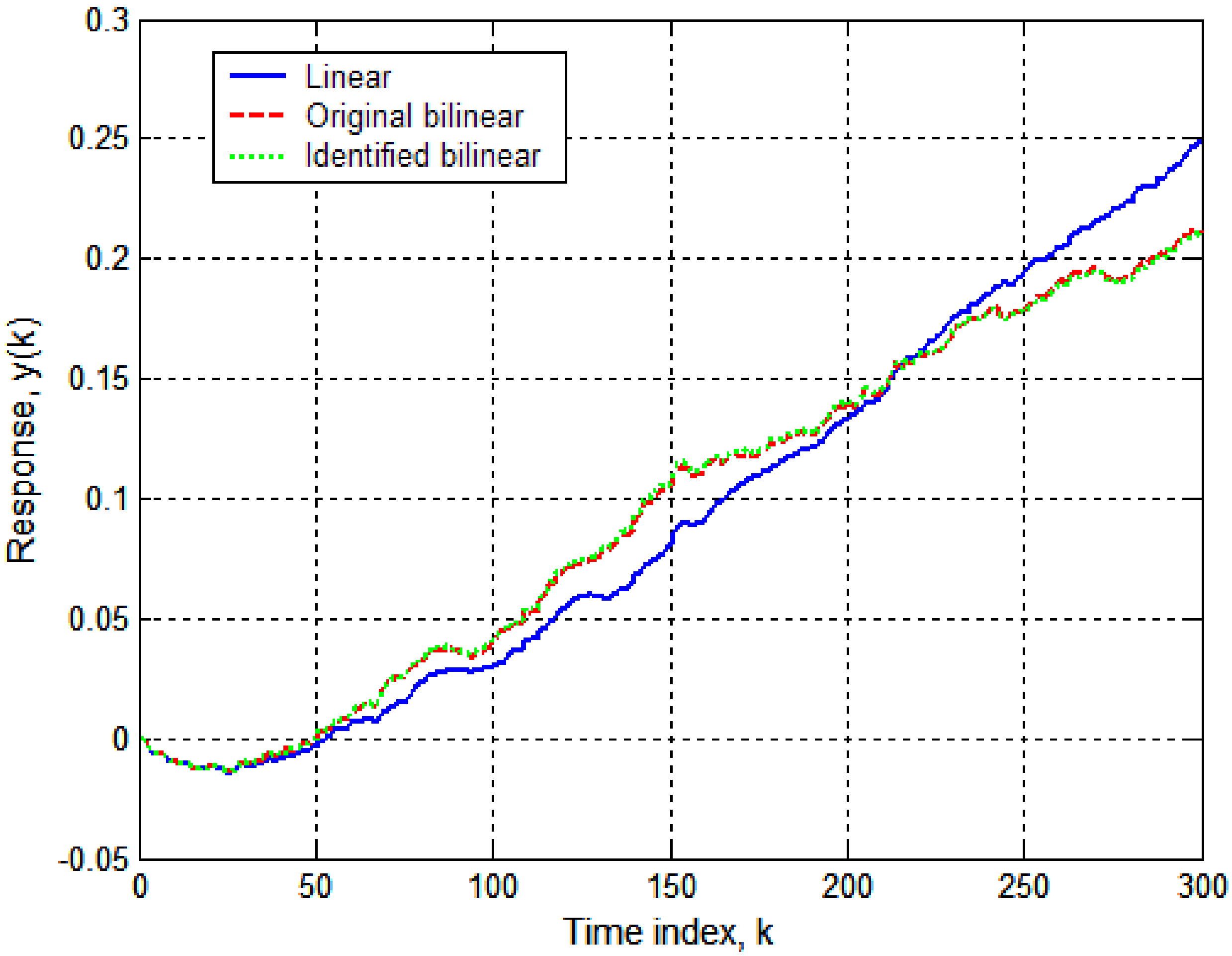


Figure 60.- Responses of linear, original bilinear, and identified bilinear equations to noisy step inputs of figure 59.

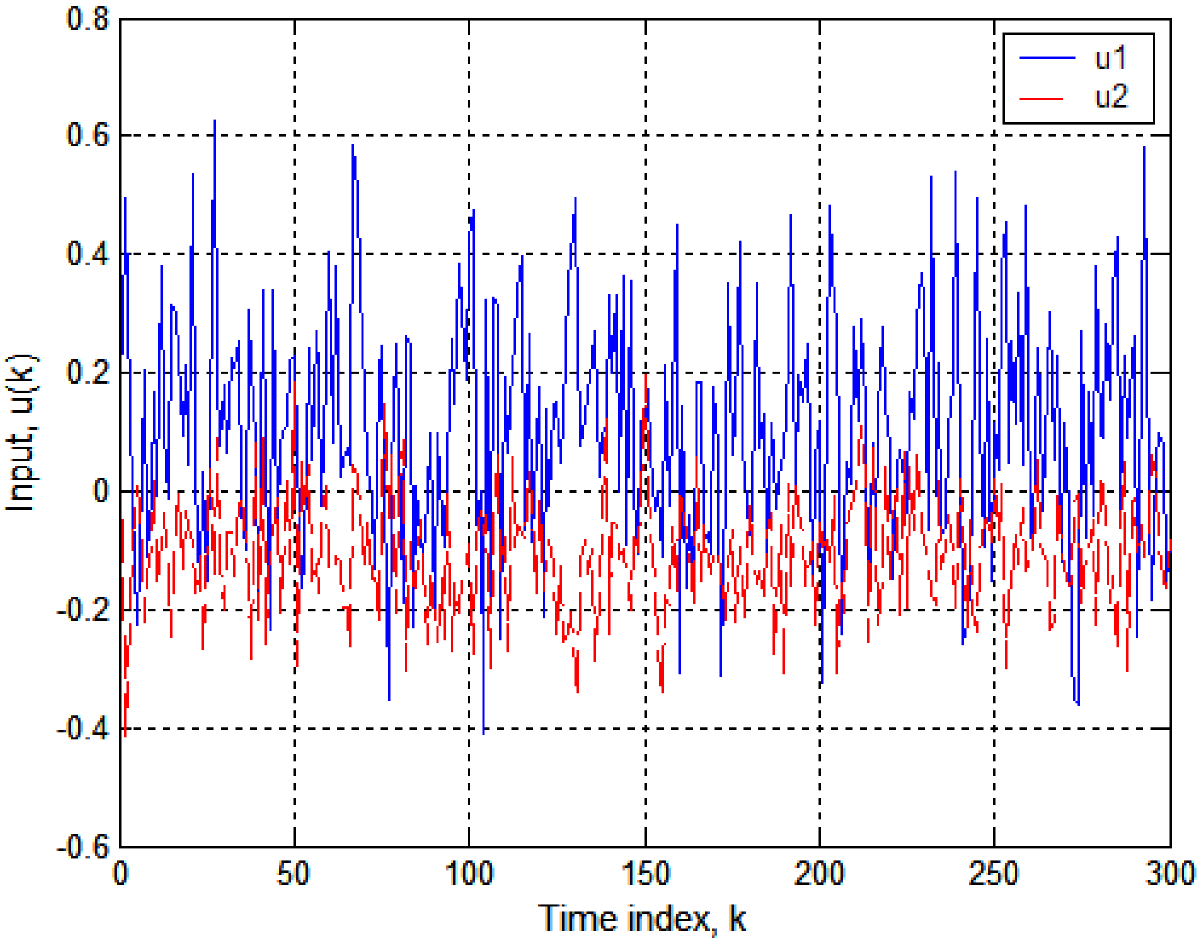


Figure 61.- Same noisy step inputs as in figure 59 but with no drift in u_2 signal.

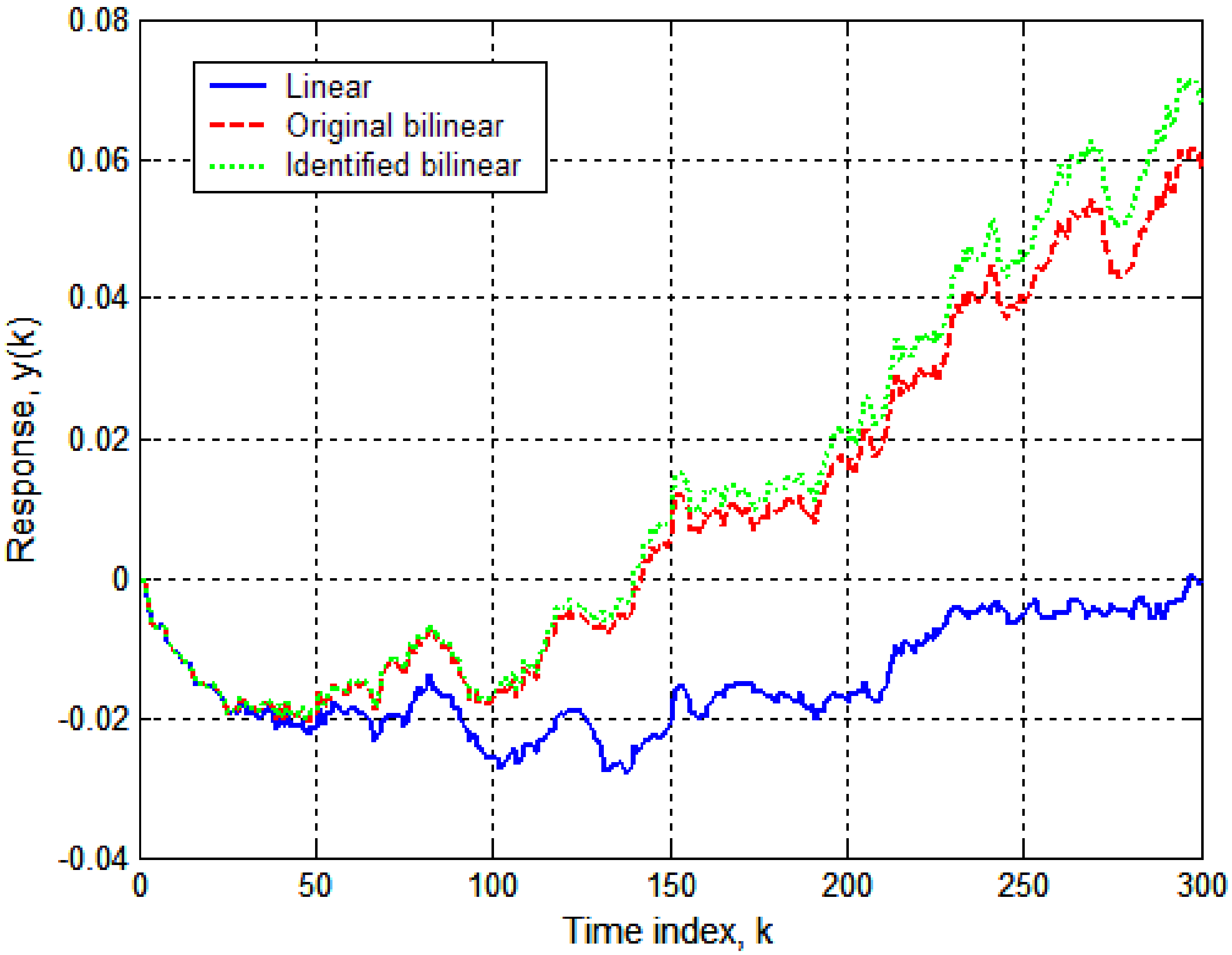


Figure 62.- Responses of linear, original bilinear, and identified bilinear equations to noisy step inputs of figure 61.

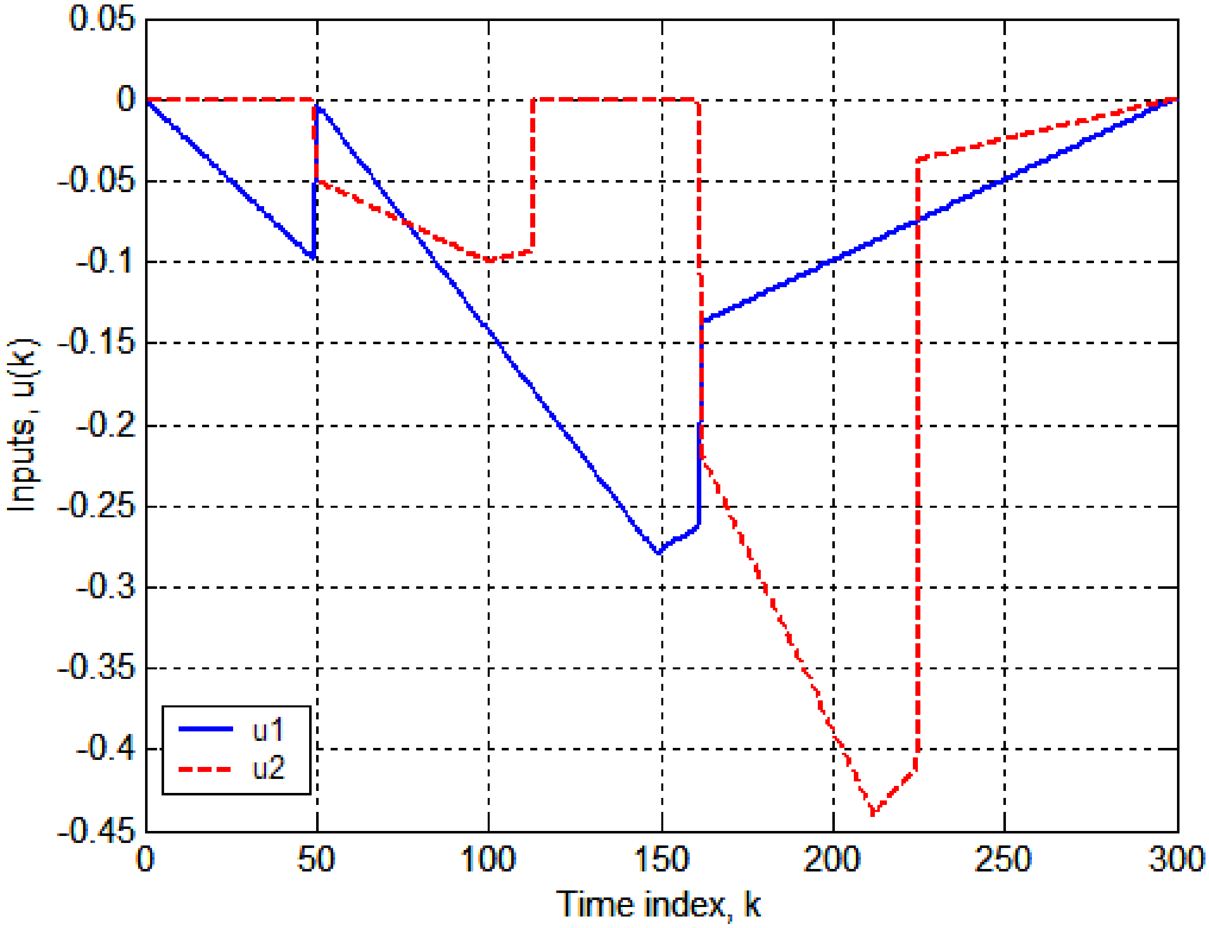


Figure 63.- Sawtooth/ramp patterns of u_1 and u_2 input time histories.

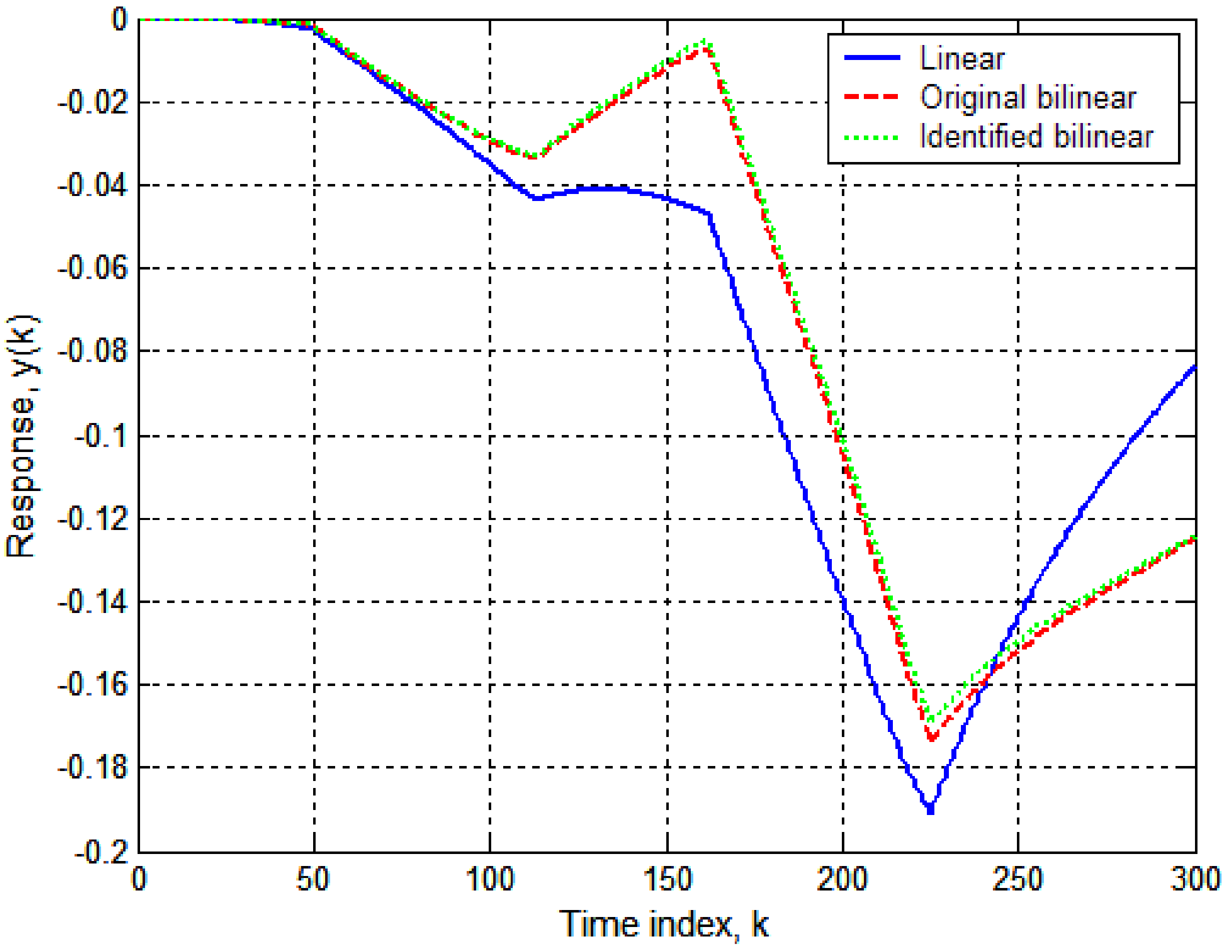


Figure 64.- Responses of linear, original bilinear, and identified bilinear equations to sawtooth/ramp inputs of figure 63.

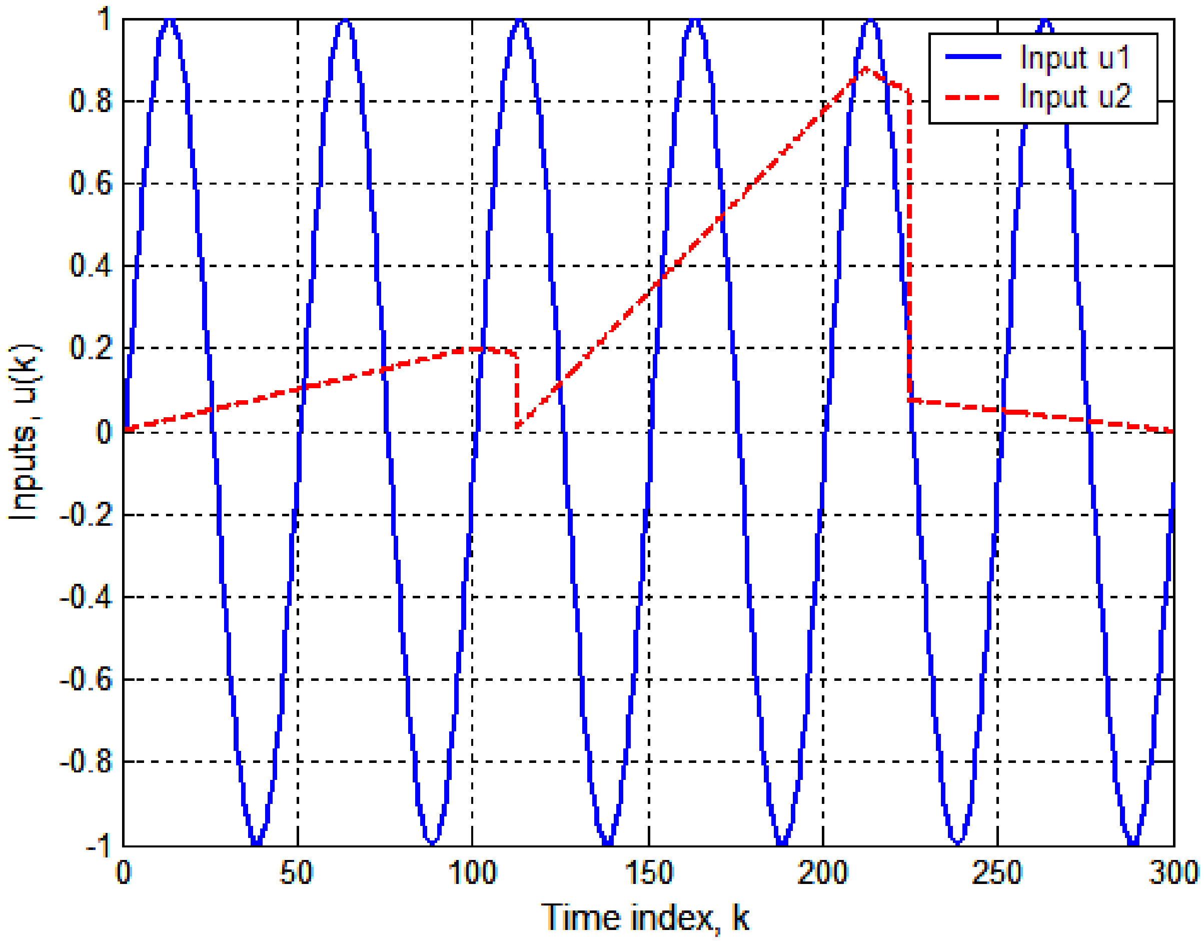


Figure 65.- Two-hertz sinusoid on u_1; series of up and down ramps on u_2.

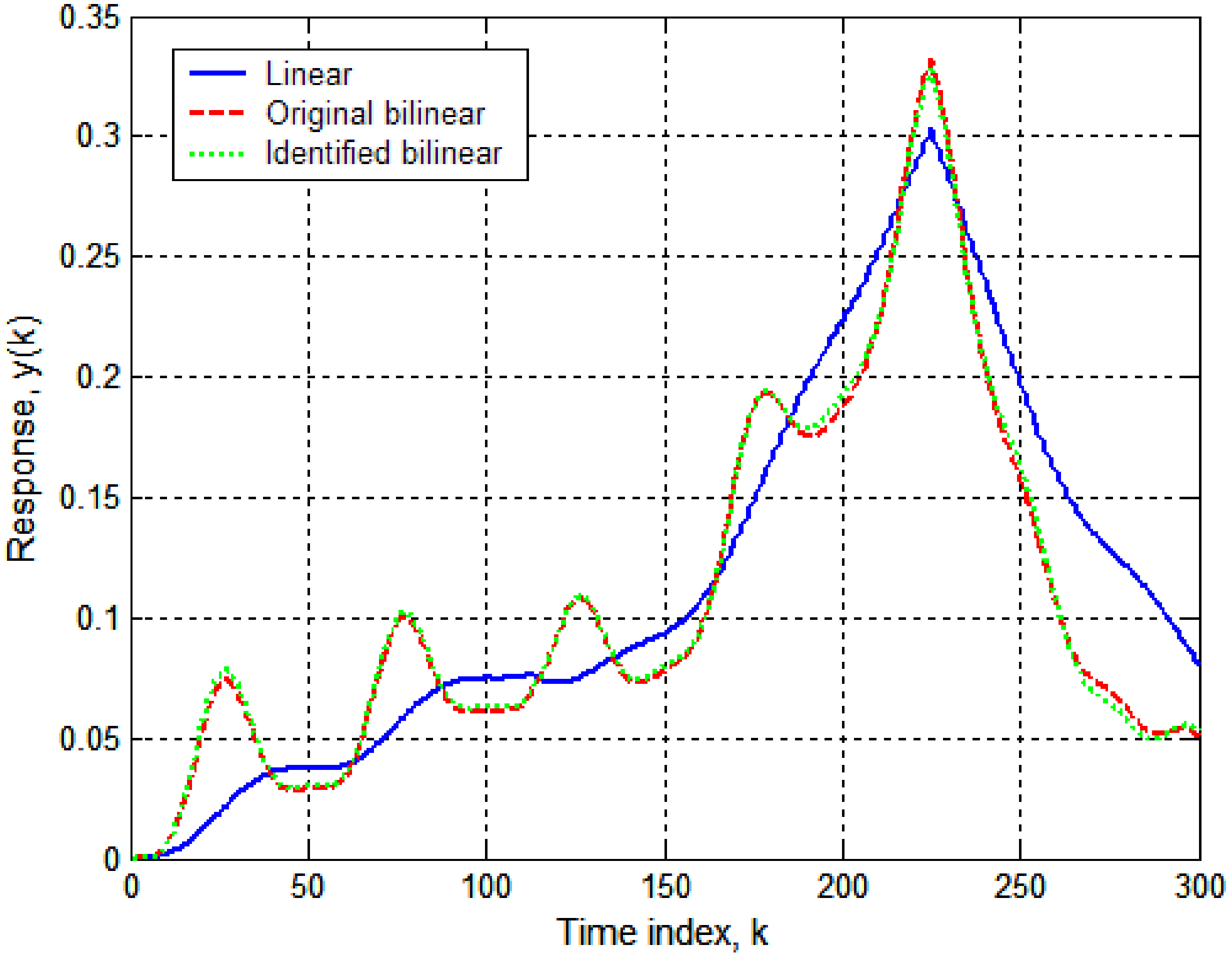
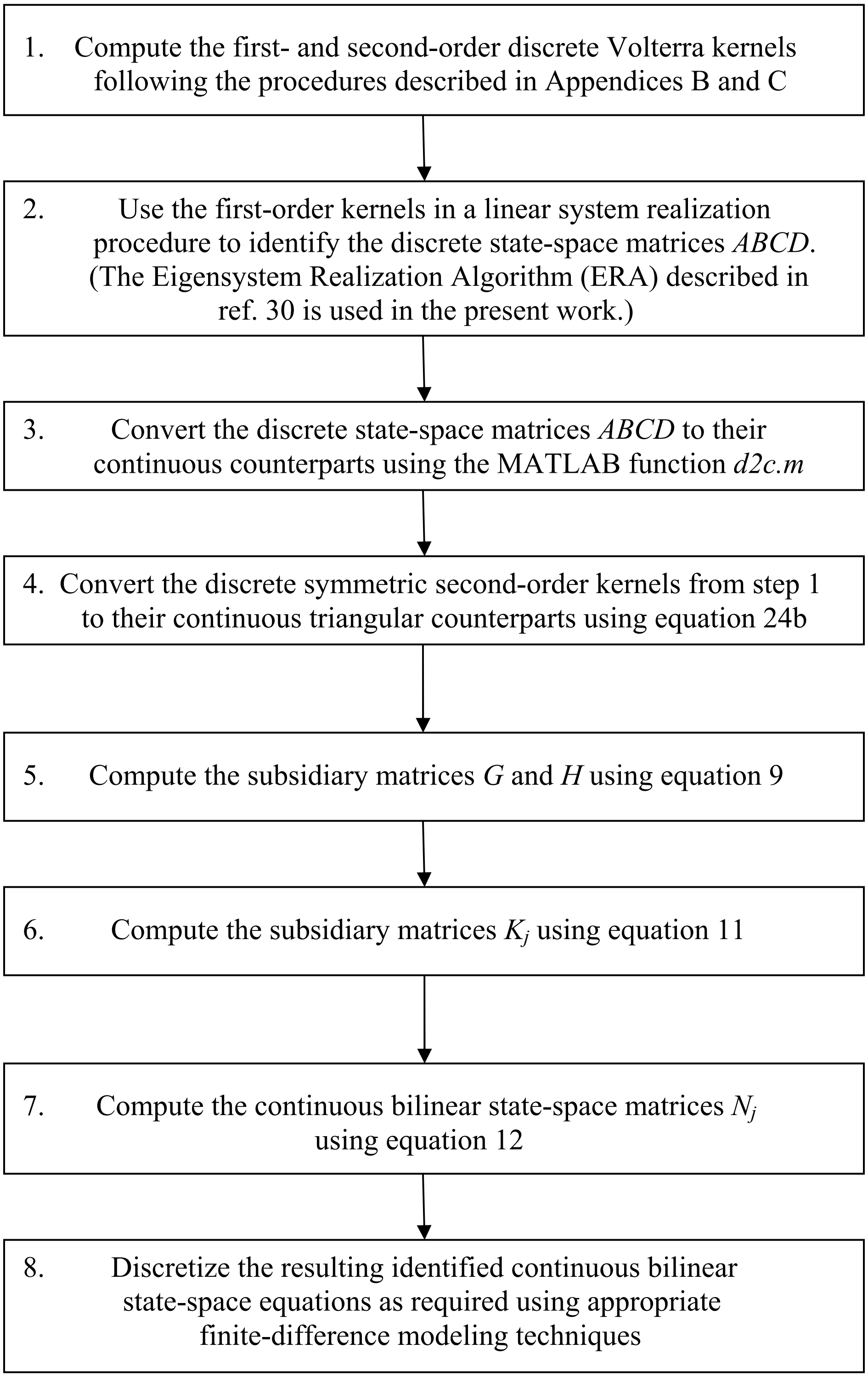


Figure 66.- Responses of linear, original bilinear, and identified bilinear equations to u_1 and u_2 inputs of figure 65.



1. Compute the first- and second-order discrete Volterra kernels following the procedures described in Appendices B and C

2. Use the first-order kernels in a linear system realization procedure to identify the discrete state-space matrices *ABCD*. (The Eigensystem Realization Algorithm (ERA) described in ref. 30 is used in the present work.)

3. Convert the discrete state-space matrices *ABCD* to their continuous counterparts using the MATLAB function *d2c.m*

4. Convert the discrete symmetric second-order kernels from step 1 to their continuous triangular counterparts using equation 24b

5. Compute the subsidiary matrices *G* and *H* using equation 9

6. Compute the subsidiary matrices K_j using equation 11

7. Compute the continuous bilinear state-space matrices N_j using equation 12

8. Discretize the resulting identified continuous bilinear state-space equations as required using appropriate finite-difference modeling techniques

Figure 67.- Block diagram indicating the sequence of tasks for identifying a discrete bilinear state-space representation of a nonlinear system using discrete Volterra kernels.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
01- 06 - 2008

2. REPORT TYPE
Technical Memorandum

3. DATES COVERED *(From - To)*

4. TITLE AND SUBTITLE
A Computational Procedure for Identifying Bilinear Representations of Nonlinear Systems Using Volterra Kernels

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Kvaternik, Raymond G.; and Silva, Walter A.

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER
984754.02.07.07.20.03

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Langley Research Center
Hampton, VA 23681-2199

8. PERFORMING ORGANIZATION REPORT NUMBER
L-19461

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSOR/MONITOR'S ACRONYM(S)
NASA

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
NASA/TM-2008-215320

12. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified - Unlimited
Subject Category 64
Availability: NASA CASI (301) 621-0390

13. SUPPLEMENTARY NOTES
An electronic version can be found at http://ntrs.nasa.gov

14. ABSTRACT

A computational procedure for identifying the state-space matrices corresponding to discrete bilinear representations of nonlinear systems is presented. A key feature of the method is the use of first- and second-order Volterra kernels (first- and second-order pulse responses) to characterize the system. The present method is based on an extension of a continuous-time bilinear system identification procedure given in a 1971 paper by Bruni, di Pillo, and Koch. The analytical and computational considerations that underlie the original procedure and its extension to the title problem are presented and described, pertinent numerical considerations associated with the process are discussed, and results obtained from the application of the method to a variety of nonlinear problems from the literature are presented. The results of these numerical studies are decidedly promising and provide sufficient credibility for further examination of the applicability of the method.

15. SUBJECT TERMS
Discrete Bilinear Systems; Volterra Kernels; Reduced-Order Modeling; Discrete Bilinear State-Space Models

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			STI Help Desk (email: help@sti.nasa.gov)
					19b. TELEPHONE NUMBER *(Include area code)*
U	U	U	UU	133	(301) 621-0390