Computationally Efficient Techniques For Discrete-time Realization Of Nonlinear Systems Represented By Volterra Series.

Order Number 9312320

Computationally efficient techniques for discrete-time realization of nonlinear systems represented by Volterra series

Garthwaite, Carl David, Ph.D.

Lehigh University, 1993

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

Computationally Efficient Techniques for Discrete-Time Realization of Nonlinear Systems Represented by Volterra Series

by

Carl David Garthwaite

Presented to the Graduate and Research Committee
of Lehigh University
in Candidacy for the Degree of
Doctor of Philosophy
in
Electrical Engineering

Lehigh University
December 1992

Approved and recommended for acceptance as a dissertation in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

12/9/92
Date

Douglas R. Frey
Dissertation Director

12/9/92
Accepted Date

Committee Members:

Bruce D. Fritchman

Meghanad D. Wagh

Richard T. Denton

Peter Hahn

Table of Contents

List of Figures

List of Tables

Abstract

The Volterra series provides a mathematically complete way to represent many realizable nonlinear systems. Predictive evaluations of nonlinear systems often rely on discrete-time simulations to determine satisfaction of design objectives. However, the Volterra series can be a cumbersome system representation from which to construct a discrete-time model. This dissertation develops and evaluates three computationally-efficient techniques for realizing the Volterra series representation of a nonlinear system in discrete-time form. The Bandlimited Volterra Series realization is potentially the least computationally expensive model, but determination of the filter coefficients constrains its usefulness. The Picard iteration realization does not preserve a good correspondance to the Volterra model and requires a significant increase in the sampling rate. The Serial Volterra series realization, which is constructed using only linear, one-dimensional filters, provides excellent accuracy in representing the response of a system. This is demonstrated for controlled inputs for which exact response determination is possible. Such validation of the technique provides a high degree of confidence that it will give accurate representations of the responses to inputs for which the exact responses cannot be readily determined.

CHAPTER 1
Introduction

This dissertation presents a study of computationally efficient, discrete-time techniques for determining the response of a nonlinear system to a specified input. Each of the techniques investigated is an implementation of the Volterra series or a form closely related to the Volterra series. The Volterra series has seen little application in the time-domain analysis of nonlinear systems due to the cumbersome nature of a direct realization of the technique. It is shown here that proper construction of a Volterra model for a system can lead to a practically useful tool for system analysis and optimization.

1.1 Motivation

Increasingly, in the development of complex systems, predictive performance evaluations rely on computer simulations to supplement closed-form analyses. Specifically, when such systems are inherently nonlinear to an extent which cannot be neglected - or when the nonlinear action of a system forms the basis for its utilization - a complete, closed-form mathematical analysis may be intractable. This is typical when the performance of a system can only be assessed for inputs which exhibit

complex spectra such as communication waveforms. In such cases, discrete-time simulation may be the preferred method for obtaining system performance estimates.

Nonlinear systems analysis, in any form, must account for the bandwidth-expanding character of these systems. In addition to the more complicated expressions for nonlinear system behavior (as compared to linear systems), the bandwidth expansion inherent to nonlinear processing makes accurate discrete-time processing computationally expensive. Consequently, discrete-time simulation of nonlinear systems will necessarily require some compromise between accuracy and compuational effort. Moreover, the computational effort may be distributed between model setup and execution such that the optimum approach for a given system may depend on the circumstances of the discrete-time processing to be performed.

1.2 Thesis

Given appropriate constraints on a nonlinear processing problem, it is possible to construct an efficient discrete-time computational model. Specifically, we consider nonlinear systems which may be described as weakly nonlinear; that is to say that the system is "nearly" linear, but not "nearly enough" to justify an assumption of strict linearity. In this case, Volterra series analysis may be advantageously applied to evaluate

the response of such a system to a bandlimited input. We investigate three specific techniques for developing discrete-time models which are more efficient than a brute-force discretization of the Volterra series.

The first technique is what we have called the Bandlimited Volterra Series. This technique assumes that components of the response which occur outside the portion of the spectrum occupied by the input are of no interest. This condition may be satisfied in instances where the system nonlinearity is undesirable, but must be considered for its contribution to response distortion. The distortion which occurs outside the input signal passband may be removed by filtering. Hence, when its elimination in discrete-time processing can be justified, a substantial processing economy may be obtained.

We have called the second efficient discrete-time processing technique the Serial Realization of the Volterra series. While this approach does not offer the benefits of bandlimiting the response, it provides a realization which requires only linear filters and memoryless product operators. The realization is constructed following the steps of a procedure which may be used to obtain the Volterra kernels.

A third method is developed following the Picard iteration technique proposed by Leon and Schaefer [1]. Its

implementation results in a ladder-type structure which allows a modular compuational organization.

Each of the techniques is assessed with regard to its computational complexity, or "cost" and accuracy. Depending on the particular assumptions or constraints of a particular system evaluation, one technique may be favored over the others.

1.3 Outline of the Dissertation

This dissertation is organized to first provide a theoretical basis for the Volterra series and an error model for assessing fidelity in nonlinear discrete-time signal processing. We follow this by presenting three specific techniques for realizing efficient processing. Throughout the theoretical development, the specific points addressed are illustrated by means of a simple, but effective circuit example which is carried through the various topics covered in the dissertation.

Chapter 2 presents a thorough review of the theory of the Volterra series. The important topic of convergence is addressed, and the dual time and frequency representations of nonlinear systems described by Volterra series are examined. Procedures for determining the Volterra kernels and their associated nonlinear transfer functions for analytically represented systems are described. The circuit example previously cited is introduced and both the

Volterra kernels and the nonlinear transfer functions for the circuit are determined to illustrate the techniques described.

Chapter 3 considers the various forms of approximation error which must be recognized and controlled in any discrete-time implementation of Volterra series-based signal processing. The theoretical basis for discrete-time approximation to the continuous-time Volterra series is presented and the approximation errors due to Volterra series truncation, aliasing, signal truncation, and filter representation are discussed in terms of appropriate error bounds. The chapter is concluded with the presentation of a composite error bound which incorporates each of individual errors described.

In *Chapter 4*, the Bandlimited Volterra Series is introduced. This technique provides a means of obtaining that part of the complete nonlinear system response which is contained within the passband of the input. The out-of-band portion of the complete response is rejected in exchange for a substantial reduction in computational burden. This approach may be particularly useful in applications where the nonlinear operation of the system is regarded as distortion and the out-of-band response components are removed by zonal filtering.

Chapter 5 presents the Serial Realization of the Volterra Series. It extends previous work [3] which has reported realizations of the Volterra series which are substantially serial realizations under special assumptions and constraints. While implementation of the Serial Realization of the Volterra Series is more cumbersome than linear discrete-time processing, it can provide a substantial computational savings over a direct realization of the Volterra series in discrete time. Moreover, the Serial Realization requires only linear filters which may be efficiently designed using familiar discrete-time techniques.

A discrete-time realization for nonlinear systems which is suggested by the Picard iteration technique is introduced in *Chapter 6*. The relationship identified by Leon and Schaefer [1] between the Picard iteration technique and the Volterra series points to this alternative implementation for discrete-time nonlinear processing. It offers a competitive approach to the Serial Realization of the Volterra Series and may be more efficient under some circumstances.

Chapter 7 provides a comparative analysis of the various discrete-time Volterra series techniques. The three approaches introduced in Chapters 4 through 6 are applied, for the circuit example developed in Chapter 2,

using sample inputs consisting of single and multiple sinusoids for which an exact response can also be determined by a frequency-domain analysis. The response of each computationally-efficient Volterra series realization is compared to the responses obtained using the other realizations. In addition, the computationally-efficient responses are compared to results obtained using a direct realization of the discrete-time Volterra series.

For the class of discrete-frequency inputs, of which the sinusoidal test signals are members, the exact responses for any finite-order Volterra series realization can be determined[1]. This provides a basis for assessing the accuracy of each realization. The results obtained for each computational model are evaluated in light of an error estimate obtained for that particular realization.

The conclusions of the dissertation regarding the benefits and drawbacks of nonlinear system realizations based on the Volterra series are presented in *Chapter 8*.

1.4 Contributions of the Dissertation

The material presented in Chapters 2 and 3 is largely based on a review of the literature; however, several parts represent extensions to earlier work. In Chapter 2, a

---

[1] While the exact response of an n^{th}-order Volterra system can be obtained for sinusoidal inputs, this is not generally the case. For the class of stochastic inputs, frequency-domain techniques cannot be applied. Therein lies the benefit of efficient discrete-time processing techniques.

clarification of and extension to Leon and Schaefer's technique for determination of the Volterra kernels is presented. In addition, the proofs related to the uniqueness of the response of a Volterra system under permutation of the arguments of an asymmetric kernel and the symmetry of the nonlinear transfer function of a symmetric kernel are original. Also, the state variable approach to treatment of systems containing multiple nonlinearities is new. In Chapter 3, the equivalences between continuous-time and discrete-time processing (subject to idealized restrictions) are established in a manner not believed to have been previously reported. Furthermore, the composite error bound is new.

The concept of the Bandlimited Volterra Series in Chapter 4 is a key, new result of this dissertation. Work performed earlier by Kim and Powers [2] made very similar assumptions regarding the control of aliasing for second-order ("quadratic") systems; however, it was focussed on system identification from discrete data sequences rather than on performance analyses of known systems.

Chapter 5 introduces the Serial Realization of the Volterra Series. This is a new presentation of the complete realization of a nonlinear system response represented by a Volterra series. It extends earlier work,

e.g. Shanmugam and Lal [3], but is not subject to the constraints which have previously applied with respect to the specific structure of the nonlinear (frequency-domain) transfer functions.

Chapter 6 presents a mechanization of the Picard iteration approach of Leon and Schaefer [1] for discrete-time implementation. The technique itself is not new, but its adaptation to discrete-time processing is original.

CHAPTER 2

The Volterra Series: A Review

The Volterra series forms the basis for the computationally efficient techniques for discrete-time evaluation of nonlinear systems which are discussed in this dissertation. Therefore, we first review the Volterra series: its applicability, determination, and associated nonlinear transfer functions.

2.1 Definition

Volterra series analysis is a mathematical technique for characterizing the behavior and determining the responses of nonlinear systems. It may be viewed as the multidimensional extension of concepts which are familiar in linear systems analysis. However, the technique has seen limited application due to the cumbersome nature of the higher order terms.

The name Volterra series derives from the work of the mathematician Vito Volterra who studied the representation of functionals during the late 19^{th} century [4]. The first application of Volterra series to systems analysis was by Norbert Wiener [5,6].

A conceptual advantage of the Volterra series representation for a nonlinear system is that it provides

an explicit representation for the response in terms of the input. The general expression for the response of a nonlinear system using the Volterra series is:

$$y(t) = \sum_{n=1}^{\infty} y_n(t) \tag{1a}$$

$$y_n(t) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\tau_1, \cdots, \tau_n) \prod_{i=1}^{n} [x(t-\tau_i) d\tau_i] \tag{1b}$$

where the input is $x(t)$ and the response is $y(t)$. The term $y_n(t)$ is the nonlinear response of order n, associated with the Volterra kernel of order n, $h_n(\tau_1, \ldots, \tau_n)$. It is formed by the n-linear operator indicated in equation (1b) [7].

While a Volterra series may be written with a constant y_0 term, such a term physically suggests the presence of a source internal to the system which produces an output, independent of any input. Such a term may be treated as an external source or eliminated using an incremental signal analysis. The inclusion of a constant term adds nothing to our understanding of the system and will be neglected in our analysis without loss of generality. However, in performing a system identification based on the input and output of a "black box" system, the zero order term, y_0, should be retained to account for unobservable sources within the system.

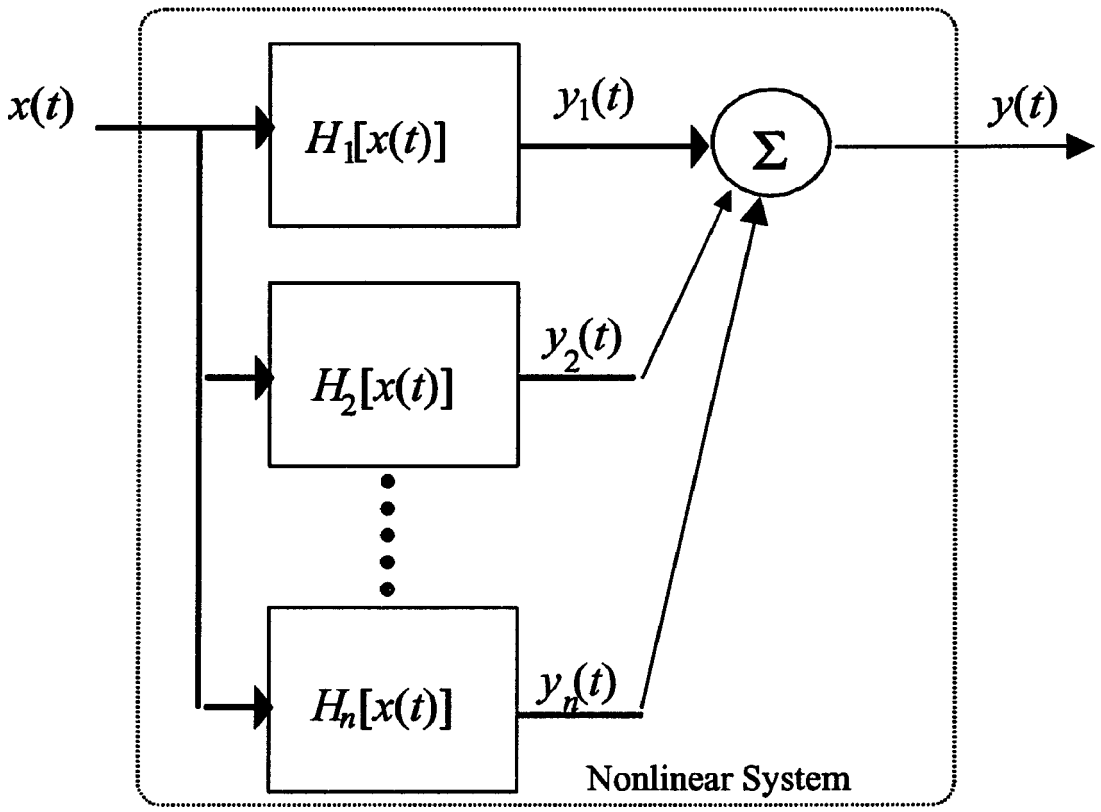


Figure 1: A Volterra Series Conceptual Realization

Conceptually, the Volterra series may be seen as a linear combination of terms of different orders. A convenient representation is shown in Figure 1. Each term may be computed independently, then the complete response formed by summing the individual component responses.

The first-order term in the Volterra series representation of a system is the familiar linear convolution integral for the response of a linear system.

It represents that component of the complete response of the system which would be considered by a linear systems analysis. Conversely, the response of a linear system may be expressed as a Volterra series in which all of the higher order terms (i.e., those for which $n > 1$) are identically zero.

The nonlinear terms of a Volterra series are successively higher order, multidimensional convolutions of input signal products with a Volterra kernel function of a corresponding number of variables. We use the expression "function" loosely here; Boyd [8] has shown that the Volterra kernels need not necessarily be functions in the strictest mathematical sense. They may also include Dirac δ functions; hence they may be distributions.

Volterra series representations can be obtained for many systems described by nonlinear differential or integral equations. Generally, these descriptions are implicit in the response variable and may have no apparent closed form solution. The corresponding Volterra series representations are explicit relations between input and output, although they contain an infinite number of terms. Typically, the terms of the Volterra series may be determined iteratively.

Nonlinear equation forms from which Volterra series may be obtained include n^{th}-order nonlinear differential equations such as:

$$x(t) = \sum_{r=0}^{n} a_r \frac{d^r}{dt^r}[y(t)] + \sum_{s=2}^{m} b_s [y(t)]^s \tag{2a}$$

and the Volterra integral equation:

$$y(t) = u(t) - \int_{-\infty}^{\infty} h(t-\tau) f[y(\tau)] d\tau \tag{2b}$$

In equations (2a) and (2b), $y(t)$ is the response of the system to the input $x(t)$. In equation (2b) (see Leon and Schaefer, [1]), the input is embodied in the $u(t)$ term which has the form:

$$u(t) = \int_{-\infty}^{\infty} g(t-\tau)\, x(\tau)\, d\tau \tag{2c}$$

The convolution kernel $g(t)$ in equation (2c) is related to the kernel $h(t)$ in equation (2b) in that their transfer functions, $G(s)$ and $H(s)$, share a common denominator.

While equations (2a) and (2b) are special cases of nonlinear differential and integral equations respectively, they are applicable to many circuits and systems of

practical interest. We will demonstrate this relationship in the context of an example in a following section.

2.2 Applicability

Realistically, all physical systems are nonlinear to some extent; materials utilized to fabricate devices ultimately exhibit saturation or breakdown, although they may have essentially linear regions of operation. Linear representations of such systems are first order approximations to their complete behavior (independent of whether or not the complete representation is known). Consequently, requirements for increased precision in system analyses must ultimately necessitate some treatment of the nonlinear aspects of the system.

The Volterra series (when it exists) for any system which contains an element characterized by a nonlinear current-voltage (I-V), flux-current (ϕ-I), or charge-voltage (Q-V) curve will contain an infinite number of terms. As a tool for evaluating a "real" system, however, the Volterra series is useful only when a small number of terms provides an acceptable representation of the true system response. Practically, the computations may become unwieldy if the number of terms exceeds 3 or 4.

2.3 Convergence

For some systems or nonlinear element characteristics, the Volterra series may not exist or may fail to converge for some inputs. That the Volterra series is a Taylor series applied to functions instead of values, suggests that the Volterra series for a system will exist and converge only if the associated Taylor series for the I-V, Q-V, or ϕ-I nonlinear element characteristic exists and is convergent [7]. Equivalently, the constitutive relationship of each nonlinear element must be memoryless and analytic [8].

In the event that a nonlinear component is *not* memoryless, further decomposition is typically possible. For example, a diode which has a non-negligible junction capacitance could be represented by a more detailed model than that of a voltage controlled conductance.

Commonly, applications of Volterra series analysis [9, 10] are restricted to "weakly" nonlinear systems (also described as systems with "mild" or "soft" nonlinearities). Such restrictions intuitively suggest satisfaction of conditions necessary for the Volterra series to exist and converge. Moreover, they also permit a series truncated to a small number of terms to satisfy the accuracy requirements of that particular analysis.

2.3.1 Radius of Convergence

Boyd [8] has presented a means for establishing a radius of convergence for a Volterra series based on what he has called the gain bound function. The gain bound function is defined to be:

$$f(z) = |h_0| + \sum_{n=1}^{\infty} \|h_n\| |z|^n \qquad \text{for} \quad |z| < \rho \tag{3}$$

where ρ is the radius of convergence of f, and $\|h_n\|$ is the norm of h_n defined as:

$$\|h_n\| = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} |h_n(\tau_1, \cdots, \tau_n)| \, d\tau_1 \cdots d\tau_n$$

The Volterra series is convergent for inputs, $x(t)$, which satisfy $|x(t)| < b$ for all t, where $0 < b \leq \rho$. In this case:

$$|y(t)| \leq f(b)$$

This can be seen from the following inequalities:

$$|y(t)| \leq |h_0| + \sum_{n=1}^{\infty} \left| \int_{-\infty}^{\infty} \cdots \int h_n(\tau_1, \cdots, \tau_n) x(t-\tau_1) \cdots x(t-\tau_n) d\tau_1 \cdots d\tau_n \right|$$

$$\leq |h_0| + \sum_{n=1}^{\infty} \int_{-\infty}^{\infty} \cdots \int |h_n(\tau_1, \cdots, \tau_n)| \, |x(t-\tau_1)| \cdots |x(t-\tau_n)| \, d\tau_1 \cdots d\tau_n$$

$$\leq |h_0| + \sum_{n=1}^{\infty} \int_{-\infty}^{\infty} \cdots \int |h_n(\tau_1, \cdots, \tau_n)| b^n d\tau_1 \cdots d\tau_n$$

$$= |h_0| + \sum_{n=1}^{\infty} \|h_n\| b^n = f(b) \tag{3a}$$

The radius of convergence for the gain bound function is:

$$\rho = \left(\limsup_{n \to \infty} \|h_n\|^{\frac{1}{n}} \right)^{-1} \tag{3b}$$

That this is true may be seen by replacing $\|h_n\|$ by a_n. Then, equation (3b) may be recognized as the radius of convergence of the power series:

$$f(x) = \sum_{n=1}^{\infty} a_n x^n$$

which converges for $|x| < \left[\lim_{n} \sup_{\to \infty} |a_n|^{\frac{1}{n}} \right]^{-1} = \rho$.

2.4 Frequency Domain Representation

In the time domain, Volterra series analysis of a system may be applied to any bounded input, including random processes. This applicability to random input signals may be particularly useful as an aid to evaluating the performance of nonlinear systems where the input has been corrupted by noise.

However, the applicability of Volterra series analysis is not restricted to the time domain. Multidimensional Fourier and Laplace transforms of the Volterra kernels may be found and are multivariable frequency domain functions known as kernel transforms [7] or nonlinear transfer functions [11].

The nonlinear transfer function of order n is defined to be:

$$H_n\left(f_1,\cdots,f_n\right)=\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}h_n(\tau_1,\cdots,\tau_n)\exp\left[-j2\pi\sum_{i=1}^{n}f_i\tau_i\right]d\tau_1\cdots d\tau_n \qquad (4)$$

where the variables f_i, $i=1,2,\cdots,n$ are coordinates in an n-dimensional frequency domain. The inverse transform, when it exists, has a similar form:

$$h_n(\tau_1,\cdots,\tau_n)=\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}H_n\left(f_1,\cdots,f_n\right)\exp\left[j2\pi\sum_{i=1}^{n}f_i\tau_i\right]df_1\cdots df_n \qquad (5)$$

The n-dimensional transform of the n^{th} order response provides useful insight into the character of the higher order responses. Before examining the n^{th} order response transform, let us define an n-dimensional n^{th} order response, $y_n(t_1,\ldots,t_n)$ as:

$$y_n(t_1,\cdots,t_n)=\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}h_n(\tau_1,\cdots,\tau_n)\prod_{i=1}^{n}[x(t_i-\tau_i)d\tau_i] \qquad (6)$$

This is related to the single dimensional n^{th} order response given in equation (1b) by:

$$y_n(t) = y_n(t_1, \cdots, t_n)\big|_{t=t_1=\cdots=t_n}$$

The n-dimensional transform of the multivariate time response is:

$$Y_n\left(f_1, \cdots, f_n\right) = \int_{-\infty}^{\infty} \cdots \int y_n(t_1, \cdots, t_n) \exp\left(-j2\pi \sum_{i=1}^{n} t_i f_i\right) dt_1 \cdots dt_n \qquad (7)$$

Substituting equation (6) into equation (7) gives an equivalent expression in terms of the input and the Volterra kernel:

$$Y_n\left(f_1, \cdots, f_n\right) = \int_{-\infty}^{\infty} \cdots \int \left[\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\tau_1, \cdots, \tau_n) \prod_{i=1}^{n} [x(t_i - \tau_i) d\tau_i] \right] \exp\left(-j2\pi \sum_{i=1}^{n} t_i f_i\right) dt_1 \cdots dt_n \qquad (8)$$

A rearrangement of the terms in equation (8) including an interchange of the order of integration yields:

$$Y_n\left(f_1, \cdots, f_n\right) = \int_{-\infty}^{\infty} \cdots \int h_n(\tau_1, \cdots, \tau_n) \left[\prod_{i=1}^{n} \int_{-\infty}^{\infty} x(t_i - \tau_i) \exp\left(-j2\pi f_i t_i\right) dt_i \right] d\tau_1 \cdots d\tau_n \qquad (9)$$

Evaluation of the inner integrals as Fourier transforms of the τ_i-delayed replicas of the input signal gives:

$$Y_n\left(f_1,\cdots,f_n\right)=\int_{-\infty}^{\infty}\cdots\int h_n(\tau_1,\cdots,\tau_n)\left[\prod_{i=1}^{n}X\left(f_i\right)\exp\left(-j2\pi f_i\tau_i\right)\right]d\tau_1\cdots d\tau_n \qquad (10)$$

from which we may immediately obtain a result analogous to the frequency domain expression for linear systems:

$$Y_n\left(f_1,\cdots,f_n\right)=H_n\left(f_1,\cdots,f_n\right)X\left(f_1\right)\cdots X\left(f_n\right) \qquad (11)$$

The relationship between the n^{th} order multidimensional frequency domain response and the Fourier transform of the n^{th} order response can be established as shown below. We begin by writing:

$$y_n(t)=\int_{-\infty}^{\infty}\cdots\int Y_n\left(f_1,\cdots,f_n\right)\exp\left(j2\pi t\sum_{i=1}^{n}f_i\right)df_1\cdots df_n \qquad (12)$$

Then by taking the Fourier transform (1-dimensional) of both sides, we obtain:

$$Y_n\left(f\right)=\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}\cdots\int Y_n\left(f_1,\cdots,f_n\right)\exp\left(j2\pi t\sum_{i=1}^{n}f_i\right)df_1\cdots df_n\right]\exp\left(-j2\pi ft\right)dt \qquad (13)$$

Interchanging the order of integrations and evaluating the inner integral as:

$$\int_{-\infty}^{\infty}\exp\left(j2\pi t\sum_{i=1}^{n}f_i\right)\exp\left(-j2\pi ft\right)dt=\delta\left(f-\sum_{i=1}^{n}f_i\right) \qquad (14)$$

immediately yields:

$$Y_n(f) = \int_{-\infty}^{\infty} \cdots \int Y_n(f_1, \cdots, f_n)\, \delta\left(f - \sum_{i=1}^{n} f_i\right) df_1 \cdots df_n \qquad (15)$$

By substituting equation (11) into equation (15), it can be seen that the n^{th} order response to a set of inputs at frequencies $f_1, \ldots, f_n$ occurs at the sum frequency, $f=f_1+\ldots+f_n$.

$$Y_n(f) = \int_{-\infty}^{\infty} \cdots \int H_n(f_1, \cdots, f_n) X(f_1) \cdots X(f_n)\, \delta\left(f - \sum_{i=1}^{n} f_i\right) df_1 \cdots df_n \qquad (16)$$

This interpretation of the result is evident from the fact that $\delta\left(f - \sum_{i=1}^{n} f_i\right) = 0$ unless $f = \sum_{i=1}^{n} f_i$. Accordingly, we may state that the value of $H_n(f_1, \ldots, f_n)$ is the weight of the n^{th} order response, at frequency $f_1+\ldots+f_n$, to a set of unit amplitude input components at angular frequencies $f_1, \ldots, f_n$. (The complete n^{th} order response at $f=f_1+\ldots+f_n$ will be $n!$ times this weight due to the $n!$ permutations of the f_i's which correspond to orderings of the arguments of H_n that produce a response to the given input.)

2.4.1 Example of a Frequency Domain Response Computation

As an example, consider an input to a nonlinear system, $x(t)$, given by:

$$x(t) = A_1 \exp\left(j2\pi\zeta_1 t\right) + A_2 \exp\left(j2\pi\zeta_2 t\right) + \cdots + A_n \exp\left(j2\pi\zeta_n t\right) \tag{17a}$$

The Fourier transform of this input signal is:

$$X\left(f\right) = A_1\delta\left(f-\zeta_1\right) + A_2\delta\left(f-\zeta_2\right) + \cdots + A_n\delta\left(f-\zeta_n\right) \tag{17b}$$

Therefore, when equation (17b) is used in equation (16), we obtain:

$$Y_n\left(f\right) = \int_{-\infty}^{\infty}\cdots\int H_n\left(f_1,\cdots,f_n\right)A_1\delta\left(f_1-\zeta_1\right)\cdots A_n\delta\left(f_n-\zeta_n\right)\delta\left(f-\sum_{i=1}^{n}f_i\right)df_1\cdots df_n \tag{18}$$

Evaluating the integrals results in:

$$Y_{p,n}\left(\zeta_1+\cdots+\zeta_n\right) = \prod_{i=1}^{n}A_i\,H\left(\zeta_1,\cdots,\zeta_n\right) \tag{19}$$

The p subscript has been added to the response in equation (19) to indicate that it is only a partial n^{th}-order response. The complete n^{th}-order response is determined by summing the output components due to each combination and permutation of all input components. In particular, note that there are $n!$ permuations of the input

components at the n distinct frequencies which comprise the input in this example. Therefore, the response at $f=f_1+...+f_n$ is $n!$ times the value obtained by equation (19). Furthermore, we have implicitly assumed that each of the n complex exponential components in this example has a frequency which cannot be synthesized as a linear combination of the others; i.e., that the frequencies are incommensurate. Otherwise, n^{th}-order response components at $f=f_1+...+f_n$ might also be obtained from other combinations of the input components.

Expressed in the time domain, the n^{th}-order response component at $f=f_1+...+f_n$ due to the input described by equations (17) is:

$$y_n(t) = n!\left[\prod_{i=1}^{n} A_i\right] H_n\left(f_1, \cdots, f_n\right) \exp\left(j2\pi t \sum_{i=1}^{n} f_i\right) \qquad (20)$$

While the creation of response components at only the sum frequencies may appear contrary to the common wisdom that nonlinear response components occur at sum and difference frequencies, it should be noted that we have considered only non-physical complex exponential inputs. Physically, we deal with real sinusoidal signals. When these signals are viewed as the composition of positive and negative frequency complex exponentials, the response components at

"differences" of the sinusoidal frequencies are realized as a natural consequence of sums of the complex exponential frequencies.

In equation (20) we have neglected any response contributions at the sum frequency due to different combinations of the input components; in general these all exist and form part of the complete response. The A_i are complex coefficients incorporating both the amplitudes and phases of the input components.

2.4.2 Application of Frequency Domain Volterra Analysis

In general, frequency domain analysis of nonlinear systems by Volterra series is difficult for complex or stochastic inputs. However, for low order systems a frequency domain analysis can readily be performed when the input contains distinct frequency components.

In the literature, a more frequent application of Volterra series analysis is seen using frequency domain representations of the systems considered [11,12,13], although time domain analysis is also found [14]. By far, however, the greatest number of published papers are oriented toward computations based on greatly simplified models [15,16,17]. Perhaps this is largely due to what Hummels and Gitchell have called "the computational burden always associated with obtaining numerical results with a Volterra systems approach" [15].

2.5 Determination of the Volterra Kernels

In practice, the identification of the Volterra kernels for a "black-box" system can be difficult. Analytical derivation of the kernels is tedious, but can be performed in a straightforward manner for a variety of circuits and systems. Bedrosian and Rice describe methods for obtaining both the Volterra kernels and the kernel transforms from a variety of nonlinear system descriptions [18]. Bussgang, et. al. have outlined a procedure for deriving the Volterra kernels from the nonlinear differential equation [11]. Leon and Schaefer demonstrate a comparable procedure for obtaining the Volterra kernels from the nonlinear Volterra integral equation [1].

In addition, system identification techniques for determining discrete-time approximations to the Volterra kernels of an unknown system have been proposed utilizing correlation techniques [14].

The most common approach for characterizing physical nonlinear systems, however, has clearly been determination of the nonlinear transfer functions using multifrequency probing techniques [9]. This approach is, perhaps, the most intuitively appealing method for characterizing physical systems in the laboratory as well.

2.5.1 Symmetry

The Volterra kernels obtained for a specific system by a particular method may or may not be symmetric with respect to permutations of their arguments. We say that a Volterra kernel is symmetric if:

$$h_n\left(\tau_1,\cdots,\tau_i,\cdots,\tau_j,\cdots,\tau_n\right)=h_n\left(\tau_1,\cdots,\tau_j,\cdots,\tau_i,\cdots,\tau_n\right),\quad i,j\in\{1,\cdots,n\} \tag{21}$$

Thus, by different choices of variable assignment, we may obtain as many as $n!$ different n^{th}-order kernels when an asymmetric form is obtained. (Since there is only one permutation of a single argument, the issue of symmetry in the present sense is non-existent for linear systems.)

2.5.1.1 The Response to an Asymmetric Kernel

Although the n^{th}-order Volterra kernel for a given system may not be unique, the responses of all kernels obtained by permutation of arguments are identical. We prove this below.

Proof: Assume that the n^{th}-order Volterra kernel, h_n, for some system is not symmetric. The n^{th}-order response of this system is given by:

$$y_n(t) = \int_{-\infty}^{\infty}\cdots\int h_n\left(\tau_1,\cdots,\tau_i,\cdots,\tau_j,\cdots,\tau_n\right)x(t-\tau_1)\cdots x(t-\tau_i)\cdots x(t-\tau_j)\cdots x(t-\tau_n) \times d\tau_1\cdots d\tau_i\cdots d\tau_j\cdots d\tau_n \quad (22)$$

Since the τ_k are dummy variables, which are eliminated by integration, we may rename them without changing the response in any way. Thus if we choose $\tau_i=a$ and $\tau_j=\alpha$, we may rewrite equation (22) as:

$$y_n(t) = \int_{-\infty}^{\infty}\cdots\int h_n(\tau_1,\cdots,a,\cdots,\alpha,\cdots,\tau_n)x(t-\tau_1)\cdots x(t-a)\cdots x(t-\alpha)\cdots x(t-\tau_n) \times d\tau_1\cdots da\cdots d\alpha\cdots d\tau_n \quad (23)$$

Let us obtain a different n^{th}-order Volterra kernel by permutation of the arguments t_i and t_j. We define:

$$\hat{h}_n\left(\tau_1,\cdots,\tau_i,\cdots,\tau_j,\cdots,\tau_n\right) = h_n\left(\tau_1,\cdots,\tau_j,\cdots,\tau_i,\cdots,\tau_n\right) \quad (24)$$

Let us write the response to this new kernel as:

$$\hat{y}_n(t) = \int_{-\infty}^{\infty}\cdots\int \hat{h}_n\left(\tau_1,\cdots,\tau_i,\cdots,\tau_j,\cdots,\tau_n\right)x(t-\tau_1)\cdots x(t-\tau_i)\cdots x(t-\tau_j)\cdots x(t-\tau_n) \times d\tau_1\cdots d\tau_i\cdots d\tau_j\cdots d\tau_n \quad (25)$$

Substituting equation (24) into equation (25), we obtain:

$$\hat{y}_n(t) = \int_{-\infty}^{\infty}\cdots\int h_n\left(\tau_1,\cdots,\tau_j,\cdots,\tau_i,\cdots,\tau_n\right)x(t-\tau_1)\cdots x(t-\tau_i)\cdots x(t-\tau_j)\cdots x(t-\tau_n) d\tau_1\cdots d\tau_i\cdots d\tau_j\cdots d\tau_n \quad (26)$$

As before, the t_k are dummy variables which we may rename as we choose. Let us assign: $\tau_i=\alpha$ and $\tau_j=a$ (the reverse of our previous choice). Then:

$$\hat{y}_n(t)=\int_{-\infty}^{\infty}\cdots\int h_n(\tau_1,\cdots,a,\cdots,\alpha,\cdots,\tau_n)x(t-\tau_1)\cdots x(t-\alpha)\cdots x(t-a)\cdots x(t-\tau_n) \times d\tau_1\cdots d\alpha\cdots da\cdots d\tau_n \quad (27)$$

The product of delayed replicas of the input signal is commutative, so that the terms may be rewritten in any order. Furthermore, we may re-order the integrations to obtain:

$$\hat{y}_n(t)=\int_{-\infty}^{\infty}\cdots\int h_n(\tau_1,\cdots,a,\cdots,\alpha,\cdots,\tau_n)x(t-\tau_1)\cdots x(t-a)\cdots x(t-\alpha)\cdots x(t-\tau_n) \times d\tau_1\cdots da\cdots d\alpha\cdots d\tau_n \quad (28)$$

The right hand side of equation (28) is identical to the right hand side of equation (23). Therefore:

$$y_n(t)=\hat{y}_n(t) \quad (29)$$

This process may, in principle, be repeated as often as we like. Consequently, the n^{th}-order response, $y_n(t)$, is the same for any permutation of the arguments of h_n, whether or not the kernel is symmetric.

2.5.1.2 Symmetrization

Since there are $n!$ permutations of the arguments $\{t_1,\ldots,t_n\}$ of an n^{th}-order Volterra kernel, each of which yields the identical response to an input, we may define a symmetric kernel [7,11]:

$$S[h_n(\tau_1,\cdots,\tau_n)] = \frac{1}{n!}\sum_{\{\tau_i\}} h_n\left(\tau_{i_1},\cdots,\tau_{i_n}\right) \tag{30}$$

where the S means "the symmetrization of" the quantity in brackets. The symmetrized kernel is unique, and no loss of generality results from treating arbitrary kernels as symmetric. Since physical systems have no preferred permutation of arguments - the assignment of which is made for the convenience of analysis - it is often helpful to define kernels which are insensitive to an exchange of arguments.

2.5.1.3 Symmetric Nonlinear Transfer Functions

Following an approach similar to that in section 2.5.1.1, it may be shown that the n-dimensional nonlinear transfer function of a symmetric Volterra kernel is also a symmetric function of its arguments. The proof is outlined below.

Let $h_n(\tau_1,\ldots,\tau_n)$ be a symmetric Volterra kernel. The corresponding nonlinear transfer function is:

$$H_n\left(f_1,\cdots,f_k,\cdots,f_l,\cdots,f_n\right)=\int_{-\infty}^{\infty}\cdots\int h_n(\tau_1,\cdots,\tau_i,\cdots,\tau_j,\cdots,\tau_n)$$

$$\exp\left\{-j2\pi\left(f_1\tau_1+\cdots+f_i\tau_i+\cdots+f_j\tau_j+\cdots+f_n\tau_n\right)\right\}d\tau_1\cdots d\tau_n \tag{31a}$$

Due to the symmetry of the kernel $h_n(\tau_1,\ldots,\tau_n)$, we may equivalently write:

$$H_n\left(f_1,\cdots,f_k,\cdots,f_l,\cdots,f_n\right)=\int_{-\infty}^{\infty}\cdots\int h_n(\tau_1,\cdots,\tau_j,\cdots,\tau_i,\cdots,\tau_n)$$

$$\exp\left\{-j2\pi\left(f_1\tau_1+\cdots+f_i\tau_i+\cdots+f_j\tau_j+\cdots+f_n\tau_n\right)\right\}d\tau_1\cdots d\tau_n \tag{31b}$$

or:

$$H_n\left(f_1,\cdots,f_k,\cdots,f_l,\cdots,f_n\right)=\int_{-\infty}^{\infty}\cdots\int h_n(\tau_1,\cdots,\tau_j,\cdots,\tau_i,\cdots,\tau_n)$$

$$\exp\left\{-j2\pi\left(f_1\tau_1+\cdots+f_i\tau_j+\cdots+f_j\tau_i+\cdots+f_n\tau_n\right)\right\}d\tau_1\cdots d\tau_n \tag{31c}$$

where in equation (31c) the transform has been written for the permuted-index kernel.

The permuted-index nonlinear transfer function can be written with respect to equation (31a) as:

$$H_n\left(f_1,\cdots,f_l,\cdots,f_k,\cdots,f_n\right)=\int_{-\infty}^{\infty}\cdots\int h_n(\tau_1,\cdots,\tau_i,\cdots,\tau_j,\cdots,\tau_n)$$

$$\exp\left\{-j2\pi\left(f_1\tau_1+\cdots+f_l\tau_k+\cdots+f_k\tau_l+\cdots+f_n\tau_n\right)\right\}d\tau_1\cdots d\tau_n \tag{32}$$

Setting $k=i$ and $l=j$ and permuting the indices of h_n, as permitted by its symmetry, we have:

$$H_n\left(f_1,\cdots,f_j,\cdots,f_i,\cdots,f_n\right)=\int_{-\infty}^{\infty}\cdots\int h_n(\tau_1,\cdots,\tau_j,\cdots,\tau_i,\cdots,\tau_n)$$
$$\exp\left\{-j2\pi\left(f_1\tau_1+\cdots+f_j\tau_i+\cdots+f_i\tau_j+\cdots+f_n\tau_n\right)\right\}d\tau_1\cdots d\tau_n \tag{33}$$

Now the right hand sides of equations (33) and (31c) are identical; therefore:

$$H\left(f_1,\cdots,f_i,\cdots,f_j,\cdots,f_n\right)=H\left(f_1,\cdots,f_j,\cdots,f_i,\cdots,f_n\right) \tag{34}$$

and, hence, the nonlinear transfer functions are symmetric.

2.5.2 Determination of the Volterra Kernels for a Nonlinear System from the Volterra Integral Equation

The paper by Leon and Schaefer [1] provides a method by which the Volterra kernels of all orders may be determined for a nonlinear system when a representation of the system by a Volterra integral equation is known. The requirements of their method are that one have a Volterra integral representation of the nonlinear system, and that there be a polynomial approximation to the nonlinear element's constitutive relationship. However, the paper stops short of recognizing that each higher order kernel has an expression in the form of a multidimensional convolution of products of the [associated] linear kernel of the system. We show below, in terms of an example, how the first several terms are obtained. Further, we provide a generalized procedure for determining the higher order terms which shows the relationships that the Volterra kernels of all orders bear to the linear portion of the system.

We begin by introducing a simple nonlinear circuit which will serve as a convenient vehicle to illustrate the derivation of a Volterra series. Next, we restate the method proposed by Leon and Schaefer [1] for completeness. Then we derive the Volterra kernels for our example circuit, providing a convenient reference for our work in

discretization of the Volterra series and in so doing, expand on the earlier work of Leon and Schaefer.

2.5.2.1 A Simple Nonlinear Circuit Example

The semiconductor diode is one of the most common nonlinear devices in modern circuits. Using the exponential characterization of the voltage controlled conductance relationship of a diode, a simple RC circuit with a diode across the capacitor provides a convenient and illuminating example for the study of Volterra series. The circuit which we will consider is shown in Figure 2.

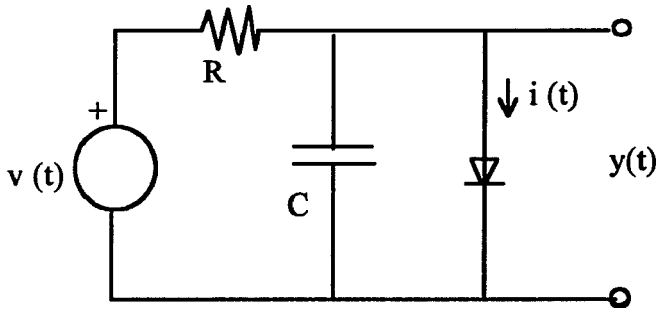


Figure 2: Nonlinear Circuit Example

We will utilize the exponential constitutive relationship for the diode:

$$i_d = I_s\left(e^{\lambda v_d} - 1\right) \tag{35}$$

where i_d and v_d are the current through and the voltage across the diode, respectively. I_s represents the reverse saturation current and λ is the inverse thermal voltage constant of the diode. No time dependence has been shown; we assume that the diode behaves as a memoryless voltage controlled conductance.

Expanding the exponential constitutive relationship for the diode in a power series, we obtain:

$$i_d = I_s \lambda v_d + \sum_{n=2}^{\infty} \frac{I_s \lambda^n}{n!} v_d^n = g_l v_d + f(v_d). \tag{36}$$

The power series expansion in equation (36) lends itself to an interpretation of the diode as a linear conductance, g_l, in parallel with a strictly nonlinear (i.e., having a polynomial representation which has no constant or linear terms) voltage controlled conductance, $f(v_d)$. It will be convenient to redraw the circuit in this fashion, also performing a Thevenin to Norton conversion of the voltage source and source resistance, as shown in Figure 3.

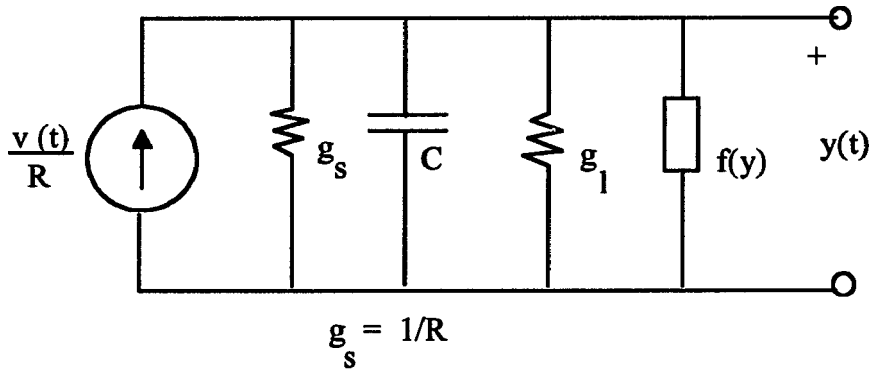


Figure 3: Redrawn Nonlinear Circuit

The linear circuit elements may be combined to form a single linear impedance, $H(s)$, expressed in the Laplace transform s-domain:

$$H(s)=\left[\frac{1}{R}+sC+g_l\right]^{-1}=\left[\frac{1}{R}+I_s\lambda+sC\right]^{-1} \tag{37}$$

Reverting to the Thevenin form of the circuit, we have the equivalent circuit shown in Figure 4. This shows the original circuit as a series-connected voltage source, a strictly linear impedance, and a strictly nonlinear, voltage-controlled conductance.

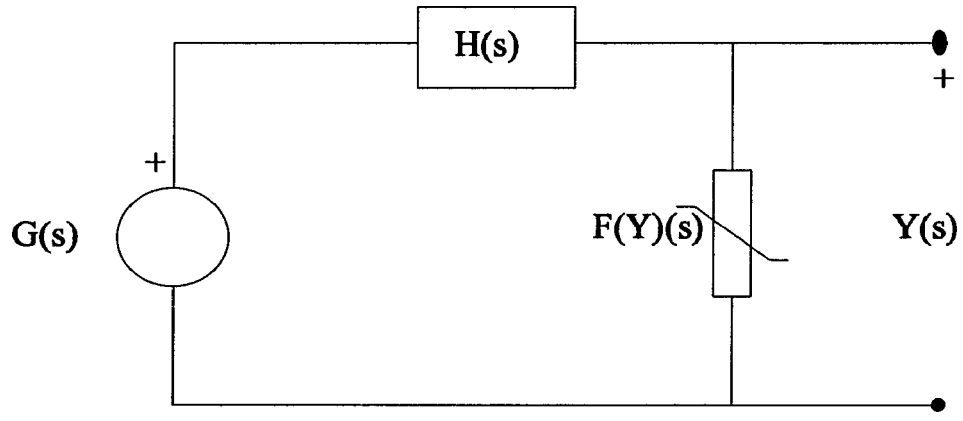


Figure 4: Thevenin Equivalent Nonlinear Circuit

In Figure 4, $G(s)$ is a modified voltage source related to the original source $V_s(s)$ by:

$$G(s) = \frac{1}{R} H(s) V_s(s) \tag{38}$$

We can now write a simple expression for the response, $Y(s)$, of the circuit (where we have substituted the customary notation, $Y(s)$, for the response in place of the diode voltage, $V_d(s)$).

$$Y(s) = G(s) - H(s)F(Y)(s) \tag{39}$$

where $F(Y)$ indicates the (nonlinear) operator on the transform domain representation of the response which is implied by the Laplace transform of the circuit equation.

The difficulty with this expression is that we have not obtained the nonlinear operator in the transform domain. While the form of the F operator can be determined, it is not needed to obtain the time-domain Volterra kernels. Instead, we revert to the time domain, where we have an expression for $f(y)$, and interpret the s-domain product of H and $F(Y)$ as a convolution:

$$y(t) = g(t) - \int_{-\infty}^{\infty} h(t-\tau) f[y(\tau)] d\tau \tag{40}$$

The convolution kernel, $h(t)$, is the inverse Laplace transform of $H(s)$ and may be written explicitly as:

$$h(t) = \frac{1}{C} \exp[-kt] u(t) \tag{41}$$

where the unit step function, $u(t)$, expresses the causality of $h(t)$, and the reciprocal time constant of the circuit, k, is given by:

$$k = \frac{1}{C}\left[\frac{1}{R} + I_s \lambda\right] \tag{42}$$

Based on our previous determination of the modified source, $g(t)$, we may write:

$$y(t) = \int_{-\infty}^{\infty} \frac{1}{R} h(t-\tau) v_s(\tau) d\tau - \int_{-\infty}^{\infty} h(t-\tau) f[y(\tau)] d\tau \tag{43}$$

2.5.2.2 Derivation of the Volterra Kernels from the Volterra Integral Equation

First, consider the Volterra integral representation of a nonlinear system:

$$y(t) = g(t) - \int_{t_0}^{t} h(t-\tau) f[y(\tau)] d\tau \tag{44}$$

where: $f(x) = \sum_{i=2}^{\infty} a_i x^i$ (44a)

Leon and Schaefer [1] restrict this infinite series expansion to a degree m polynomial; however, we prefer to allow this to remain an infinite series representation of the nonlinearity and permit the truncation to be a matter of practical convenience rather than conceptual restriction. In addition, we observe that the upper limit of integration, t, is a consequence of the causality of the kernel $h(t)$. Therefore, we may replace the limit by ∞ without loss of generality. Furthermore, while solutions to the Volterra integral equation may not be obtainable by the method of Picard when the lower limit is replaced by $-\infty$ that is in fact the most general condition for which we seek a solution. We shall make the substitution $t_0 = -\infty$, with the recognition that it also is not a practical concern for causal inputs, $v_s(t)$.

Second, we write the (as yet unknown) Volterra series expression for that same system:

$$y(t) = \sum_{i=1}^{\infty} y_i(t) \tag{45}$$

Next, we substitute the right hand side of equation (45) into both sides of equation (44). This yields:

$$\sum_{i=1}^{\infty} y_i(t) = g(t) - \int_{-\infty}^{\infty} h(t-\tau) f\left[\sum_{i=1}^{\infty} y_i(\tau)\right] d\tau \tag{46}$$

The important realization by Leon and Schaefer was that the integral on the right hand side of equations (44) and (46) can contain no linear (i.e. first order Volterra series) component. Hence, g(t) must be the entire linear part of the complete system response, i.e. $y_1(t)$. Therefore, a recursive procedure can be established for successively determining the higher order Volterra kernels in equation (45).

Leon and Schaefer point out that the only second-order contribution must be that which results from squaring the now-known first order term. Consequently, we have:

$$y_2(t) = -\int_{-\infty}^{\infty} h(t-\tau)\, a_2[y_1(\tau)]^2 d\tau \tag{47}$$

Since we know that $y_1(t) = g(t)$, we may substitute this result into equation (47) to determine the second order Volterra kernel. The Leon and Schaefer result is:

$$h_2(\tau_1,\tau_2) = -a_2 h(\tau_1)\delta(\tau_1 - \tau_2) \tag{48}$$

[A remark regarding our notation is appropriate. We have denoted the strictly linear kernel (impulse response) of the Volterra integral equation by the unsubscripted $h(t)$. The (also linear) first order Volterra kernel is denoted $h_1(\tau_1)$ and, not accidentally, bears a functional resemblance to $h(t)$. The higher order Volterra kernels are subscripted according to their order.]

2.5.2.3 Extension of Leon and Schaefer's Method

It is at this point which we shall depart from the Leon and Schaefer work. We recognize from our example that $g(t)$ is a modified source and has an expression as a convolution of the original source, $v_s(t)$, with the impulse response of the linear part of the circuit.

For a source which has a known Fourier transform, $G(f)$, the waveform, $g(t)$, may be explicitly determined. However, for a stochastic source, $g(t)$ may only be expressible as a convolution. Therefore, we want to represent $g(t)$ in its

more general form. In terms of the original voltage source, we have:

$$y_1(t) = \frac{1}{R}\int_{-\infty}^{\infty} h(t-\tau_1)v_s(\tau_1)d\tau_1 = \frac{1}{R}\int_{-\infty}^{\infty} h(\tau_1)v_s(t-\tau_1)d\tau_1 \qquad (49)$$

Thus the first order Volterra kernel is:

$$h_1(\tau_1) = \frac{1}{R}h(\tau_1) \qquad (50)$$

Substituting equation (49) into equation (47) and using separate dummy variables in each term, we obtain:

$$y_2(t) = -\int_{-\infty}^{\infty} h(\tau)\, a_2 \frac{1}{R}\int_{-\infty}^{\infty} h(\tau_1-\tau)v_s(t-\tau_1)d\tau_1 \frac{1}{R}\int_{-\infty}^{\infty} h(\tau_2-\tau)v_s(t-\tau_2)d\tau_2\, d\tau \qquad (51)$$

Interchanging the order of integration and rearranging terms yields:

$$y_2(t) = -\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} a_2\frac{1}{R^2}h(\tau)h(\tau_1-\tau)h(\tau_2-\tau)d\tau\, v_s(t-\tau_1)v_s(t-\tau_2)d\tau_1\, d\tau_2 \qquad (52)$$

Now, instead of being one integration "short" of the desired form for the second order Volterra series as in equation (47), we in fact have one integration too many. Clearly, in order to satisfy the Volterra series form requirement, we must carry out the integration with respect to τ. Consequently, we have:

$$h_2(\tau_1,\tau_2) = -\int_{-\infty}^{\infty} a_2 \frac{1}{R^2} h(\tau)h(\tau_1-\tau)h(\tau_2-\tau)d\tau \qquad (53)$$

Using the previously obtained result for $h(t)$, i.e.:

$$h(t) = \frac{1}{C}\exp(-kt)u(t) \qquad (54)$$

where $u(t)$ is the unit step function:

$$u(t) = \begin{cases} 1, & t>0 \\ 0, & t<0 \end{cases}$$

We can state the first and second order Volterra kernels explicitly. Substituting equation (54) into equation (50) yields:

$$h_1(\tau_1) = \frac{1}{RC}\exp(-k\tau_1)u(\tau_1) \qquad (55)$$

Substituting equation (54) into equation (53) in three places gives:

$$h_2(\tau_1,\tau_2) = \frac{-a_2}{C^3R^2}\int_{-\infty}^{\infty} \exp[-k\tau]\exp[-k(\tau_1-\tau)]\exp[-k(\tau_2-\tau)]u(\tau)u(\tau_1-\tau)u(\tau_2-\tau)d\tau \qquad (56)$$

Reflecting the unit step functions in the limits of integration, we obtain:

$$h_2(\tau_1,\tau_2) = \frac{-a_2}{C^3R^2}\int_0^{\min\{\tau_1,\tau_2\}} \exp[-k\tau]\exp[-k(\tau_1-\tau)]\exp[-k(\tau_2-\tau)]d\tau \qquad (57)$$

Evaluating the integral results in:

$$h_2(\tau_1,\tau_2) = \frac{a_2}{C^3R^2k}\{\exp[-k\tau_1]\exp[-k\tau_2] - \exp[-k\max\{\tau_1,\tau_2\}]\}u(\tau_1)u(\tau_2) \qquad (58)$$

where the $u(\tau_1)$, $u(\tau_2)$ conditions are derived from the fact that if $\tau<\tau_1,\tau_2$ there is no region of integration in equation (57).

The reader is cautioned that care must be exercised in determining the sense of the $\min\{\tau_1,\tau_2\}$ and $\max\{\tau_1,\tau_2\}$ expressions. The upper limit of integration, $\min\{\tau_1,\tau_2\}$, in equation (57) is determined by the unit step function which most restricts the region of integration. That limit cancels the corresponding term in the integrated result, leaving the $\max\{\tau_1,\tau_2\}$ expression in equation (58). An alternative form of equation (58) which may be useful is:

$$h_2(\tau_1,\tau_2) = \frac{a_2}{C^3R^2k}\{\exp[-k(\tau_1+\tau_2)] - \exp[-k\tau_1]u(\tau_1-\tau_2) - \exp[-k\tau_2]u(\tau_2-\tau_1)\}$$
$$u(\tau_1)u(\tau_2) \qquad (59)$$

2.5.2.4 Higher Order Kernels

We proceed similarly to the determination of $h_3(\tau_1,\tau_2,\tau_3)$. In this case, we recognize that there are

contributions to the third order Volterra series term from the multinomial product terms $[y_1(\tau)]^3$ and $[y_1(\tau)y_2(\tau)]$ in equation (3). Let us label these $h_{3,a}(\tau_1,\tau_2,\tau_3)$ and $h_{3,b}(\tau_1,\tau_2,\tau_3)$ respectively; we shall determine them separately, for convenience, and then combine the results. The complete third order Volterra kernel is:

$$h_3(\tau_1,\tau_2,\tau_3) = h_{3,a}(\tau_1,\tau_2,\tau_3) + 2h_{3,b}(\tau_1,\tau_2,\tau_3) \tag{60}$$

where the $h_{3,b}$ contribution is taken twice since it results from two terms in the multinomial expansion.

Using the same sort of substitutions of previously derived Volterra series terms which led to equation (51) and following a rearrangement of integrations, we extract the components of the third order kernel as was done for the second order kernel in equation (53). We obtain:

$$h_{3,a}(\tau_1,\tau_2,\tau_3) = -a_3 \int_{-\infty}^{\infty} h(\tau)h_1(\tau_1-\tau)h_1(\tau_2-\tau)h_1(\tau_3-\tau)d\tau \tag{61}$$

$$h_{3,b}(\tau_1,\tau_2,\tau_3) = -a_2 \int_{-\infty}^{\infty} h(\tau)h_1(\tau_1-\tau)h_2(\tau_2-\tau,\tau_3-\tau)d\tau \tag{62}$$

Using equations (50) and (53), we can further expand equations (61) and (62). This yields:

$$h_{3,a}(\tau_1,\tau_2,\tau_3) = -\frac{a_3}{R^3}\int_{-\infty}^{\infty} h(\tau)h(\tau_1-\tau)h(\tau_2-\tau)h(\tau_3-\tau)d\tau \qquad (63)$$

$$h_{3,b}(\tau_1,\tau_2,\tau_3) = -a_2\int_{-\infty}^{\infty} h(\tau)\frac{1}{R}h(\tau_1-\tau)\left\{\frac{-a_2}{R^2}\int_{-\infty}^{\infty} h(\alpha)h(\tau_2-\tau-\alpha)h(\tau_3-\tau-\alpha)d\alpha\right\}d\tau \qquad (64)$$

We observe that, in equation (64), there are now <u>two</u> integrals to be evaluated in order to obtain the desired contribution to the third order Volterra kernel. This typifies the general character of the higher-order Volterra kernels.

The characteristic which will be observed as we derive higher order kernels is that each term in the expression for the n^{th} order kernel, $h_n(\tau_1,\ldots,\tau_n)$, will be generated by the convolution of the linear impulse response $h(t)$ with a product of lower order Volterra series kernels. This convolution requires one integration (which we have shown with respect to the variable τ). Each kernel, h_m, in the integrand for which $m>1$ can be replaced by its counterpart expression to equation (64). Each such substitution will result in the introduction of an additional convolution integral. Since m is necessarily less than n, the expression can be reduced ultimately to one containing only the function $h(\tau)$. There will be p replicas of $h(\tau)$ in each term of $h_n(\tau_1,\ldots,\tau_n)$ where $p=k+n$ and k is the number of

integrations to be performed in order to obtain the kernel, h_n. In general, the different terms of h_n will be generated with a different number of integrations.

We may substitute the expressions previously found for the h_k (e.g. equations (55), (58)) into equations (61) and (62) in order to explicitly determine the higher order terms. While this is cumbersome for increasing n, it is a straightforward procedure. For the third order kernels, we obtain:

$$h_{3,a}(\tau_1,\tau_2,\tau_3) = \frac{-a_3}{R^3C^4}\int_{-\infty}^{\infty} \exp[-k\tau]\exp[-k(\tau_1-\tau)]\exp[-k(\tau_2-\tau)]\exp[-k(\tau_3-\tau)]$$

$$u(\tau)u(\tau_1-\tau)u(\tau_2-\tau)u(\tau_3-\tau)d\tau$$

$$= \frac{-a_3}{R^3C^4}\int_{0}^{\min\{\tau_1,\tau_2,\tau_3\}} \exp[+k(2\tau-\tau_1-\tau_2-\tau_3)]d\tau \qquad (65)$$

Evaluating the integral yields:

$$h_{3,a}(\tau_1,\tau_2,\tau_3) = \frac{a_3}{2R^3C^4k}\{\exp[-k\tau_1]\exp[-k\tau_2]\exp[-k\tau_3]$$

$$-\exp[+k\tau_i]\exp[-k\tau_j)]\exp[-k\tau_k]\}u(\tau_1)u(\tau_2)u(\tau_3) \qquad (66)$$

where we have defined: $\tau_i=\min\{\tau_1, \tau_2, \tau_3\}$ and $\{\tau_j,\tau_k\} = \{\tau_1,\tau_2,\tau_3\}\backslash\tau_i$. [The set notation $\{a,b,c\}\backslash x$ is taken to mean: the set of elements a, b, and c excluding the element x. The

implication here is that we know the membership of the complete set, i.e. $a, b,$ and c but the correspondance between the deleted element x and the set members is not explicit.]

Similarly, we derive the second part of the third order kernel, $h_{3,b}(\tau_1,\tau_2,\tau_3)$. Substituting equations (55) and (58) into equation (62) gives one of six possible expressions for this term:

$$h_{3,b}(\tau_1,\tau_2,\tau_3) =$$

$$-a_2 \int_{-\infty}^{\infty} \frac{1}{C}\exp[-k\tau]\frac{1}{RC}\exp[-k(\tau_1-\tau)]\frac{a_2}{C^3R^2k}\{\exp[-k(\tau_2-\tau)]\exp[-k(\tau_3-\tau)]$$

$$-\exp[-k(\max\{\tau_2,\tau_3\}-\tau)]\}u(\tau)u(\tau_1-\tau)u(\tau_2-\tau)u(\tau_3-\tau)d\tau \qquad (67)$$

In equations (62) and (67) we have arbitrarily assigned the variables t_1 to h_1 and t_2, t_3 to h_2. We could have chosen to assign either t_2 or t_3 to h_1 with the complementary assignments to h_2. It was by our choice in the assignment of variables that we have determined the shape of the *asymmetric* $h_{3,b}$ term which we will obtain. No choice is better than another; we have previously shown (Section 2.5.1.1) that responses calculated from all versions of $h_{3,b}$ obtained by permutation of its arguments are identical.

We have ignored, here, the ordering of the arguments of h_2 because we know already by equation (58) that the second-order kernel is symmetric. Similarly, the $h_{3,a}$ kernel obtained in equation (66) is inherently symmetric.

Using the variable assignment of equation (67) we can find the $h_{3,b}$ kernel as:

$$h_{3,b}(\tau_1,\tau_2,\tau_3) = \tag{68}$$

$$-\frac{a_2^2}{R^3C^5k}\int_0^{\min\{\tau_1,\tau_2,\tau_3\}} \exp[+k(2\tau-\tau_1-\tau_2-\tau_3)]-\exp[-k(\tau_1+\max\{\tau_2,\tau_3\}-2\tau)]d\tau$$

Integrating and simplifying yields:

$$h_{3,b}(\tau_1,\tau_2,\tau_3) = \frac{a_2^2}{2R^3C^5k^2}\{\exp[-k(\tau_1+\tau_2+\tau_3)]-\exp[-k(\tau_j+\tau_k-\tau_i)] \tag{69}$$

$$-\exp[-k(\tau_1+\max\{\tau_2,\tau_3\})]+\exp[-k(\tau_1+\max\{\tau_2,\tau_3\}-2\tau_i)]\}\,u(\tau_1)u(\tau_2)u(\tau_3)$$

Since $h_{3,b}$ as presented in equation (69) is asymmetric, it may be symmetrized as described in Section 2.5.1.2. The resulting kernel, $S[h_{3,b}]$, is:

$$S[h_{3,b}(\tau_1,\tau_2,\tau_3)] = \frac{1}{3}\frac{a_2^2}{2R^3C^5k^2}\{3\exp[-k(\tau_1+\tau_2+\tau_3)] - 3\exp[-k(\tau_j+\tau_k-\tau_i)]$$

$$-\exp[-k(\tau_1+\max\{\tau_2,\tau_3\})] - \exp[-k(\tau_2+\max\{\tau_1,\tau_3\})] - \exp[-k(\tau_3+\max\{\tau_1,\tau_2\})]$$

$$+\exp[-k(\tau_1+\max\{\tau_2,\tau_3\}-2\tau_i)] + \exp[-k(\tau_2+\max\{\tau_1,\tau_3\}-2\tau_i)]$$

$$+\exp[-k(\tau_3+\max\{\tau_1,\tau_2\}-2\tau_i)]\}u(\tau_1)u(\tau_1)u(\tau_3) \qquad (70)$$

The presence of a growing exponential term should not be disturbing, since it is always the smallest value, $\exp\{k\tau_i\}$, and is always dominated by the other terms.

In determining the fourth order term, h_4, there are contributions from $[y_1]^4$, $[y_1y_3]$, $[y_2]^2$, and $[y_1^2y_2]$. We shall call these $h_{4,a}$, $h_{4,b}$, $h_{4,c}$, and $h_{4,d}$. The complete fourth order Volterra kernel is then (accounting for the number of terms in the multinomial expansion):

$$h_4 = h_{4,a} + 2h_{4,b} + h_{4,c} + 3h_{4,d}$$

The individual terms which comprise the fourth-order kernel may be written as:

$$h_{4,a}(\tau_1,\tau_2,\tau_3,\tau_4) = -a_4\int_{-\infty}^{\infty} h(\tau)h_1(\tau_1-\tau)h_1(\tau_2-\tau)h_1(\tau_3-\tau)h_1(\tau_4-\tau)d\tau$$

$$= \frac{-a_4}{R^4}\int_{-\infty}^{\infty} h(\tau)h(\tau_1-\tau)h(\tau_2-\tau)h(\tau_3-\tau)h(\tau_4-\tau)d\tau \qquad (71)$$

$$h_{4,b}(\tau_1,\tau_2,\tau_3,\tau_4) = -a_2 \int_{-\infty}^{\infty} h(\tau) h_1(\tau_1-\tau) h_3(\tau_2-\tau,\tau_3-\tau,\tau_4-\tau)\,d\tau$$

$$= \frac{a_2 a_3}{R^4} \int_{-\infty}^{\infty} h(\tau)\, h(\tau_1-\tau) \int_{-\infty}^{\infty} h(\alpha-\tau) h(\tau_2-\alpha) h(\tau_3-\alpha) h(\tau_4-\alpha)\,d\alpha\,d\tau \qquad (72)$$

$$+\frac{a_2^2}{R^4} \int_{-\infty}^{\infty} h(\tau) \frac{1}{R} h(\tau_1-\tau) \int_{-\infty}^{\infty} h(\alpha-\tau) h(\tau_2-\alpha) \int_{-\infty}^{\infty} h\left(\beta-\alpha\right) h\left(\tau_3-\beta\right) h\left(\tau_4-\beta\right) d\beta\,d\alpha\,d\tau$$

$$h_{4,c}(\tau_1,\tau_2,\tau_3,\tau_4) = -a_2 \int_{-\infty}^{\infty} h(\tau) h_2(\tau_1-\tau,\tau_2-\tau) h_2(\tau_3-\tau,\tau_4-\tau)\,d\tau \qquad (73)$$

$$= -\frac{a_2^3}{R^4} \int_{-\infty}^{\infty} h(\tau) \int_{-\infty}^{\infty} h(\alpha-\tau) h(\tau_1-\alpha) h(\tau_2-\alpha)\,d\alpha \int_{-\infty}^{\infty} h\left(\beta-\tau\right) h\left(\tau_3-\beta\right) h\left(\tau_4-\beta\right) d\beta\,d\tau$$

$$h_{4,d}(\tau_1,\tau_2,\tau_3,\tau_4) = -a_3 \int_{-\infty}^{\infty} h(\tau) h_1(\tau_1-\tau) h_1(\tau_2-\tau) h_2(\tau_3-\tau,\tau_4-\tau)\,d\tau \qquad (74)$$

$$= \frac{a_3 a_2}{R^4} \int_{-\infty}^{\infty} h(\tau) h(\tau_1-\tau) h(\tau_2-\tau) \int_{-\infty}^{\infty} h(\alpha-\tau) h(\tau_3-\alpha) h(\tau_4-\alpha)\,d\alpha\,d\tau$$

Recognizing the complexity of the first three kernels for our example, one can see that the fourth order kernel will be exceedingly cumbersome to write out explicitly. Nevertheless, if desired it may be determined in the same straightforward manner. We will not present it here, however.

2.5.3 *Determination of Volterra Kernel Transforms by the Harmonic Input Method*

A method, called the harmonic input method, for obtaining the multivariate frequency-domain kernel transforms, $H_n(f_1,\ldots,f_n)$, (see equation (4)) was given by Bedrosian and Rice [18]. By introducing inputs of the form:

$$x(t)=\exp\left(j2\pi f_1 t\right)+\exp\left(j2\pi f_2 t\right)+\cdots+\exp\left(j2\pi f_n t\right) \tag{75}$$

into the appropriate nonlinear differential equation (see equation (2a)) the coefficient of the response at a frequency $f=f_1+\ldots+f_n$ is found to be $n!$ times the value of $H_n(f_1,\ldots,f_n)$, the n^{th}-order nonlinear transfer function. If the arguments, f_i are introduced as parameters, the kernel transform, H_n, is determined as a function.

2.5.3.1 *Continuation of the Nonlinear Circuit Example*

Returning to the example presented in section 2.5.2.1, we may obtain a differential equation for the response voltage. With reference to Figure 2, we write:

$$\frac{1}{R}v_s(t)=g_s y(t)+C\frac{d}{dt}y(t)+g_l y(t)+f\left[y(t)\right] \tag{76}$$

Multiplying both sides by R, recognizing that $g_s=1/R$, and substituting the series expansion in equation (36) for $f[y(t)]$ yields:

$$v_s(t)=[1+Rg_1]y(t)+RC\frac{d}{dt}y(t)+R\sum_{n=2}^{\infty}\frac{I_s\lambda^n}{n!}y(t)^n \tag{77}$$

If we arbitrarily truncate the power series expansion at some finite degree, m, then we have a representation in the form of equation (2a). (The value chosen for m may be made sufficiently large that no error is introduced into the first N terms of a finite Volterra series.)

2.5.3.2 Harmonic Probing

In presenting the harmonic probing method for obtaining the nonlinear kernel transforms, we follow the development given by Bussgang, et.al. [11]. We will utilize several of the relationships derived in section 2.4.

Assume that the response to the system described by equation (77) can be represented by a Volterra series:

$$y(t)=\sum_{n=1}^{\infty}y_n(t) \tag{78}$$

where the $y_n(t)$ are expressed in the following form:

$$y_n(t) = \int_{-\infty}^{\infty} \cdots \int H_n\left(f_1, \cdots, f_n\right) X\left(f_1\right) \cdots X\left(f_n\right) \exp\left(j2\pi t \sum_{i=1}^{n} f_i\right) df_1 \cdots df_n \qquad (79)$$

Setting $v_s(t) = x(t)$ and substituting equation (78) into equation (77), we obtain:

$$x(t) = \left[1 + Rg_1\right] \sum_{n=1}^{\infty} y_n(t) + RC\frac{d}{dt} \sum_{n=1}^{\infty} y_n(t) + R \sum_{n=2}^{\infty} \frac{I_s \lambda^n}{n!} \left[\sum_{n=1}^{\infty} y_n(t)\right]^n \qquad (80)$$

Choosing $x(t) = \exp(j2\pi ft)$, we recognize that the only terms on the right hand side of equation (80) which can contain complex exponentials at the input frequency are the $y_1(t)$ components of the first two terms; all other $y_n(t)$ terms necessarily contain complex exponentials at multiples of the input frequency. Likewise, the nonlinear term necessarily contains only products of the input signal and higher order complex exponentials. Thus, expressing the equality of the terms in $\exp(j2\pi ft)$, we may write:

$$\exp\left(j2\pi ft\right) = \left[1 + Rg_1\right] H_1\left(f\right) \exp\left(j2\pi ft\right) + RC\frac{d}{dt}\left[H_1\left(f\right) \exp\left(j2\pi ft\right)\right] + 0 \qquad (81)$$

Carrying out the differentiation and cancelling the $\exp(j2\pi ft)$ factors leaves:

$$1 = \left[1 + Rg_1\right] H_1\left(f\right) + RCj2\pi f H_1\left(f\right) \qquad (82)$$

After a simple algebraic manipulation, the first order (i.e. linear) transfer function is obtained:

$$H_1\left(f\right)=\frac{1}{1+Rg_l+j2\pi fRC} \tag{83}$$

This may also be written:

$$H_1(s)=\frac{1}{RC}\frac{1}{s+\frac{1+Rg_l}{RC}} \tag{84}$$

Recognizing the term $(1+Rg_l)/RC$ as k in equation (42), we may write:

$$H_1(s)=\frac{1}{RC}\frac{1}{s+k} \tag{85}$$

By the inverse Laplace transform, we have:

$$h_1(t)=\frac{1}{RC}\exp(-kt)u(t) \tag{86}$$

which is identical to equation (55).

With this knowledge of the first order transfer function, we set the input of the system (i.e., the forcing function of the differential equation) to:

$$x(t)=\exp\left(j2\pi f_1 t\right)+\exp\left(j2\pi f_2 t\right) \tag{87}$$

We recognize that a second order response component due to the two exponentials in the input will occur at a frequency $f=f_1+f_2$. Therefore, inserting this input into equation (80) and equating the coefficients of the terms which have a component at the sum frequency we obtain:

$$0=[1+Rg_l]H_2(f_1,f_2)+RCH_2(f_1,f_2)j2\pi(f_1+f_2)+Ra_2H_1(f_1)H_1(f_2) \qquad (88)$$

where we have defined: $a_2=\frac{1}{2}I_s\lambda^2$. This is the coefficient of the only term in the nonlinear function approximation polynomial which can produce a sum frequency of f_1+f_2. Solving algebraically for $H_2(f_1,f_2)$, we obtain:

$$H_2(f_1,f_2)=-a_2H_1(f_1)H_1(f_2)\frac{R}{1+Rg_l+j2\pi(f_1+f_2)RC} \qquad (89)$$

Recognizing the final factor in equation (89) to be $H(f_1+f_2)$, we may write:

$$H_2(f_1,f_2)=-a_2H_1(f_1)H_1(f_2)H(f_1+f_2) \qquad (90)$$

This may be shown to be identically the Fourier transform of equation (58) as we should expect.

$$F^{-1}[H_2(f_1,f_2)]=-a_2\{F^{-1}[H_1(f_1)]F^{-1}[H_1(f_2)]\}*F^{-1}[H(f_1+f_2)] \qquad (91)$$

Performing the individual inversions, we obtain:

$$h_2(\tau_1,\tau_2) = -a_2\{h_1(\tau_1)h_1(\tau_2)\} * h(\tau)\big|_{\tau=\tau_1+\tau_2} \qquad (92)$$

The relationship may be compared to equation (55).

2.6 Systems Containing Multiple Nonlinearities

The example previously shown illustrates how a Volterra integral equation may be obtained for the simplest of nonlinear circuits: a first order circuit containing a single nonlinearity. Comparable, although more complicated results may be obtained for circuits (or systems) of higher order and containing multiple nonlinear elements. The state variable formulation for a system provides a convenient basis for illustration.

Let a nonlinear system be described by the following state variable representation (with the independent variable t suppressed):

$$\dot{x} = Ax + \Gamma f(x) + Bu \qquad (93)$$

where: x is the n-dimensional state vector of the system

A and Γ are n x n constant coefficient matrices

B is an $n \times l$ constant coefficient matrix

u is an l dimensional input vector

$f(x)$ is an n-dimensional vector of strictly nonlinear functions of the state variables, i.e.

$$f(x) = [f_1(x_1), f_2(x_2), \cdots, f_n(x_n)]^T$$

In defining $f(x)$, we have assumed that no nonlinear element is controlled by more than one state variable or by the derivative of a state variable. The requirement that $f(x)$ be comprised of strictly nonlinear functions $f_i(x_i)$ is readily accomplished by decomposition of the functions. Suppose that $f_i(z)$ is given by:

$$f_i(z) = \sum_{j=0}^{\infty} a_j z^j \tag{94}$$

If a_0 is nonzero, then it may be set to zero and replaced by a source u_{l+1} with value a_0, and the $(l+1)^{st}$ column in B becomes the i^{th} column of Γ.

If a_1 is nonzero, then it may be set to zero and the i^{th} column of A increased by a_1 times the i^{th} column of Γ.

Taking the Laplace transform of both sides of equation (93) yields:

$$sX(s) = AX(s) + \Gamma F[X(s)] + BU(s) \tag{95}$$

where $F[X(s)]$ is the Laplace transform of $f[x(t)]$. The transform of the n^{th} degree term in each $f_i(x_i)$ exists as the n-fold convolution of $X_i(s)$ with itself. It is clearly cumbersome and uninformative to express it in this form; however, it will not be needed explicitly, so the notation $F[X(s)]$ will suffice.

A rearrangement of equation (95) gives:

$$(sI-A)X(s)=\Gamma F[X(s)]+BU(s) \tag{96}$$

where I is the $n\,x\,n$ identity matrix. Defining $H(s)=(sI-A)^{-1}$, we may write:

$$X(s)=H(s)BU(s)+H(s)\Gamma F[X(s)] \tag{97}$$

Now inverting the Laplace transform of both sides of equation (97), we obtain:

$$x(t)=h(t)*Bu(t)+h(t)*\Gamma f[x(t)] \tag{98}$$

where $h(t)=L^{-1}[H(s)]$. Keep in mind that both H(s) and h(t) are nxn matrices.

Equation (98) is the multidimensional equivalent of equation (2b) with a change of sign for the second term on the right hand side.

It is customary in state variable formulations to describe an output vector $y(t)$ of the form:

$$y = Cx + Du \tag{99}$$

This may, of course, be performed in the present situation as well; however, the essential solution of the nonlinear system is embodied in the solution of equation (98).

The solution of equation (98) for a large system may entail a large number of convolution integrals, however, the solution may be accomplished, albeit tediously, in precisely the manner we have described in our example.

CHAPTER 3

Discrete-Time Volterra Series

In order to effectively utilize Volterra series representations of systems in simulations or other discrete- time applications, it is necessary to obtain discrete approximations to the continuous-time Volterra series kernels described in Chapter 2. As with linear systems, discretization entails approximation. In the following sections, we discuss the nature of the approximations and means for bounding the errors introduced by them.

3.1 Approximation

Knowledge of the Volterra series for a particular system is, by itself, insufficient to accurately determine the response of the system to arbitrary inputs. In practice, one must obtain an approximation to the complete Volterra series response, as the generally infinite Volterra series for a system is not computable.

To make an approximation a useful tool, we must establish a bound on the error. Furthermore, it is preferable that the bound be such that it may be reduced, at the expense of additional computational effort, to any desired value. Below, we consider the ways in which

approximation errors must be introduced into a Volterra series representation of a nonlinear system response in order to obtain a computable discrete time model.

Although an infinite Volterra series is not computable, we can, nevertheless, often derive a practically useful tool for computing a system response. This requires a satisfactory approximation of the complete Volterra series response, $y(t)$, to be realized by the first N terms of equation (1) for a very small value of N.

Furthermore, if the nonlinear system input is stochastic, for example a communication signal, then explicit calculation of a response via the Volterra integrals may not be possible. In such cases, the most practical means of system evaluation may be Monte Carlo computer simulation [19]. This necessarily requires realization of discrete time approximations to the N kernels of the truncated Volterra series - a second form of approximation.

In discrete-time signal simulation, operations are performed on samples of the input process; typically, these samples represent values of the process at uniform periodic instants of time. Where the process is strictly bandlimited, the Nyquist criterion determines a sampling rate at which full signal fidelity is preserved in *linear* discrete-time processing. However, few processes may be

regarded a strictly bandlimited; therefore, selection of a sampling rate necessarily reflects the compromise between accuracy and computational burden.

Hence, sampling of the input signal introduces a third form of approximation. The error due to signal sampling has been given the specific name "aliasing" due to the nature of its manifestation for distinct frequency tones.

Nonlinear system representation in discrete time is significantly complicated by the bandwidth-expanding character of nonlinear systems. Recognizing that the n-fold product of the input in the n^{th} term of the Volterra series (1) corresponds to an n-fold convolution of the input signal spectrum, it is immediately apparent that an N-term truncation of the Volterra series can produce a significant response over a bandwidth N times as great as the essential bandwidth of the input[1]. Therefore, while the compromise between accuracy and computational burden is not a consequence of nonlinearity, the optimization of sampling rate is far more difficult than for linear discrete time processing.

While coefficient quantization, roundoff, and saturation errors due to finite wordlength in digital signal processing systems are often treated as a fourth

---

[1] This applies to lowpass signals; for bandpass signals, the spectral convolutions implied by the higher order signal products may not overlap, resulting in greater bandwidth expansion.

form of approximation error, we do not consider them here. Simulation ordinarily utilizes floating point arithmetic wherein we assume that finite wordlength effects are controlled to a degree which reduces their impact to a level far below that of the three sources of approximation error which we examine here.

3.2 Discretization of the Linear Convolution Integral

As a prelude to quantifying and bounding the various sources of error which accompany the discrete-time simulation of continuous-time systems, we present a derivation of the discrete-time convolution summation as a counterpart to the continuous-time convolution integral. We examine the discretization of the linear convolution integral because this sets the stage for all that we shall do with the multidimensional, higher-order Volterra series terms. It permits us to identify the approximations previously described in the context of a familiar mathematical framework. It further provides a basis for discrete time simulation of analog systems and establishes the conditions which are implicit in such simulations for linear systems. This is a necessary point of departure for a study of simulations of nonlinear systems.

With regard to the difference between simulation and digital signal processing, we have previously noted that

simulation is normally not concerned with the finite wordlength effects which can be an important issue in digital signal processing. Digital signal processing, as used here, implies quantization of signals. Furthermore, in digital signal processing, filter structures are often optimized with respect to filter specifications. Simulation, on the other hand, is concerned with optimization with regard to a particular set of analog characteristics which may be less easily described than a filter specification.

Let $x(t)$ represent the input to a linear system, **H**, which has the impulse response $h(t)$. Let $y(t)$ be the response of **H** to $x(t)$. Then:

$$y(t) = \int_{-\infty}^{\infty} h(t-\tau)\,x(\tau)\,d\tau \qquad (1)$$

Assume that we can choose a frequency, W Hertz, such that both $x(t)$ and $h(t)$ are bandlimited by W, i.e.

$$X(f) = 0, \qquad |f| > W$$

$$H(f) = 0, \qquad |f| > W$$

where $X(f)$ and $H(f)$ are the Fourier transforms of $x(t)$ and $h(t)$ respectively.

Then we may represent $x(t)$ and $h(t)$ with sampling expansions:

$$x(t) = \sum_{n=-\infty}^{\infty} x\left(\frac{n}{2W}\right) \operatorname{sinc}\left[2W\left(t-\frac{n}{2W}\right)\right] \tag{2}$$

$$h(t) = \sum_{k=-\infty}^{\infty} h\left(\frac{k}{2W}\right) \operatorname{sinc}\left[2W\left(t-\frac{k}{2W}\right)\right] \tag{3}$$

where $\operatorname{sinc}(z) = \frac{\sin(\pi z)}{\pi z}$. The infinite limits (or, as a minimum, semi-infinite if we assume causality) of summation are an inescapable consequence of our assumption that $x(t)$ and $h(t)$ are bandlimited.

If we substitute equations (2) and (3) into equation (1) we obtain:

$$y(t) = \int_{-\infty}^{\infty} \sum_{k=-\infty}^{\infty} h\left(\frac{k}{2W}\right) \operatorname{sinc}\left[2W\left(t-\tau-\frac{k}{2W}\right)\right] \sum_{n=-\infty}^{\infty} x\left(\frac{n}{2W}\right) \operatorname{sinc}\left[2W\left(\tau-\frac{n}{2W}\right)\right] d\tau \tag{4}$$

Rearrangement gives:

$$y(t) = \sum_{k=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} h\left(\frac{k}{2W}\right) x\left(\frac{n}{2W}\right) \int_{-\infty}^{\infty} \operatorname{sinc}\left[2W\left(t-\tau-\frac{k}{2W}\right)\right] \operatorname{sinc}\left[2W\left(\tau-\frac{n}{2W}\right)\right] d\tau \tag{5}$$

Recognizing that the integral represents the convolution of two ideal lowpass filters (weighted by $1/2W$)

with delays $k/2W$ and $n/2W$ it can be seen that it evaluates to: $\left(\frac{1}{2W}\right)\text{sinc}\left[2W\left(t-\frac{k-n}{2W}\right)\right]$. Therefore,

$$y(t)=\frac{1}{2W}\sum_{n=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}h\left(\frac{k}{2W}\right)x\left(\frac{n}{2W}\right)\text{sinc}\left[2W\left(t-\frac{k}{2W}-\frac{n}{2W}\right)\right] \tag{6}$$

But, this is the same as:

$$y(t)=\frac{1}{2W}\sum_{n=-\infty}^{\infty}\sum_{k=-\infty}^{\infty}h\left(\frac{k}{2W}\right)x\left(\frac{n-k}{2W}\right)\text{sinc}\left[2W\left(t-\frac{n}{2W}\right)\right] \tag{7}$$

Since $x(t)$ is W-bandlimited and **H** is linear, $y(t)$ must also be W-bandlimited. Therefore:

$$y(t)=\sum_{n=-\infty}^{\infty}y\left(\frac{n}{2W}\right)\text{sinc}\left[2W\left(t-\frac{n}{2W}\right)\right] \tag{8}$$

Then by comparing equations (7) and (8), we recognize that:

$$y\left(\frac{n}{2W}\right)=\frac{1}{2W}\sum_{k=-\infty}^{\infty}h\left(\frac{k}{2W}\right)x\left(\frac{n-k}{2W}\right) \tag{9}$$

This apparently error-free result suggests that, perhaps, we were too hasty in the earlier assertion that approximation necessarily accompanies discretization. Equations (8) and (9) justify the notion that discrete time signal processing can simulate the processing of a linear, continuous time system under the constraints that both the signal and system are bandlimited. Unfortunately, the

summation in equation (9) is infinite, hence not computable. Furthermore, the bandlimitation constraint is not strictly achievable for either signals or systems in the physical world. Therefore, we must accept some approximation in order to obtain a realizable simulation.

3.3 Discretization of the Second Order Response

The second order response component of a Volterra system has potentially twice the bandwidth, W, of the input signal. Therefore, the discrete time counterpart to the second order term of equation (1) must, in general, produce samples of the $y_2(t)$ term which permit reconstruction of a signal having bandwidth $2W$. Otherwise, the response will be incorrect, corrupted by aliasing. Accordingly, we seek a representation for $y_2(t)$ of the form:

$$y_2(t) = \sum_{j=-\infty}^{\infty} y_2\left(\frac{j}{4W}\right) \operatorname{sinc}\left[4W\left(t - \frac{j}{4W}\right)\right] \tag{10}$$

where $y_2(j/4W)$ can be determined by an operation on samples of $x(t)$ and the second order Volterra kernel $h_2(\tau_1, \tau_2)$.

Let us assume that the second-order Volterra kernel, $h_2(\tau_1,\tau_2)$ is known. Then the second order response component is given by:

$$y_2(t) = \iint_{-\infty}^{\infty} h_2(\tau_1,\tau_2)x(t-\tau_1)x(t-\tau_2)d\tau_1 d\tau_2 \qquad (11)$$

Assume, as before, that the input $x(t)$ is W-bandlimited. Equation (2) applies; however, it will be convenient to express $x(t)$ with respect to the sampling interval $T_s=1/4W$ used in equation (10):

$$x(t) = \sum_{k=-\infty}^{\infty} x\left(\frac{k}{4W}\right)\mathrm{sinc}\left[4W\left(t-\frac{k}{4W}\right)\right] \qquad (12)$$

Further assume that the second-order nonlinear transfer function is $2W$-bandlimited in each dimension:

$$H_2\left(f_1,f_2\right)=0 \quad \text{if} \quad |f_1|,|f_2|>2W$$

where $H_2(f_1,f_2)$ is defined by equation (2-4). Then:

$$h_2(\tau_1,\tau_2) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} h_2\left(\frac{m}{4W},\frac{n}{4W}\right)\mathrm{sinc}\left[4W\left(\tau_1-\frac{m}{4W}\right)\right]\mathrm{sinc}\left[4W\left(\tau_2-\frac{n}{4W}\right)\right] \qquad (13)$$

Our bandwidth constraint on $h_2(\tau_1,\tau_2)$ permits sampling at a rate consistent with that necessary to correctly represent a doubled bandwidth response. Substituting for $x(t)$ and $h_2(\tau_1,\tau_2)$ in equation (11) using equations (12) and

(13) but maintaining separate subscripts, for the moment, on the τ variables, yields:

$$y_2(t_1,t_2)=\int_{-\infty}^{\infty}\int\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}h_2\left(\tfrac{m}{4W},\tfrac{n}{4W}\right)\operatorname{sinc}\left[4W\left(\tau_1-\tfrac{m}{4W}\right)\right]\operatorname{sinc}\left[4W\left(\tau_2-\tfrac{n}{4W}\right)\right] \qquad (14)$$

$$\times\sum_{k=-\infty}^{\infty}x\left(\tfrac{k}{4W}\right)\operatorname{sinc}\left[4W\left(t_1-\tau_1-\tfrac{k}{4W}\right)\right]\times\sum_{l=-\infty}^{\infty}x\left(\tfrac{l}{4W}\right)\operatorname{sinc}\left[4W\left(t_2-\tau_2-\tfrac{l}{4W}\right)\right]d\tau_1 d\tau_2$$

Note that $y_2(t) = y_2(t_1,t_2)$ evaluated for $t_1=t_2=t$. A rearrangement of equation (14) yields:

$$y_2(t_1,t_2)=\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}h_2\left(\tfrac{m}{4W},\tfrac{n}{4W}\right)x\left(\tfrac{k}{4W}\right)x\left(\tfrac{l}{4W}\right)$$

$$\times\int_{-\infty}^{\infty}\operatorname{sinc}\left[4W\left(\tau_1-\tfrac{m}{4W}\right)\right]\operatorname{sinc}\left[4W\left(t_1-\tau_1-\tfrac{k}{4W}\right)\right]d\tau_1 \qquad (15)$$

$$\times\operatorname{sinc}\left[4W\left(\tau_2-\tfrac{n}{4W}\right)\right]\operatorname{sinc}\left[4W\left(t_2-\tau_2-\tfrac{l}{4W}\right)\right]d\tau_2$$

Each of the integrals is a convolution of two ideal lowpass functions with equal bandwidth but different magnitudes and delays. Each integral evaluates to an ideal lowpass function having the same bandwidth, the product of the magnitudes, and the sum of the delays. Equation (15) then becomes:

$$y_2(t_1,t_2)=\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}h_2\left(\frac{m}{4W},\frac{n}{4W}\right)x\left(\frac{k}{4W}\right)x\left(\frac{l}{4W}\right) \tag{16}$$

$$\left\{\frac{1}{4W}\,\mathrm{sinc}\left[4W\left(t_1-\frac{m}{4W}-\frac{k}{4W}\right)\right]\right\}\left\{\frac{1}{4W}\,\mathrm{sinc}\left[4W\left(t_2-\frac{n}{4W}-\frac{l}{4W}\right)\right]\right\}$$

The $m/4W$ and $n/4W$ delays in the sinc[ ] functions may be transferred to the $x(\ )$ terms to yield a two-dimensional discrete time convolution of the expected form:

$$y_2(t_1,t_2)=\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}h_2\left(\frac{m}{4W},\frac{n}{4W}\right)x\left(\frac{k-m}{4W}\right)x\left(\frac{l-n}{4W}\right)$$

$$\times\left(\frac{1}{4W}\right)^2\mathrm{sinc}\left[4W\left(t_1-\frac{k}{4W}\right)\right]\mathrm{sinc}\left[4W\left(t_2-\frac{l}{4W}\right)\right] \tag{17}$$

Setting $t_1=t_2=t$, we may rewrite equation (17) as:

$$y_2(t)=\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}h_2\left(\frac{m}{4W},\frac{n}{4W}\right)x\left(\frac{k-m}{4W}\right)x\left(\frac{l-n}{4W}\right)$$

$$\times\left(\frac{1}{4W}\right)^2\mathrm{sinc}\left[4W\left(t-\frac{k}{4W}\right)\right]\mathrm{sinc}\left[4W\left(t-\frac{l}{4W}\right)\right] \tag{18}$$

In order to determine the required values, $y_2(j/4W)$, to satisfy equation (11), we evaluate the right hand side of equation (18) at $t=j/4W$.

$$y_2\left(\frac{j}{4W}\right)=\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}h_2\left(\frac{m}{4W},\frac{n}{4W}\right)x\left(\frac{k-m}{4W}\right)x\left(\frac{l-n}{4W}\right)\left(\frac{1}{4W}\right)^2\mathrm{sinc}[j-k]\,\mathrm{sinc}[j-l] \tag{19}$$

Since $\text{sinc}[j-k]=0$ unless $j=k$, equation (19) can be simplified to:

$$y_2\left(\frac{j}{4W}\right)=\left(\frac{1}{4W}\right)^2\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}h_2\left(\frac{m}{4W},\frac{n}{4W}\right)x\left(\frac{j-m}{4W}\right)x\left(\frac{j-n}{4W}\right) \qquad (20)$$

Substituting equation (20) in equation (11) gives the deisred result:

$$y_2(t)=\left(\frac{1}{4W}\right)^2\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}h_2\left(\frac{m}{4W},\frac{n}{4W}\right)x\left(\frac{j-m}{4W}\right)x\left(\frac{j-n}{4W}\right)\text{sinc}\left[4W\left(t-\frac{j}{4W}\right)\right] \qquad (21)$$

Equations (20) and (21) provide the extension to two dimensions of the linear discrete-time convolution foundation which was presented in section 3.2. Comparable expressions for the higher order Volterra series responses may be similarly obtained.

3.4 Aliasing and Signal Truncation Error

Aliasing is the name given to the error introduced by insufficient sampling. This error is introduced when sampling is not performed at a rate greater than the Nyquist rate and becomes unavoidable when a signal or a transfer function is not strictly bandlimited. When this occurs, equations (2) or (3) fail to yield an exact reconstruction of the signal or impulse response and the

discrete-time convolution derived in section 3.2 is not a perfect simulation of the continuous-time system response.

Signal truncation is described here as a companion to aliasing. It is, in fact, a dual problem. While aliasing results when a signal spectrum is nonzero beyond the Nyquist frequency associated with the sampling rate (i.e., the signal spectrum is *not* truncated), truncation error occurs when arbitrary elimination of samples of a discrete-time domain representation is performed to preserve a manageable finite representation.

Aliasing and truncation errors must be understood and controlled when discrete time techniques are applied to any problem. Bandlimited signals and systems exist only as idealizations of physical waveforms and hardware. Moreover, from Fourier transform theory, it is clear that no signal or response can be both bandlimited and duration limited. Consequently, discrete-time processing of a finite duration signal record as a simulation of an analog system's behavior is inherently in error. Thus understanding, quantifying, and bounding aliasing and signal truncation errors is essential to meaningful discrete-time signal processing and simulation.

While aliasing of a sampled signal representation and of a discrete time system impulse response are fundamentally equivalent, we choose to differentiate

between them here for two reasons. First, we view signals in general as stochastic processes, while we treat systems as deterministic operators. For this reason, the bounds which will be applied to aliasing of signals and system impulse responses (in general, the Volterra kernels) differ somewhat: the error component of a stochastic process can only be bounded in an average sense (e.g. mean square), while an absolute error bound may be obtained for a deterministic waveform, such as a filter impulse response.

Second, the nature of the cause of aliasing differs between signals and systems. We can analytically force a "known" system to be bandlimited. Thus, the error in system representation will be caused by truncation of (or perhaps by a recursive filter approximation to) the system response. This results from the need to make the system response computable, i.e. calculable in a finite number of operations.

On the other hand, the physically justified assumption of causality, applied to both a system and signal, permits us to avoid truncation of the signal, except on halting the execution of discrete time processing. However, no error is incurred within the segment of the response which has been computed at the time of the halt. Therefore, our primary concern about aliasing error in signals is due to

signal energy outside the assumed bandlimit used to establish the sampling rate.

The remedies available for minimizing aliasing and truncation error may also differ between systems and signals. The means of minimizing aliasing error for signals is to increase the sampling rate of the discrete time processor. The reduction of aliasing in system responses (e.g., Volterra kernels) is achieved by extending the duration of the response approximation by lengthening the truncation window (or by a comparable complexity increase in a recursive filter implementation).

Aliasing and truncation errors have been treated at length by numerous authors [20,21,22,23,24,25]. We offer a summary of relevant results below.

3.4.1 Aliasing Error in a Sampled Non-Bandlimited Signal

When the assumption that a signal (or system response) is bandlimited to one half the sampling rate is violated, equation (2) fails to correctly reconstruct the signal. That is:

$$\bar{x}(t) = \sum_{k=-\infty}^{\infty} x\left(\frac{k}{2W}\right) \operatorname{sinc}\left[2W\left(t-\frac{k}{2W}\right)\right] \neq x(t) \text{ when } X(f) \neq 0 \text{ for all } |f| > W \quad (22)$$

Nevertheless, for many realizable waveforms, it is feasible to establish a frequency interval $(-W,W)$ which

represents the *essential* bandlimit of the signal [26]. Consequently, while aliasing error may be present in the approximation, it may be bounded. Furthermore, if $\lim_{|f|\to\infty} X(f) = 0$ is satisfied (a condition met by a system with a strictly proper transfer function), the bound will be monotonically decreasing for increasing W.

When $x(t)$ is a stochastic process with power spectral density $S_x(f)$, a bound on the mean square error due to aliasing is given by Brown [22]:

$$\mathrm{E}\{[x(t)-\tilde{x}(t)]^2\} \le 8\int_W^\infty S_x(f)\,df \tag{23}$$

The aliasing error introduced by improperly sampling a deterministic signal, such as the impulse response of a linear system, can be upper bounded in absolute value by [23]:

$$|h(t)-\tilde{h}(t)| \le 4\int_W^\infty |H(f)|\,df \tag{24}$$

Observe that equations (23) and (24) are fundamentally different bounds and cannot be directly compared. Equation (23) provides a bound based on the L^2 norm. Equation (24), on the other hand, is based on the L^∞ norm. Therefore, no attempt should be made to compare the two bounds on an equivalent basis. Furthermore, equation (23) implicitly

accounts for the fact that while the absolute error may be large at some instants of time, it is identically zero at the sampling instants.

3.4.2 Signal Truncation Error

Various bounds may also be obtained for the error introduced by truncating the sampling expansion of a bandlimited signal. These bounds are useful in establishing limits on the error introduced in a finite, hence computable, realization of a filter. We present two bounds which apply to the accuracy of a reconstructed signal estimate based only on a finite number of samples about that instant. The first bound, given by Helms and Thomas [25], can be applied only to oversampled signals (i.e., where the bandlimit applied to the signal is less than the Nyquist frequency for the particular sampling rate). A second bound, due to Papoulis [21,27] is more general in its applicability to truncated representations of filter responses; however, it is applicable only to finite energy signals. Under the constraints of a particular situation, one bound may perform better than the other, if, in fact, both are applicable.

The Helms-Thomas bound applies to the partial sum nominally centered on the time instant of interest:

$$x_N(t) = \sum_{n=K-N}^{K+N} x\left(\frac{n}{2W}\right) \operatorname{sinc}\left[2W\left(t-\frac{n}{2W}\right)\right], 0 < N < \infty \qquad (25)$$

where the integer K has a functional dependence on the time, t, at which the error bound is desired:

$$2Wt-\tfrac{1}{2}\le K(t)\le 2Wt+\tfrac{1}{2}$$

Then a bound on the maximum absolute deviation of the finite sampling reconstruction when $x(t)$ is sampled at $t = k/2W$ and $X(f)=0$ for $|f|>rW$, $0<r<1$ is:

$$|x(t)-x_T(t)|\le \frac{4M}{\pi^2 N(1-r)} \qquad -\infty<t<\infty \tag{26}$$

where M is defined as:

$$M=\max|x(t)|\ ,\ -\infty<t<\infty$$

The extension of equation (26) to asymmetric partial sums of the form:

$$x_{N_1,N_2}(t)=\sum_{n=K-N_1}^{K+N_2} x\left(\frac{n}{2W}\right)\operatorname{sinc}\left[2W\left(t-\frac{n}{2W}\right)\right] \tag{27}$$

is given by:

$$|x(t)-x_{N_1,N_2}(t)|\le \frac{2M}{\pi^2(1-r)}\left(\frac{1}{N_1}+\frac{1}{N_2}\right) \tag{28}$$

A basic limitation of either equation (26) or equation (28) is that the bound is applicable only to instants of time between the first and last sampling instants which define the partial sum, i.e. equation (25) or equation (27). Furthermore, these estimates tend to be very poor when the time instant for which the error bound is desired falls close to the point of truncation (i.e., N_1 or N_2 is small). However, the bound may be applied to either finite energy or finite power signals.

The second bound was derived by Papoulis [21,27] and applies to the error at any time - either within or external to the signal truncation window - but is necessarily limited to finite energy signals. When the truncated sampling expansion is given by:

$$x_N(t) = \sum_{n=-N}^{N} x\left(\frac{n}{2W}\right) \operatorname{sinc}\left[2W\left(t-\frac{n}{2W}\right)\right] \tag{29}$$

The energy contained in that portion of the signal discarded by truncation is:

$$E_N = \int_{-W}^{W} \left|X\left(f\right)\right|^2 df - T \sum_{n=-N}^{N} |x(nT)|^2 \tag{30}$$

Then the maximum absolute error is bounded by:

$$|x(t) - x_N(t)| \leq \sqrt{2WE_N} \tag{31}$$

Under appropriate conditions, we may compare the bounds offered by Helms and Thomas (equation (28)) and Papoulis for finite energy signals. First, we must choose a signal bandlimit, B, which is less than W, the Nyquist frequency for the chosen sampling interval, T. Then, we may use the following relationship between the maximum value of a bandlimited signal and its energy [27]:

$$|x(t)| \leq \sqrt{2WE} \tag{32}$$

where: $$E = \int_{-W}^{W} \left| X(f) \right|^2 df \tag{33}$$

By replacing M in equation (28) with the right hand side of equation (32), we obtain:

$$|x(t) - x_{N_1,N_2}(t)| \leq \frac{2\sqrt{2BE}}{\pi^2(1-r)} \left(\frac{1}{N_1} + \frac{1}{N_2} \right) \tag{34}$$

where we have substituted B for the specific bandlimit of the signal, where $B = rW$. Then, the Papoulis bound is:

$$|x(t) - x_N(t)| \leq \sqrt{2BE_N} \tag{35}$$

By computing both equations (34) and (35), the tighter bound may be selected as:

$$\text{Bound} = \min\left\{\frac{2\sqrt{2BE}}{\pi^2(1-r)}\left(\frac{1}{N_1}+\frac{1}{N_2}\right), \sqrt{2BE_N}\right\} \tag{36}$$

If we express the truncation error energy in relation to the total bandlimited signal energy as:

$$E_N = \eta E \tag{37}$$

then, the tighter bound may be selected according to the minimum of:

$$\frac{2}{\pi^2\left(1-r\right)}\left(\frac{1}{N_1}+\frac{1}{N_2}\right), \sqrt{\eta} \tag{38}$$

Clearly, the tighter bound will depend on the fraction of total energy contained in the truncation error signal and the location of the time instant for which the error is to be computed. The former dependency is related both to the relative error parameter, η, and the truncated signal duration as expressed through the parameters N_1 and N_2. Clearly, the minimum value of the Helms-Thomas bound will be obtained for $N_1 = N_2 = N$. In this case, selection of the tighter bound depends on the lesser of:

$$\frac{4}{\pi^2\left(1-r\right)N}, \sqrt{\eta} \tag{39}$$

Ultimately, however, the optimum bound will be determined by the specific signal or impulse response of interest.

3.4.3 *Multidimensional Extension of the Papoulis Signal Truncation Bound*

In order to establish bounds for the errors inherent to finite realizations of the higher order Volterra series terms, we may extend the bound derived by Papoulis to multiple dimensions. Consider the n^{th}-order Volterra series term:

$$y_n(t) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\tau_1, \cdots, \tau_n) x(t-\tau_1) \cdots x(t-\tau_n) d\tau_1 \cdots d\tau_n \tag{40a}$$

$$= \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} H\left(f_1, \cdots, f_n\right) X\left(f_1\right) \cdots X\left(f_n\right) \exp\left(j2\pi t \sum_{i=1}^{n} f_i\right) df_1 \cdots df_n \tag{40b}$$

$$= y_n(\tau_1, \cdots, \tau_n)\big|_{\tau_1 = \cdots \tau_n = t} \tag{40c}$$

In order to establish the normalized energy of the n^{th}-order Volterra kernel, let us evaluate equations (40) for $x(t) = \delta(t)$. Since $X(f) = 1$, we have:

$$y_n(t) = h_n(t, \cdots, t) \tag{41a}$$

$$= \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} H\left(f_1, \cdots, f_n\right) \exp\left(j2\pi t \sum_{i=1}^{n} f_i\right) df_1 \cdots df_n \tag{41b}$$

From equation (40a) it is evident that we must evaluate the response over the n-dimensional space in which $h_n(\ )$ is defined, i.e., not only on the hyperdiagonal defined by: $\tau_1 = \ldots = \tau_n = t$. Therefore, define the n^{th}-order impulse response energy as:

$$E_n = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} |h_n(\tau_1, \cdots, \tau_n)|^2 d\tau_1 \cdots d\tau_n \tag{42}$$

This expression is equivalent to:

$$E_n = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \left| H\left(f_1, \cdots, f_n\right) \right|^2 df_1 \cdots df_n \tag{43}$$

and for bandlimited Volterra kernels, equation (42) may also be expressed as:

$$E^{(n)} = \left(\frac{1}{2B}\right)^n \sum_{i_1=-\infty}^{\infty} \cdots \sum_{i_n=-\infty}^{\infty} \left| h_n\left(\frac{i_1}{2B}, \cdots, \frac{i_n}{2B}\right) \right|^2 \tag{44}$$

Equation (44) is applicable when the n^{th}-order nonlinear transfer function satisfies:

$$H_n\left(f_1, \cdots, f_n\right) = 0, \quad |f_i| > B, \quad i = 1, \cdots, n$$

Then for a truncated sampling expansion of a bandlimited n^{th}-order Volterra kernel:

$$h_{n,N}(\tau_1,\cdots,\tau_n)=\sum_{i_1=-N}^{N}\cdots\sum_{i_n=-N}^{N}h_n\left(\frac{i_1}{2B},\cdots,\frac{i_n}{2B}\right)\prod_{l=1}^{n}\operatorname{sinc}\left[2B\left(\tau_l-\frac{i_l}{2B}\right)\right] \tag{45}$$

the maximum absolute error of the n^{th}-order impulse response is given by:

$$\left|h_n(t,\cdots,t)-h_{n,N}(t,\cdots,t)\right|\le\sqrt{2NBE_N^{(n)}} \tag{46}$$

where the truncation error energy is obtained as:

$$E_N^{(n)}=E^{(n)}-\left(\frac{1}{2B}\right)^n\sum_{i_1=-N}^{N}\cdots\sum_{i_n=-N}^{N}\left|h_n\left(\frac{i_1}{2B},\cdots,\frac{i_n}{2B}\right)\right|^2 \tag{47}$$

The factor, N, on the right hand side of equation (46) reflects the potential bandwidth expansion of the n^{th}-order response relative to the bandwidth restriction imposed in each dimension of the Volterra kernel.

3.5 Volterra Series Truncation Error

It was previously noted that an infinite Volterra series is not computable. Consequently, in order to be a generally useful tool for evaluation of nonlinear systems, the Volterra series for a system to be evaluated must admit to a bound on the error introduced by truncating the series to a finite order, N. Such a bound was obtained by Boyd

[8] by a simple extension of the development of his gain bound function, described in section 2.3.1. We present Boyd's result for completeness.

Let a truncated Volterra series be given by:

$$y^{(N)}(t) = h_0 + \sum_{n=1}^{N} \int_{-\infty}^{\infty} \cdots \int h_n(\tau_1, \cdots, \tau_n) x(t-\tau_1) \cdots x(t-\tau_n) d\tau_1 \cdots d\tau_n \tag{47}$$

where we have included the zero-order term, h_0, to maintain consistency with Boyd's notation. Then:

$$y(t) - y^{(N)}(t) = \sum_{n=N+1}^{\infty} \int_{-\infty}^{\infty} \cdots \int h_n(\tau_1, \cdots, \tau_n) x(t-\tau_1) \cdots x(t-\tau_n) d\tau_1 \cdots d\tau_n \tag{48}$$

We may bound the absolute error as:

$$\begin{aligned} \left| y(t) - y^{(N)}(t) \right| &\le \sum_{n=N+1}^{\infty} \left| \int_{-\infty}^{\infty} \cdots \int h_n(\tau_1, \cdots, \tau_n) x(t-\tau_1) \cdots x(t-\tau_n) d\tau_1 \cdots d\tau_n \right| \\ &\le \sum_{n=N+1}^{\infty} \|h_n\| b^n \end{aligned} \tag{49}$$

where $|x(t)| < b \le \rho$ for all t where ρ is the radius of convergence for the complete (infinite) Volterra series.

This may easily be recognized as the tail of the gain bound function. The convergence of the series implies that for any arbitrary real number, ε, there exists an integer, N, such that:

$$|y(t) - y^{(N)}(t)| < \varepsilon \qquad (50)$$

Consequently, for any system which has a valid Volterra series representation, we may approximate the response to a bounded signal, $x(t)$, which has its greatest absolute value within the radius of convergence of the series, by a truncated series with an absolute error less than ε.

While it is helpful to know that the truncation error implicit in $y^{(N)}(t)$ is bounded, it may be less simple to compute a tight bound for a given system. Alternatively, it may be acceptable to show that amplitudes of frequency components of the k^{th}-order response are less than a factor, μ, times the amplitudes of the linear response components.

3.6 Continuous-Time Filter Approximation in Discrete Time

Previously we indicated that approximation to the response computation of equation (6) is necessary in order to yield a computable response. If we implement the discrete-time response computation by an FIR realization of the system suggested by equations (6) and (9), then an approximation error results from truncation of the infinite duration filter response. On an intuitive level, this is a convenient way to think of the approximation; however, it may be neither the most accurate nor the most computationally efficient approach to a discrete-time

response computation. In the first place, the assumption of a bandlimited impulse response is not satisfied in practice; hence, the sampling expansion for $h(t)$ includes some aliasing error to start.

The classical problem in emulating the behavior of a realizable continuous-time system with a discrete-time computational structure is that there is no unique, error-free mapping from the continuous-time domain (the s-domain) to the discrete-time domain (the z-domain). Consequently, every discrete-time counterpart to a continuous-time prototype system represents an approximation based on optimization with respect to some criterion, typically the minimization of error in some sense (e.g., mean square error, maximum absolute error, etc.). Unfortunately, the error bound obtained in any case is signal specific; that is, it is applicable only with respect to a particular input. We explore the problems associated with controlling the error in discretizing filter structures in the following sections.

3.6.1 Determination of a Signal-Optimized Discrete Filter

The following discussion of the derivation of a discrete-time linear filter follows the presentation given by Kowalczuk [28]. It is presented here for completeness.

Assume that we have a linear system which is described by the following equivalent relationships:

$$Y(s) = H(s)X(s) \tag{51a}$$

and

$$y(t) = h(t) * x(t) = \int_{-\infty}^{\infty} h(\tau)x(t-\tau)d\tau \tag{51b}$$

where $X(s)$, $Y(s)$, and $H(s)$ are, respectively, the Laplace transforms of $x(t)$, $y(t)$, and $h(t)$.

For the continuous time system, the $H(s)$ which is defined by:

$$H(s)=Y(s)/X(s) \tag{52}$$

is perfectly general; it will be the same for any input $X(s)$ and the corresponding system response $Y(s)$.

In the discrete-time case, however, the situation is different. Let us determine the discrete-time transfer function $H_d(z)$ as follows. Choose a training input signal, $x_1(t)$. Using the continuous-time relationship, equation (51b), compute the continuous-time response, $y_1(t)$. For each of these signals, determine their respective z-transforms,

$X_1(z)$ and $Y_1(z)$, with reference to a set of sampling instants, $t_k = kT$. It is preferable that the sampling interval, T, be chosen to satisfy the Nyquist criterion for the essential bandwidth of $x_1(t)$, although this has no direct bearing on the immediate derivation of $H_d(z)$.

We may now define the discrete-time transfer function $H_{d,1}(z)$ as:

$$H_{d,1}(z) = Y_1(z)/X_1(z) \tag{53}$$

Alternatively, we may repeat the same procedure for a different training signal, $x_2(t)$, to obtain $H_{d,2}(z)$. In general:

$$H_{d,1}(z) \neq H_{d,2}(z)\,. \tag{54}$$

Accordingly, error-free discretization of a continuous-time filter can be performed only with respect to a specific signal; that filter will yield errors for other signals. Moreover, the z-domain transfer function obtained for a particular signal with reference to a sampling interval T_1, will in general, be different than the transfer function obtained with reference to a sampling interval T_2. While the specific transfer function obtained

for a particular continuous-time prototype system will precisely respond to its training signal for the particular sampling interval chosen, it will in general, produce better representations of other responses for shorter sampling intervals.

The difficulties of designing a discrete-time filter (or system) by preserving the response to a particular input are well known; the inadequacies of the impulse-invariance design technique have been understood for some time [29]. In particular, the poor reproduction of the continuous-time frequency response in the corresponding discrete-time filter makes the impulse-invariance technique a poor choice in many applications.

3.6.2 Discrete Filter Determination by s-to-z Mapping

As an alternative to determining the z-domain transfer function for a system based on a specific signal, contemporary signal processing design methods rely heavily on techniques for mapping s-domain transfer functions into the z-domain by some form of s-to-z transformation. We present a brief description of a family of mappings based on numerical integration formulas. The presentation follows the work by Schneider, Kaneshige, and Groutage [30]. A significant contribution of that paper is the determination of the error introduced by discrete-time

approximation of a continuous-time prototype filter for various s-to-z mappings and sampling intervals. It must be noted, however, that these error assessments were made with respect to a specific input signals. No perfectly general bound on the approximation error for filter discretization is evident in the literature.

Based on interpretation of s^{-1} as an integration operator, where s is the Laplace s-operator, numerical integration techniques may be brought to bear on the problem of establishing a discrete-time counterpart to a continuous-time transfer function, $H(s)$. Schneider, *et al* [30] offer an analysis of the performance of several higher-order s-to-z mapping functions based on the Adams-Moulton family of numerical integration formulas. Among them, the first-order mapping function is the familiar bilinear transformation:

$$s = \frac{2}{T}\frac{z-1}{z+1} \tag{55}$$

The second-order mapping function takes the form:

$$s = \frac{12}{T}\frac{z^2-z}{5z^2+8z-1} \tag{56}$$

In each case, the z-domain transfer function is obtained by substituting the right hand side of the appropriate mapping

function for s in the continuous-time system transfer function which is to be approximated.

As with the signal-specific approach to deriving a discrete-time transfer function for a continuous-time prototype system, the s-to-z mapping technique benefits, in terms of accuracy, from smaller sampling intervals, T. More important, however, the error in reproducing the response of a continuous-time prototype for a specific input is significantly reduced when higher-order mappings are utilized.

The penalty for applying a higher-order s-to-z mapping function is that the complexity of the resulting z-domain transfer function is increased. The discrete-time filter resulting from an m^{th}-order mapping being applied to an n^{th} order continuous-time prototype transfer function has order mn. Consequently, the computational burden is increased by using a mapping of higher order than required to satisfy the accuracy requirements of a particular application.

In an example presented by Schneider, *et al* [30], the reduction of rms response error for a sinusoidal input decreased approximately linearly with the sampling rate $1/T$. The slope of the error decrease was proportional to the order of the mapping function which had been applied to

obtain the discrete-time transfer function from the continuous-time transfer function.

3.6.3 Reduction of Filter Discretization Error for Bandlimited Applications

In any simulation application, we are necessarily interested in essentially bandlimited systems; otherwise, aliasing error will distort results beyond usefulness. Therefore, it is beneficial to consider filters which are optimized for bandlimited applications.

If we sample a filter impulse response, $h(t)$, at instants $t_k = kT, \quad -\infty \le k \le \infty$, then the frequency response of the sampled filter is related to the original analog filter frequency response by:

$$H_s(f) = \begin{cases} \sum_{k=-\infty}^{\infty} H\left(f - \frac{k}{T}\right), & -\frac{1}{2T} \le f \le \frac{1}{2T} \\ 0, & \text{otherwise} \end{cases} \tag{57}$$

where $H(f)$ is the Fourier transform of $h(t)$. Consequently, the difference between $H(f)$ and $H_a(f)$ is not only the truncation of $H(f)$ beyond $|f|=1/2T$, but also the inclusion of the aliasing components within the passband of $H_s(f)$.

Furthermore, when the input signal to a simulation is essentially bandlimited to W Hertz, frequency components of

the filter response beyond the essential bandwidth of the signal offer no benefit to the response. Therefore, if the analog filter frequency response is arbitrarily set to zero for $|f| > W$, then aliasing will be eliminated. (Of course, the bandlimited filter necessarily has an infinite duration impulse response; therefore, some signal (impulse response) truncation error will be implicit in an FIR realization of the filter.)

3.7 Composite Error Bound

Having considered each of the sources of error -- input signal aliasing, filter impulse response truncation, and Volterra series truncation -- it is necessary to establish the manner in which these errors manifest themselves jointly in the discretized nonlinear system response. Clearly, the truncation of the Volterra series simply removes the series terms beyond the truncation order from the response estimate. The approximation inherent to the terms which are retained consists of processing a corrupted (aliased) input with imperfect realizations, in discrete time, of the first N Volterra filters (i.e., implementations of the n^{th}-order convolutions with the n^{th}-order Volterra kernel). Expressed in continuous time (i.e., the discrete-time response is presumed to be

reconstructed, as for example, in equation (8)) the N^{th}-order approximation to the complete nonlinear system response is:

$$\hat{y}^{(N)}(t) = y(t) + e^{(N)}(t) \tag{58}$$

where the (N) superscript denotes that this is an N^{th}-order approximation. The error term may be written:

$$e^{(N)} = -\sum_{i=N+1}^{\infty} y_i(t) + \sum_{i=1}^{N} \left\{ \mathrm{H}_i[x] - \hat{\mathrm{H}}_i[\hat{x}] \right\} \tag{59}$$

In equation (59) the first summation on the right hand side is the Volterra series truncation error. The second summation reflects the term-by-term difference between the ideal Volterra series term and the approximation to the Volterra filter as applied to the aliased input. While this is not precisely the form in which we have obtained the individual contributions to the error, we may expand the second summation term to obtain:

$$e^{(N)} = -\sum_{i=N+1}^{\infty} y_i(t) + \sum_{i=1}^{N} \left\{ \mathrm{H}_i[x] - \hat{\mathrm{H}}_i[x] \right\} + \sum_{i=1}^{N} \left\{ \hat{\mathrm{H}}_i[x] - \hat{\mathrm{H}}_i[\hat{x}] \right\} \tag{60}$$

The three summation terms in equation (60) may now be associated with the Volterra series truncation, filter approximation and aliasing errors. Figure 1 depicts the

realization of a system as a truncated Volterra series and shows the points at which the various sources of error enter the system.

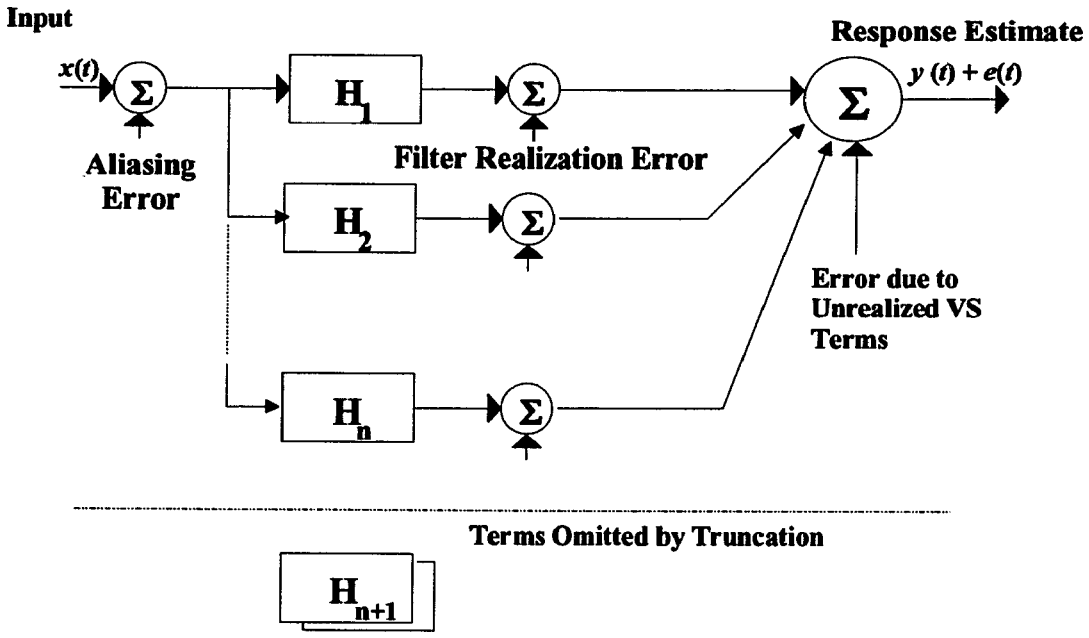


Figure 1: Composite Error Model

We may begin to establish a bound on the error expressed in equation (60) by writing:

$$\left|e^{(N)}\right| \le \left|\sum_{i=N+1}^{\infty} y_i(t)\right| + \sum_{i=1}^{N} \left|\left\{\mathrm{H}_i[x] - \hat{\mathrm{H}}_i[x]\right\}\right| + \sum_{i=1}^{N} \left|\left\{\hat{\mathrm{H}}_i[x] - \hat{\mathrm{H}}_i[\hat{x}]\right\}\right| \qquad (61)$$

When $|x(t)| < b$ for a value of b within the radius of convergence for the Volterra series, we may substitute equation (3-49) into equation (60) for the Volterra series

truncation term. For the filter realization error term, we may substitute the expression:

$$\sum_{i=1}^{N} \left| \left\{ \mathrm{H}_i[x] - \hat{\mathrm{H}}_i[x] \right\} \right| = \sum_{i=1}^{N} \left\| h_{e,n} \right\| b^n \tag{62}$$

where $\|h_{e,n}\|$ is the norm of the nth-order filter realization error defined following equation (2-3). This approach is outlined for the linear component in Appendix A (Section A.5).

The aliasing error was bounded in a mean square sense in Section 3.4 for stochastic input signals. While we cannot obtain an absolute bound for the aliasing error in this case, we can choose to establish an $r\sigma$ bound, where σ is the root mean square aliasing error and r is any arbitrary positive constant. To establish an $r\sigma$ bound, we choose an appropriately large value of r that the probability of the aliasing error magnitude exceeding $r\sigma$ is sufficiently small. Thus for the aliasing term, we obtain:

$$\sum_{i=1}^{N} \left| \left\{ \hat{\mathrm{H}}_i[x] - \hat{\mathrm{H}}_i[\hat{x}] \right\} \right| \leq \sum_{i=1}^{N} \left\| \hat{h}_n \right\| \left[b^n - (b + r\sigma)^n \right] \tag{63}$$

where the inequality applies in the $r\sigma$ sense, i.e., to cases where the aliasing error is less than $r\sigma$. In equation (63) we define σ as:

$$\sigma = \left[8 \int_{W}^{\infty} S_x(f) df \right]^{\frac{1}{2}} \tag{64}$$

Equation (61) may now be rewritten as:

$$|e^{(N)}| \leq \sum_{n=N+1}^{\infty} \|h_n\| b^n + \sum_{n=1}^{N} \|h_{e,n}\| b^n + \sum_{n=1}^{N} \left\|\hat{h}_n\right\| \left[b^n - (b + r\sigma)^n \right] \tag{65}$$

The bound expressed by equation (65) applies in the sense that the error magnitude is less than or equal to the expression on the right-hand side of equation (65) with probability equal to the probability that the aliasing error magnitude is less than $r\sigma$.

CHAPTER 4

The Bandlimited Volterra Series

In general, the response of an N^{th}-order Volterra system occupies up to N times the bandwidth of the system input. Therefore, a complete discrete-time response computation entails the determination of response sample values for N times as many samples as required to represent the input. As a consequence, even for small values of N, the computational burden of computing a discrete-time response approximation may become enormous when the number of response samples and the multidimensional convolution summation are considered.

Although it may be tempting to reduce the computational burden by undersampling, failure to account for the bandwidth expansion necessarily produces a response which is corrupted by aliasing. Undersampling, therefore, does not provide a reduced-effort alternative. We show, however, that a portion of the complete response can be computed at a reduced level of effort. Specifically, that part of the system response which shares the same passband as the input signal can be determined without the need to compute response samples at the N-1 interpolated sampling instants (with respect to the Nyquist samples of the input)

which are required to faithfully construct the complete response. The result is that the desired, i.e. "in-band", part of the system response is computed at the Nyquist rate of the input signal, without aliasing and without any explicitly implemented interpolation or decimation operations.

4.1 Scope of the Compuational Burden of the Discrete-Time Volterra Series

In order to facilitate comparisons of computationally efficient discrete-time techniques to a direct Volterra series computation of a system response, we first obtain a reference point in the form of an estimate of the computational complexity of a direct discrete-time Volterra series realization. This will provide both the motivation for the Bandlimited discrete-time Volterra series and a basis for measuring the improvement in computational efficiency obtained. Below, we derive an estimate of the computational effort required to compute a full-bandwidth response using a truncated Volterra series for a bandlimited input signal. The metric applied is the number of multiplication operations necessary to compute a one-second segment of the response (not including any initialization).

Assume that the input signal to a Volterra system is bandlimited to W Hertz. Then the minimum sampling rate, i.e., the Nyquist rate, for the input is $R=2W$. An N^{th}-order Volterra series will, in general, expand the signal bandwidth by a factor of N, resulting in a required sampling rate of NR for the response. Efficient discrete-time processing will seek to avoid oversampling and match this rate as closely as possible, consistent with satisfying the accuracy requirements imposed on the processing.

To minimize the aliasing error in discrete-time represention of the Volterra kernels, the associated (non)linear transfer functions should be bandlimited to one half the selected sampling rate before discretization is performed. Since the input is bandlimited, however, this has no direct effect on the computed response.

4.1.1 A Computational Estimate for the Linear Response Component

While the computational effort required to compute an Nth-order response is dominated by the Nth-order term, it is easier to grasp the difficulties of Nth-order response

computation by first examining the computation of the Nth-order response for a first-order (linear) system.

The first-order response computation is most easily considered in a finite form related to equation (3-9). Let us assume that an acceptable-fidelity approximation to the infinite summation is obtained using:

$$\hat{y}_1\left(\frac{n}{2W}\right) = \frac{1}{2W}\sum_{k=N_1}^{N_2} h\left(\frac{k}{2W}\right)x\left(\frac{n-k}{2W}\right) = \frac{1}{2W}\sum_{k=n-N_1}^{n-N_2} h\left(\frac{n-k}{2W}\right)x\left(\frac{k}{2W}\right) \quad (1)$$

The determination of the values N_1 and N_2 is discussed in Appendix A.

Equation (1) indicates that we must compute $N = N_2 - N_1 + 1$ terms in the summation for each linear response sample. We shall refer to the number of coefficients, N, as the length, L_1, of the convolution sum. Each term in the linear convolution sum requires a single multiplication ($M_1 = 1$). Thus, $L_1M_1 = N$ multiplications per response sample are required.

Response samples must be computed at the input signal's sampling rate. Therefore, the discrete-time linear response component requires computation of $R_1 = 2W$ samples per second with L_1M_1 multiplications per sample. Thus, the compuational complexity of a linear system with input bandwidth W is on the order of:

$$C_1 = R_1 L_1 M_1 \tag{2}$$

Substituting the previously determined values for each factor in the computational complexity expression, we obtain:

$$C_1 = 2WN \tag{3}$$

Recognizing that the number of terms, N, in the filter approximation is proportional to the bandwidth, W, of the input and the time-bandwidth product of the bandlimited impulse response of the filter, we may alternatively write equation (3) as:

$$C_1 \propto \beta W^2 \tag{3a}$$

where β is the time-bandwidth product of the impulse response for the established accuracy requirements.

4.1.2 A Computational Estimate for the Second Order Response Component

Maintaining the assumption of a W-bandlimited input signal, the general second-order Volterra series response will exhibit spectral components to a maximum frequency of

$2W$. Therefore, samples of the second-order response must be computed at a rate $R_2=4W$. Equation (3-33) provides the basis for our estimate of the computational effort required to determine the second-order discrete-time response. Substituting finite limits of summation, we obtain:

$$\hat{y}_2(n)=\left(\frac{1}{4W}\right)^2\sum_{k=N_3}^{N_4}\sum_{l=N_5}^{N_6}h_2\left(\frac{k}{4W},\frac{l}{4W}\right)x\left(\frac{n-k}{4W}\right)x\left(\frac{n-l}{4W}\right) \tag{4}$$

Recognizing (see equation (2-53)) that the second-order kernel is necessarily symmetric, the summation limits should be chosen to be equal, i.e. $N_3=N_5$ and $N_4=N_6$.

Furthermore, the symmetry of the kernel permits us to reduce the computational effort. Recognizing that:

$$h_2\left(\frac{k}{4W},\frac{l}{4W}\right)x\left(\frac{n-k}{4W}\right)x\left(\frac{n-l}{4W}\right)=h_2\left(\frac{l}{4W},\frac{k}{4W}\right)x\left(\frac{n-l}{4W}\right)x\left(\frac{n-k}{4W}\right)$$

we may rewrite equation (4) as:

$$\hat{y}_2(n)=2\sum_{k=N_3}^{N_4}\sum_{l=N_3}^{k-1}h_2\left(\frac{k}{4W},\frac{l}{4W}\right)x\left(\frac{n-k}{4W}\right)x\left(\frac{n-l}{4W}\right)+\sum_{k=N_3}^{N_4}h_2\left(\frac{k}{4W},\frac{k}{4W}\right)x\left(\frac{n-k}{4W}\right)x\left(\frac{n-k}{4W}\right) \tag{4a}$$

The continuous-time second-order Volterra kernel will have an essential duration in each dimension which is on the order of that applicable to the first-order kernel. The second-order nonlinear transfer function, equation

(2-90) necessarily falls off at a faster rate than does the first-order transfer function because the roll-off of the $H_1(f_i)$ terms is accentuated by the $H(f_1 + f_2)$ term. This suggests a somewhat longer essential duration than that of the first order response; however, the overall magnitude of the second-order term (as expressed by the a_2 coefficient in equations (2-53) and (2-90)) is expected to be smaller. This implies that the absolute error contribution due to second-order response truncation is relatively less than that of the first-order response component. Thus it permits the second-order term to accept a greater relative error contribution and correspondingly shorter essential duration. Consequently we shall base our estimates of the computational load on the same essential duration, $T = N/2W$, which was used for the first order response estimate. The number of FIR coefficients for the discrete-time second-order kernel must, however, be based on the R_2 sampling rate. This means that the number of coefficients *in each dimemsion* will be approximately double the number required to represent the first-order response[1]. Therefore, we have $L_2 = 2N = N_4 - N_3 + 1$, and since the

[1] As the density of filter coefficients increases to match the required sampling rate, the total number of coefficients must increase by the same proportion to span the same essential duration.

second-order response computation is two-dimensional, the effort will be proportional to length squared.

Within each term to be computed in the summation, the second-order response requires two multiplications, hence $M_2 = 2$. Therefore, each term also requires additional effort to compute.

The resultant computational effort required for the second-order response component, as expressed in equation (4), is:

$$C_2 = R_2 L_2^2 M_2 \tag{5}$$

Substitution of the specific values yields:

$$C_2 = (4W)(2N)^2(2) = 32\,WN^2 \tag{6}$$

When the computation is performed as indicated in equation (4a), we obtain, in place of equation (5):

$$C_{2,S} = R_2 \tfrac{1}{2} L_2 (L_2 + 1) M_2 \tag{5a}$$

For large L_2, the value of equation (5a) is approximately one half the value computed in equation (5), i.e.:

$$C_{2,S} \approx \tfrac{1}{2} R_2 L_2^2 M_2, \qquad L_2 >> 1 \tag{5b}$$

4.1.3 *Higher Order Response Computational Estimates*

The foregoing discussion of computational complexity may be generalized in a straightforward manner; the n^{th}-order, asymmetric discrete-time response computation[2] requires on the order of:

$$C_n = R_n L_n^n M_n = (2nW)(nN)^n(n) = 2WN^n n^{n+2} \tag{7}$$

multiplications. Exploitation of the symmetry in an n^{th}-order symmetrized kernel permits the computational burden to be reduced by a factor of $n!$ in the limit. Thus, we obtain:

$$C_{n,S} \approx \frac{1}{n!} R_n L_n^n M_n \approx \frac{1}{n!}(2nW)(nN)^n(n) \approx \frac{1}{(n-1)!} 2WN^n n^{n+1} \tag{7a}$$

It is apparent that even for small values of N, the n^{th}-order response computational effort for asymmetric kernels grows enormously fast due to the n^{n+2} factor. For

---

[2] Strictly speaking, we are discussing the computational burden of the N^{th}-order response term only for a Volterra series truncated to N terms. For the N^{th}-order response term in a series truncated to a number of terms greater than N, the N^{th}-order response component would likely be expressed at the sampling rate for the maximum order of the series (otherwise, interpolation would also be required in order to express the composite response). The compuational effort would be correspondingly greater, although the highest order response component will dominate the computational effort.

$n=2$, the value of this factor is 16; for $n=3$, it becomes 243; and for $n=4$, it grows to 4096. With symmetric kernels, the situation is improved somewhat, although full realization of the symmetry benefits comes only with long filters. Moreover, the structure of computations to exploit the symmetry is more complex. In the symmetric case, the preceding values become (approximately) for $n=2$, the value of the $n^{n+1}/(n-1)!$ factor is 8; for $n=3$, it becomes approximately 40; and for $n=4$, it grows to approximately 170. While the decrease in these factors is substantial, the overall computations are unwieldy for any but the shortest impulse response durations.

4.2 Definition of the Response of Interest

In many practical situations, the nonlinear part of a system response (corresponding to the terms $n\geq 2$ in the Volterra series expansion) represents unwanted distortion. This is not to suggest that these terms may be neglected; the distortion is real and must be considered as a part of the response. However, requiring that the essential bandwidth [26] of the input be finite (e.g., its essential bandwidth is W), means that the components of the response which lie outside the essential bandwidth of the input

necessarily result from the nonlinear effects of the system.

In such systems, the out-of-band response components can be effectively eliminated by zonal filtering. Figure 1 shows a nonlinear system with a lowpass filter cascaded with the nonlinear system to eliminate out-of-band response components. If the nonlinear system has a Volterra series representation of the form presented in Chapter 2, then the essential bandwidth of the desired response is effectively limited to the essential bandwidth of the linear (first-order) transfer function.

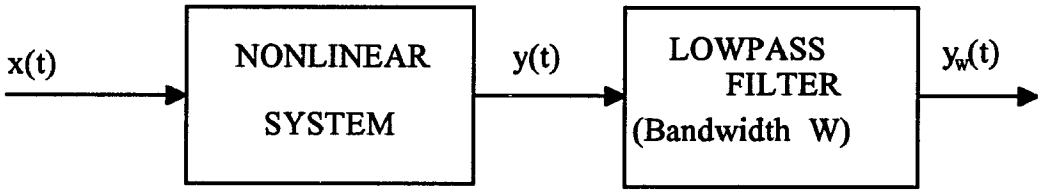


Figure 1: A Zonally-Filtered Nonlinear System

4.2.1 Proposition

Because we have constrained the input processes under consideration to be essentially bandlimited, we may represent any input signal by its Nyquist samples with an

error bounded as described in Chapter 3. Similarly, the response of a system which is bandwidth constrained as shown in Figure 1 must necessarily be representable by samples taken at the Nyquist rate corresponding to the filter bandwidth[3].

All of the information about the input process is contained in its Nyquist samples. We postulate, therefore, that an operator exists which computes samples of the discrete-time, bandlimited, truncated Volterra series response directly from samples of the input process without the need to operate on any additional data samples[4]:

$$\sum_{k=-\infty}^{\infty} {}^{(N)}y(kT) = \mathrm{V}_{\mathrm{bl}}\left[\sum_{k=-\infty}^{\infty} x(kT)\right] \tag{8}$$

We shall call this operator, V_{bl}, the bandlimited, discrete-time Volterra series operator. It must be recognized that this is *not* the same operator as that which computes every N^{th} sample of the discrete Volterra series response; such an operation would necessarily alias the

---

[3] We shall choose the filter bandwidth to be the the same as the essential bandwidth of the input process; however, any suitable limit may be selected. In the event that a response bandwidth is selected which is different than that of the input, the sampling rate should be based on the greater of the two bandwidths.

[4] In effect, the operator could generate intermediate input values by interpolation and discard intermediate response values by decimation in a manner transparent to external observation.

out-of-band components of the response back into the bandwidth-of-interest rather than eliminating these components as we propose.

4.3 Construction of The Second-Order Component of the Bandlimited Discrete-Time Volterra Series

It will be easier to grasp the concept of a bandlimited Volterra series if we approach the subject in terms of a specific case. Therefore, we begin our presentation by considering the second-order term in the bandlimited discrete-time Volterra series.

Let the ideal lowpass filter of bandwidth W be designated by:

$$R_W(f) = \begin{cases} 1, & |f| \le W \\ 0, & |f| > W \end{cases} \tag{9a}$$

The filter impulse response is:

$$r_W(t) = \frac{1}{2W}\operatorname{sinc}(2Wt) \tag{9b}$$

Let the second-order response of a Volterra system be given by $y_2(t)$. Then the lowpass portion of y_2 is given by:

$$y_{2,W}(t) = y_2(t) * r_W(t) \tag{10}$$

Employing the general expression for a second-order Volterra series response, we may write equation (10) as:

$$y_{2,W}(t) = \int_{-\infty}^{\infty} \iint_{-\infty}^{\infty} h_2(\tau_1,\tau_2)x(\tau-\tau_1)x(\tau-\tau_2)d\tau_1 d\tau_2\, r_W(t-\tau)d\tau \qquad (11a)$$

or

$$y_{2,W}(t) = \int_{-\infty}^{\infty} \iint_{-\infty}^{\infty} h_2(\tau-\tau_1,\tau-\tau_2)x(\tau_1)x(\tau_2)d\tau_1 d\tau_2\, r_W(t-\tau)d\tau \qquad (11b)$$

By interchanging the order of integration in equations (11a) and (11b), we obtain:

$$y_{2,W}(t) = \iint_{-\infty}^{\infty} h_2(\tau_1,\tau_2) \int_{-\infty}^{\infty} x(\tau-\tau_1)x(\tau-\tau_2)r_W(t-\tau)d\tau\, d\tau_1 d\tau_2 \qquad (12a)$$

or

$$y_{2,W}(t) = \iint_{-\infty}^{\infty} x(\tau_1)x(\tau_2) \int_{-\infty}^{\infty} h(\tau-\tau_1,\tau-\tau_2)r_W(t-\tau)d\tau\, d\tau_1 d\tau_2 \qquad (12b)$$

Equation (12a) expresses the bandlimited second-order response as the two-dimensional convolution of the second-order kernel with a bandlimited signal product. Alternately, equation (12b) expresses the bandlimited second-order product in terms of a two-dimensional convolution of a second-order signal product with a bandlimited second-order kernel. These equations motivate the following approaches to bandlimitation of the second-order Volterra series response.

4.3.1 Bandlimitation of the Signal Product

If we assume that the system input, $x(t)$, is essentially bandlimited to W (or less), then the bandlimited signal product of equation (12a) may be written in terms of the sampling expansion of $x(t)$:

$$u_2(t;\tau_1,\tau_2) = \int_{-\infty}^{\infty} x(\tau-\tau_1)x(\tau-\tau_2)r_W(t-\tau)d\tau$$

$$= \int_{-\infty}^{\infty}\left[\sum_{m=-\infty}^{\infty} x\left(\frac{m}{2W}-\tau_1\right)\operatorname{sinc}\left(2W\left(\tau-\frac{m}{2W}\right)\right)\right] \tag{13}$$

$$\left[\sum_{n=-\infty}^{\infty} x\left(\frac{n}{2W}-\tau_1\right)\operatorname{sinc}\left(2W\left(\tau-\frac{n}{2W}\right)\right)\right][2W\operatorname{sinc}(2W(t-\tau))]d\tau$$

Interchanging the order of integration and summation yields:

$$u_2(t;\tau_1,\tau_2) = \sum_{m=-\infty}^{\infty} x\left(\frac{m}{2W}-\tau_1\right)\sum_{n=-\infty}^{\infty} x\left(\frac{n}{2W}-\tau_2\right) \tag{14}$$

$$\int_{-\infty}^{\infty}\operatorname{sinc}\left[2W\left(\tau-\frac{m}{2W}\right)\right]\operatorname{sinc}\left[2W\left(\tau-\frac{n}{2W}\right)\right]\operatorname{sinc}[2W(t-\tau)]d\tau$$

Equation (14) gives an expression for the bandlimited signal product in a form which might be useful if the integral could be obtained in closed form by a simple means which included the delays in a parametric fashion. Unfortunately, this sort of representation is not available. Nevertheless, even if, for a specific input

case or for a class of inputs, equation (13) can be solved in closed form, we must also find a means for representing the second-order Volterra kernel such that a minimized-sampling-rate expansion of the kernel is valid.

4.3.2 Bandwidth Restriction of the Volterra Kernel

We may approach equation (12b) in the same manner that we considered bandlimiting the signal product in the second-order Volterra series response. Using the expression developed for the second-order kernel in Chapter 2, we write the inner integral of equation (12b) as:

$$h_{2,W}(t-\tau_1,t-\tau_2)=-\frac{a_2}{R^2}\int_{-\infty}^{\infty}h(\alpha-\tau_1)h(\alpha-\tau_2)\int_{-\infty}^{\infty}h(\tau-\alpha)\,2W\,\mathrm{sinc}\,[2W(t-\tau)]d\tau d\alpha \quad (15)$$

If we define the result of the convolution integral with respect to τ as $h_W(t\text{-}\alpha)$, we may rewrite equation (15) as:

$$h_{2,W}(t-\tau_1,t-\tau_2)=-\frac{a_2}{R^2}\int_{-\infty}^{\infty}h(\alpha-\tau_1)h(\alpha-\tau_2)h_W(t-\alpha)d\alpha \quad (16)$$

While equation (16) provides an expression for the bandlimited kernel, it is not obvious that a useful simplification can be obtained in a general way in this form. The difficulty is, in part, due to the fact that we are attempting to apply a one-dimensional bandwidth restriction to an inherently two-dimensional function. In

addition, the fact that we attempted to separate the kernel and signal product before applying the bandlimitation made it difficult to take a systems approach. We may address the issue more directly by taking a frequency-domain approach to the problem.

4.3.3 Frequency Domain Modification of the Nonlinear Transfer Function

Bandlimiting a linear filter to a maximum frequency, W, may be accomplished analytically by truncating the transfer function of the filter at $|f| = W$ as described in Section 3.5.3. This can be achieved by multiplying the original transfer function by a "spectral mask" which is the frequency-domain representation of the ideal lowpass filter:

$$M_1(f) = \begin{cases} 1, & |f| \le W \\ 0, & |f| > W \end{cases} \tag{17}$$

It was demonstrated in Chapter 2 that the response at a frequency $f = f_r$ is due, for the second-order case, to complex exponential excitation components at all frequencies f_1 and f_2 such that $f_1 + f_2 = f_r$. This suggests that the response can be constrained to contain no response components at frequencies $|f| > W$ if the second-order

nonlinear transfer function, $H_2(f_1,f_2)$, is multiplied by a spectral mask which is zero for all f_1,f_2 such that $|f_1+f_2|>W$. Conversely, preserving the characteristics of $H_2(f_1,f_2)$ at frequencies f_1 and f_2 such that $|f_1+f_2|<W$ assures that the computed response will include the desired "in-band" distortion components. Furthermore, since we intend to sample the two-dimensional Volterra kernel at the minimum allowable rate, we also want to set the mask to zero for $|f_i|>0$, $i=1,2$. Since we intend to consider only W-bandlimited input signals, this causes no unintended error because there is no input component (by assumption) at the discarded frequencies. The resulting spectral mask is:

$$M_2(f_1,f_2)=\begin{cases} 0, & |f_1|>W \\ 0, & |f_2|>W \\ 0, & |f_1+f_2|>W \\ 1, & \text{otherwise} \end{cases} \tag{18}$$

The mask characteristic is shown graphically in Figure 2.

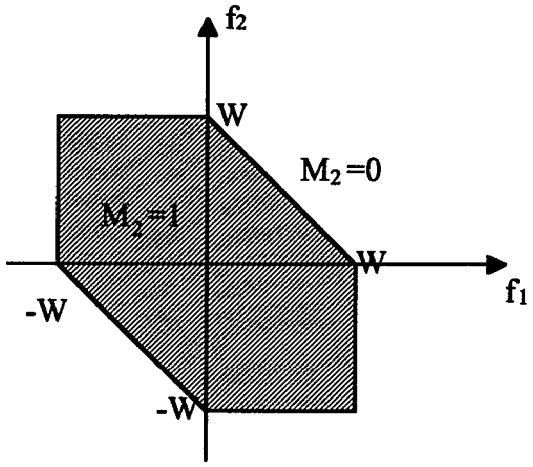


Figure 2: Bandlimiting Mask for Second-Order Nonlinear Transfer Function

The bandlimited, second-order nonlinear transfer function is:

$$H_{2,W}\left(f_1,f_2\right)=H_2\left(f_1,f_2\right)M_2\left(f_1,f_2\right) \tag{19}$$

The corresponding second-order bandlimited Volterra kernel is:

$$h_{2,W}(\tau_1,\tau_2)=h_2(\tau_1,\tau_2)*m_2(\tau_1,\tau_2) \tag{20}$$

where: $$m_2(\tau_1,\tau_2)=F^{-1}\left[M_2\left(f_1,f_2\right)\right] \tag{21}$$

and where the * implies a two-dimensional convolution.

4.3.4 *The N-Dimensional Bandlimited Nonlinear Transfer Function*

The approach taken to obtain a bandlimited second-order nonlinear transfer function may be generalized to arbitrary order. For the n^{th}-order transfer function, in order to eliminate response components at frequencies greater than W, we require:

$$H_n\left(f_1,\cdots,f_n\right)=0, \quad \left\{f_i, i=1,\cdots,n \middle| \left|\sum_{i=1}^{n} f_i\right| > W\right\} \tag{22a}$$

and: $$H_n\left(f_1,\cdots,f_n\right)=0, \quad |f_i| > W,\ i=1,\cdots n \tag{22b}$$

Equation (22a) expresses the fundamental requirement that the response contain no components at frequencies $|f| > W$. Equation (22b) permits the sampling of the associated n^{th}-order Volterra kernel at a rate $2W$ in each of the n dimensions by recognizing that the input signal is W-bandlimited.

4.4 *A Computational Complexity Estimate for the Bandlimited Volterra Series*

The advantage of the bandlimited discrete-time Volterra series is that the computational complexity of computing a response is drastically reduced. The bandlimited Volterra series eliminates the need to sample

both the Volterra kernels and the system response at a rate consistent with an N-fold bandwidth expansion relative to the W-bandwidth of the input. Therefore, both the number of response samples to be computed and the length of *each dimension* of the convolution sum are reduced by a factor N.

The computation of the N^{th}-order term in the series will still dominate the computational complexity of the bandlimited response. The complexity of this computation is:

$$C_{W,N} = R_{W,N} L_{W,N}^{N} M_{W,N} \tag{23}$$

In equation (23), the W subscript indicates that the terms apply to the W-bandlimited response computation. The rate, $R_{W,N}$ is $2W$, identically the rate R_1 used in section 4.2 to express the sampling rate required for the linear response term computation.

The length term is decreased from that value given in section 4.1.2, because the Volterra kernels need only be sampled at intervals corresponding to the required computational rate, hence:

$$L_{W,N}^{N} = (2WT)^{N}$$

where T is the essential duration of the linear impulse response.

The multiplication count for each Volterra kernel coefficient, $M_{W,N}$ is the same as the multiplication count M_N previously described as it has no bandwidth dependence. Thus equation (23) becomes:

$$C_{W,N} = R_1 L_{W,N}^N M_N = (2W)(2WT)^N(N) = (2W)^{N+1} T^N N \qquad (24)$$

By comparison with equation (7), the computational complexity has been reduced by a factor of N^{N+1}. For $N=2$, this represents an 8-fold reduction of computation; for $N=3$ it is a factor of 81 decrease; and for $N=4$, computational complexity is reduced by a factor of 1024 - three orders of magnitude. This decrease is sufficient to make discrete-time Volterra series computationally feasible for engineering problems where previously they represented a burden which could not be afforded.

CHAPTER 5

Serial Realization of the Volterra Series

A direct realization of the multidimensional convolution operators is not the only approach to synthesis of a Volterra series response. We illustrate here an approach which we call the Serial Realization of the Volterra Series.

5.1 The Basis for the Serial Realization

Based on the form of the Volterra kernels and their associated nonlinear transfer functions, obtained as demonstrated in sections 2.5.2 and 2.5.3, an alternative realization of the n^{th}-order response can be synthesized. Referring to equation (2-46) and the subsequent derivations of the first several Volterra kernels, it may be seen that every n^{th}-order response component[1] has the form:

$$y_{n,c}(t) = -\int_{-\infty}^{\infty} h(t-\tau) a_k \prod_{i=1}^{k} y_{l_i}(\tau) d\tau \qquad (1)$$

where: $$n = \sum_{i=1}^{k} l_i, \quad k \geq 2. \qquad (1a)$$

Thus a realization based on equation (1) can be constructed as shown in Figure 1. The k-fold product is filtered by

---

[1] The n^{th}-order term in a Volterra series may have several components, i.e. a number of sets $\{l_1,\ldots,l_k\}$ which satisfy equation (1a).

the associated linear characteristic of the system and weighted by the k^{th} degree series expansion coefficient[2] to yield an n^{th}-order response component. The n^{th}-order character of the response is determined by the sum of the k orders of the product terms.

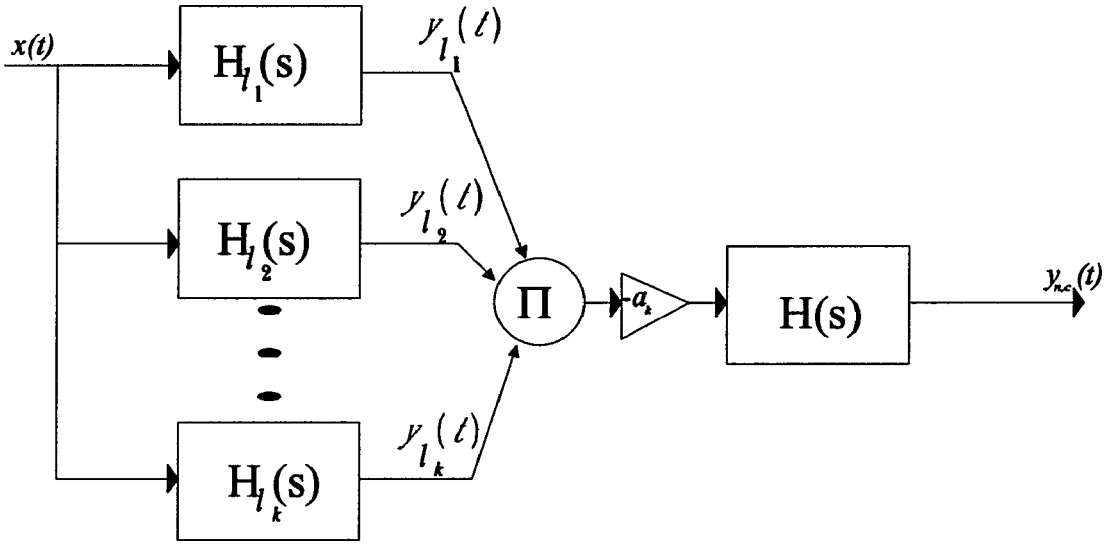


Figure 1: An n^{th}-order Volterra Series Component Response Realization

Due to the constraints on k and the l_i, it is clear that each lower-order response which contributes to the product operation must be of order $n-k+1$ or less. Therefore, a "serial realization" of the Volterra series response may be

[2] This refers to the power series expansion of the nonlinear element characteristic, equation (2-44a).

synthesized recursively in analogous fashion to the determination of the Volterra kernels. Each input term, of second or higher order, to the product operator may be replaced by its serial realization until all of the operators are reduced to linear filters, product operators, or linear gain elements. The advantage of this serial implementation is that only linear filters are required - in fact, only one linear filter type is required.

5.2 An Example of the Serial Realization

The simplicity of this approach may be somewhat obscured as the number of terms increases; however, it is instructive to consider the detailed implementation for a low order system (i.e. truncated at third order, for the purpose of this example). This will illustrate the process which may be followed in constructing a serial realization of the Volterra series for any finite order.

Figure 2 shows the construction of the first order (linear) response. It consists of the linear filter, $H(s)$, cascaded with the constant gain element, a_1.

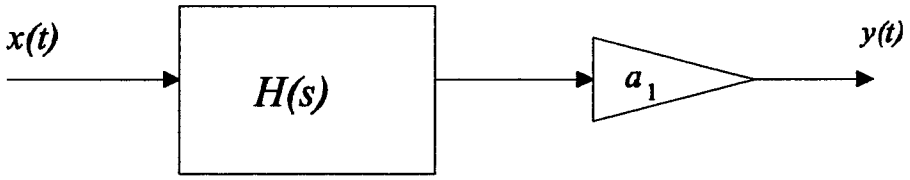


Figure 2: Linear Response Component of a Volterra Series

Figure 3 shows how the second-order response may be synthesized. As indicated by equation (2-47), the second-order response consists of the square of the first-order response, a constant gain stage, $-a_2$, and filtering by the linear filter, $H(s)$. Although the realization of the first-order response is indicated twice in the Figure (analogous to the approach taken in equation (2-52)), this is redundant and there is no need to do so in practice. The first-order response may simply be squared instead.

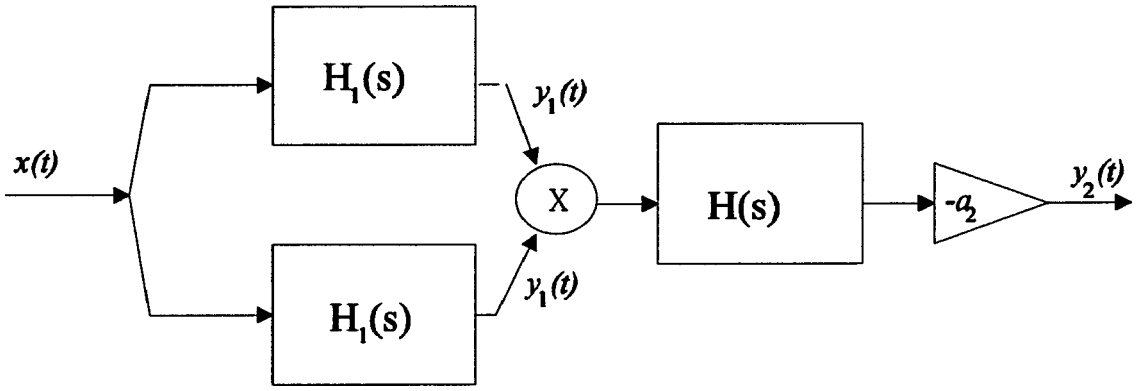


Figure 3: Second Order Response Component Realization

A combined implementation of the first- and second-order responses is shown in Figure 4. Since the linear response is a constituent of the second order response, the combined second-order system realization requires less computation than the aggregation of Figures 2 and 3. In fact, only a summation operation needs to be added to the simplification of Figure 3 which was previously discussed.

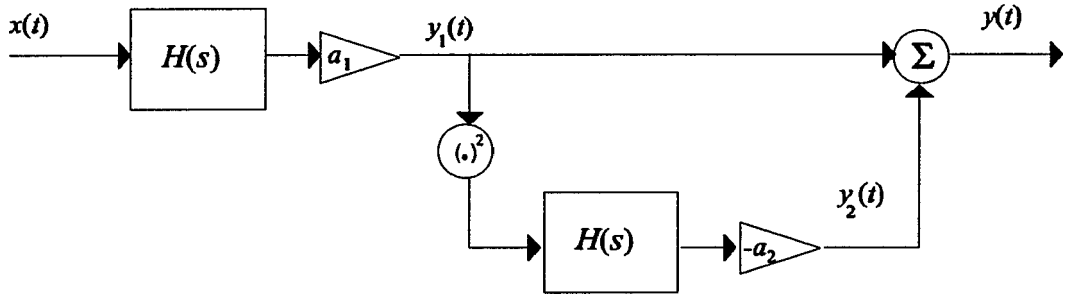


Figure 4: A Combined Second-Order Response Realization

In Figure 4, the redundant first order operation has been removed, and the multiplier replaced with a squarer. The sequential nature of the serial realization is evident for the second-order system. Only two linear filters are required; however, the second filter requires, as its input, a processed (squared and amplified) version of the first filter response. Therefore, it can be seen that the signal dispersion characteristic of the second order response is effectively double that which is introduced by the first order response.

5.3 *A Third-Order Serial Realization*

Continuing with the third order response, the third order term has two components, shown in Figures 3a and 3b. The first component is constructed from the cube of the first order response, while the second component is formed from the product of the first and second order responses. Therefore, the composite third order response requires a succession of three linear filters.

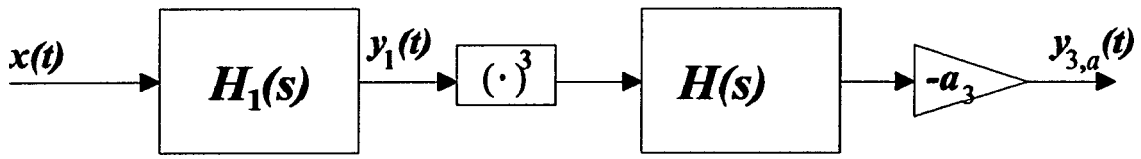


Figure 5a: First Component of the Third-Order Response

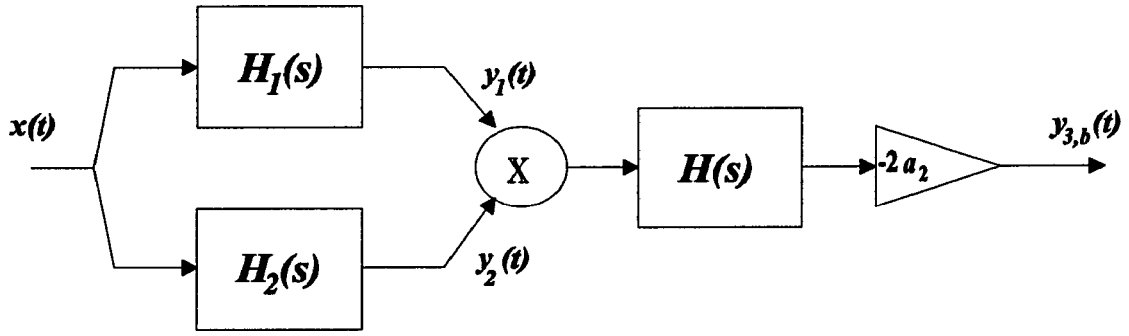


Figure 5b: Second Component of the Third-Order Response

When the two components of the third order response are combined, they may be summed before the final filter, $H(s)$. In order to do this, it is necessary to place the constant gain operators ahead of the linear filter as different gain values are applied to the two components. A complete third order system realization is shown in Figure 6. Only three linear filters are required, but they must be connected sequentially. Accordingly, the signal dispersion of the third order response is effectively three times that of the linear response component. In addition to the filters, three two-input multipliers (including the squarer), three constant gain multipliers, and two summers are required.

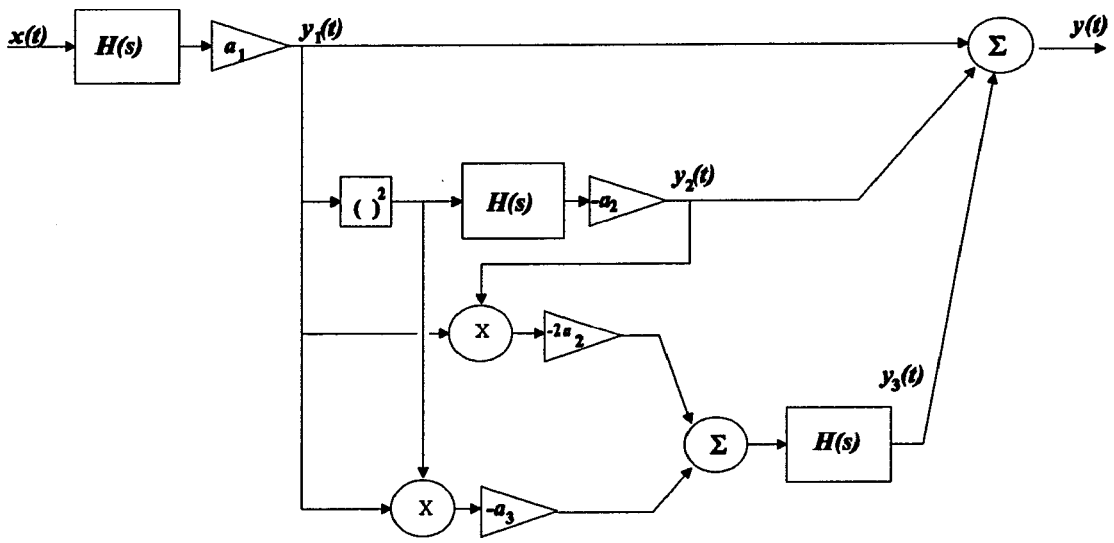


Figure 6: Combined Third-Order Response Serial Realization

5.4 *Generalization of the Serial Realization to N^{th} Order*

It is straightforward to extend the serial realization approach discussed above for second- and third-order systems to an N^{th}-order system. Each successively higher-order term is synthesized in exactly the same manner as the corresponding Volterra series term was realized in section 2.5.2. As only linear filters are required to obtain the response, multidimensional convolution is unnecessary. However, the bandwidth expansion of the higher-order terms is still present and must be accomodated in the discrete-time representation of the system.

We observe that the synthesis of Volterra systems suggested by Shanmugam and Lal [3] is similar to several of the component responses which have been illustrated in the

preceding sections. The premise of Shanmugam and Lal was that *if* the nonlinear transfer functions of a Volterra series admitted to a particular form of factorization, *then* the realization of that term of the Volterra series response could be conveniently obtained as a linear filter followed by a memoryless power-law nonlinearity followed by a second linear filter as shown in Figure 7. This is substantially similar to components of the nonlinear response which have been given above, i.e., the second order-response and the first component of the third-order response. What we have shown, however, is that for systems which can be represented by equations (2-2a) and (2-2b), the corresponding Volterra series and their associated nonlinear transfer functions have a well-defined form which does not coincide precisely with the form expected by Shanmugam and Lal.

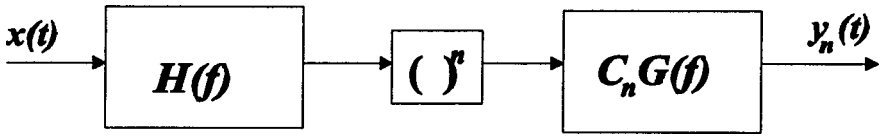


Figure 7: Shanmugam and Lal's Realization of the n^{th}-order Volterra Series Term

The general form of the n^{th}-order nonlinear transfer function for which Shanmugam and Lal [3] presented their system realizations is:

$$H_n(f_1,\cdots,f_n) = C_n\left[\prod_{i=1}^{n} H(f_i)\right] G\left(\sum_{i=1}^{n} f_i\right) \tag{2}$$

If we interpret the coefficient C_n and linear filter $G(f)$ as:

$$C_n = -a_n \quad \text{and} \quad G(f) = H(f),$$

then we obtain, identically the first component of each of the nonlinear transfer functions. Since there is only one component to the second-order response, the complete second-order Volterra series term is obtained.

We have also shown, however, that there are components of the most general form of the response which do not fit the form for which Shanmugam and Lal presented their realization. In particular, the second component of the third-order response cannot be so obtained. More generally, any nth-order response component for which the representation as in equation (1) contains less than n terms in the product of lower-order responses (i.e. any $l_i > 1$) cannot be represented in the form of equation (2). Therefore, except in instances where particular elements of the complete response are degenerate or contribute

insignificantly to the overall response, the system may only be partially realized in the Shanmugam and Lal form.

5.5 *A Computational Estimate for the Serial Realization*

With reference to the notation established in section 4.1, we may evaluate the computational burden associated with implementing a discrete-time serial realization of the Volterra series. Considering, however, the necessity of evaluating the $k-1$ lower-order terms of the Volterra series in order to compute the k^{th}-order response term by the serial realization, we determine the computational estimates jointly for the complete k^{th}-order response rather than separately as was done previously.

To compute a discrete-time approximation to the second-order response as shown in Figure 4, we require two linear filters, two linear gain elements and one squarer (essentially a two-input multiplier with identical inputs).

Although the response of the first linear filter is necessarily bandlimited to the bandwidth of the input, it would be impractical to sample and process the input at the Nyquist rate for the input. This is due to the necessity of deriving the samples for the interpolated instants required to represent the doubled bandwidth present at the output of the squarer. Therefore, we assume that the input process is sampled at a rate consistent with the bandwidth

of the system response and that the linear filters used in a k^{th}-order system have a computational length, $L_k = kN$, where N is the number of filter coefficients required at the Nyquist rate of the input signal.

For a second-order system realization, the computational effort is dominated by the two linear filters ($F_2 = 2$), each having a length, $L_2 = 2N$, and operated at a computational rate, $R_2 = 4W$. Since each filter is linear, the number of multiplications per filter coefficient is $M = 1$. Thus the computational burden for the second order response is:

$$C_2 = R_2(F_2 L_2 + G_2)M_2 = 4W[2(2N) + 3](1) \qquad (3)$$

where G_2 represents the number of isolated multiplications. Assuming that $N >> G_2$, equation (3) reduces to:

$$C_2 \approx 16WN \qquad (3a)$$

which is a factor of $2N$ less than indicated by equation (4-6) for a full-bandwidth direct realization of the second-order Volterra series component and a factor of $N/4$ less than indicated by equation (4-24) for a bandlimited second-order Volterra series component realization.

For a third-order system serial realization, the computational savings is even more dramatic. With reference to Figure 6, it can be seen that the third-order response requires three linear filters ($F_3 = 3$) and seven auxilliary multipliers ($G_3 = 7$), including linear gain elements and signal product multiplications. The processing rate required is three times the Nyquist rate for the input process and the computational length of each linear filter is also three times the reference value, N ($R_3 = 6W, L_3 = 3N$). The computational effort required to obtain the complete third-order response by the discrete-time serial realization is thus:

$$C_3 = R_3(F_3L_3 + G_3)M_3 = (6W)[3(3N) + 7](1) \tag{4}$$

Again, neglecting the auxilliary multiplications, we have that the third-order, discete-time serial realization computational effort is approximately:

$$C_3 \approx 54WN \tag{4a}$$

Equation (4a) is readily generalized. For a k^{th}-order discrete-time serial realization, the computational effort is (neglecting the auxilliary multiplications):

$$C_k = 2k^3WN \tag{5}$$

Comparing this expression to that which was obtained for the computational burden of the nth-order term of a bandlimited Volterra series in Chapter 4:

$$C_n = \frac{1}{n!}(2W)nN^n \tag{6}$$

Thus, while the serial realization necessitates processing samples at the higher Nyquist rate of the response, the computational savings of n 1-dimensional filters instead of one n-dimensional filter becomes the dominant factor in determining the more computationally efficient approach for all but the shortest filter lengths. For systems up to seventh order, comparison of equations (5) and (6) reveals that unless the truncated length of the linear filter is less than 8 samples less computational effort will be required to implement the system as a serial realization. However, the representation of the Volterra series in the serial form makes it clear that extremely short response duration truncation of the direct (or bandlimited) Volterra series kernels is likely to result in significant response error contributions.

CHAPTER 6

Computation of a Nonlinear System Response by Picard Iteration

In addition to illustrating a procedure for obtaining the Volterra series kernels for a nonlinear system from the Volterra integral equation for that system, Leon and Schaefer [1] presented the development of an iterative method (Picard iteration) for solving the integral equation (see equations (2-2b) and (2-2c)):

$$y(t) = \int_{-\infty}^{t} g(t-\tau)x(\tau)d\tau - \int_{-\infty}^{t} h(t-\tau)f[y(\tau)]d\tau \qquad (1)$$

when $f(z)$ has a polynomial representation or approximation of the form:

$$f(z) = \sum_{i=2}^{m} a_i z^i \qquad (1a)$$

The iteration sequence for equation (1) is:

$$y_{p,1}(t) = \int_{-\infty}^{t} g(t-\tau)x(\tau)d\tau \qquad (2a)$$

$$y_{p,2}(t) = \int_{-\infty}^{t} g(t-\tau)x(\tau)d\tau - \int_{-\infty}^{t} h(t-\tau)f[y_{p,1}(\tau)]d\tau \qquad (2b)$$

$$y_{p,k}(t) = \int_{-\infty}^{t} g(t-\tau)x(\tau)d\tau - \int_{-\infty}^{t} h(t-\tau)f[y_{p,k-1}(\tau)]d\tau \qquad (2c)$$

The double subscript, p,k, indicates the k^{th} Picard iterate so as to avoid any confusion with Volterra series terms.

Leon and Schaefer [1] show that the k^{th} Picard iterate includes the first k Volterra series terms and some of the information contained in components of Volterra series terms having orders $k+1$ and greater. This makes Picard iteration potentially a very powerful tool. Furthermore, the iteration procedure may be discretized very efficiently.

6.1 A Computational Structure for Picard Iteration

Figure 1 shows how the successive Picard iterates may be computed. Recognize, however, that the iteration procedure outlined by equations (2a) through (2c) implies that the k^{th} iterate is computed for all time before the $k+1^{st}$ iterate is computed. Therefore, the procedure is not immediately adaptable to real-time signal processing. However, recognition of the implications of causality will permit sample-by-sample computation of the response.

Figure 2 shows a concatenation of multiple basic iteration blocks with several simplifications incorporated. First, the calculation of the linear portion of the response is common to all iterates; there is no need to repeat it. Second, the first iteration has no 0^{th} iterate

result (i.e., a component on the lower path) as an input. Alternatively, we may consider that input to be identically

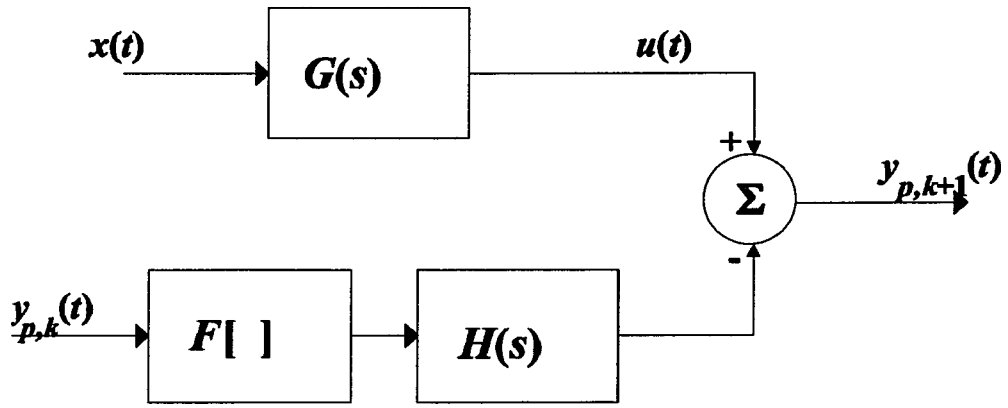


Figure 1: A Single Element for Computation of the Picard Iteration Approximation to the Volterra Integral Equation

zero. The result is a ladder structure which has no computations performed along the top path past the first iteration.

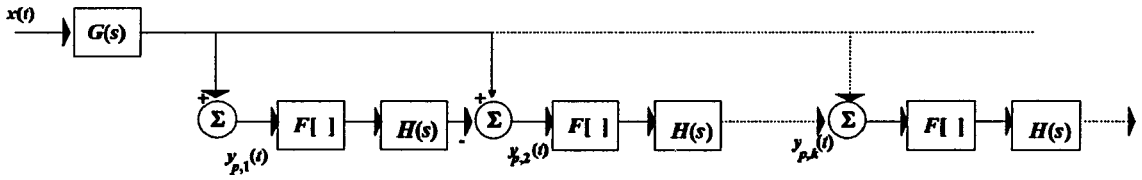


Figure 2: Basic Iteration Blocks Cascaded to Create an Iteration Ladder Structure

6.2 Contraint on Bandwidth Expansion

In anticipation of discretizing the structure in Figure 2, we observe that since the linear filter, $H(s)$, follows the element nonlinearity, $F[\,]$, the response bandwidth at the input to each summer is restricted by the essential bandwidth of the filter. Furthermore, at the input to each filter, the bandwidth expansion is limited by the degree, m, of the polynomial, $f(z)$ (see equation (1a)), which represents the nonlinear element in the system. Therefore, the maximum sampling rate required for a discrete-time processor based on the Picard iteration method of Leon and Schaefer is restricted to:

$$R_p = 2Bm \tag{3}$$

where B is the essential bandwidth of the linear filter characteristic associated with the system. This may offer a particularly efficient approach to discrete-time processing for cases where the input signal bandwidth is greater than the bandwidth of the system.

6.3 Constraints Due to Causality

If we assume that the system is causal and that the input is identically zero for $t < t_0$ with the system initially at rest, then equation (1) becomes:

$$y(t) = \int_{t_0}^{t} g(t-\tau)x(\tau)d\tau - \int_{t_0}^{t} h(t-\tau)f[y(\tau)]\,d\tau \qquad (4)$$

Therefore, for two arbitrary instants of time, t_1 and t_2, we may write:

$$y(t_1) = \int_{t_0}^{t_1} g(t_1-\tau)x(\tau)d\tau - \int_{t_0}^{t_1} h(t_1-\tau)f[y(\tau)]\,d\tau \qquad (5a)$$

$$y(t_2) = \int_{t_0}^{t_2} g(t_2-\tau)x(\tau)d\tau - \int_{t_0}^{t_2} h(t_2-\tau)f[y(\tau)]d\tau \qquad (5b)$$

Choosing t_1, t_2, such that $t_2 > t_1 > t_0$ and taking the difference of equations (5b) and (5a) gives the result:

$$y(t_2)-y(t_1) = \int_{t_0}^{t_2} g(t_2-\tau)x(\tau)d\tau - \int_{t_0}^{t_1} g(t_1-\tau)x(\tau)d\tau - \int_{t_0}^{t_2} h(t_2-\tau)f[y(\tau)]d\tau \qquad (6)$$

$$+\int_{t_0}^{t_1} h(t_1-\tau)f[y(\tau)]d\tau$$

Moving the $y(t_1)$ term to the right hand side and splitting the third integral yields:

$$y(t_2) = y(t_1) + \int_{t_0}^{t_2} g(t_2-\tau)x(\tau)d\tau - \int_{t_0}^{t_1} g(t_1-\tau)x(\tau)d\tau - \int_{t_0}^{t_1} h(t_2-\tau)f[y(\tau)]d\tau \qquad (7)$$

$$-\int_{t_1}^{t_2} h(t_2-\tau)f[y(\tau)]d\tau + \int_{t_0}^{t_1} h(t_1-\tau)f[y(\tau)]d\tau$$

Finally, combining the third and fifth integrals gives:

$$y(t_2) = y(t_1) + \int_{t_0}^{t_2} g(t_2-\tau)x(\tau)d\tau - \int_{t_0}^{t_1} g(t_1-\tau)x(\tau)d\tau - \int_{t_0}^{t_1} [h(t_2-\tau)-h(t_1-\tau)]f[y(\tau)]d\tau$$
$$- \int_{t_1}^{t_2} h(t_2-\tau)f[y(\tau)]d\tau \qquad (8)$$

Assume that $y(t)$ is known for $t \leq t_1$. This would be the case when, for example, equation (5a) had been solved by Picard iteration. Then, only the last term on the right hand side of equation (8) needs to be established by iteration. The first term is, by assumption, known, the first two integrals depend only on the input, and the third integral can be solved explicitly from the knowledge of $y(t)$ over the limits of integration.

6.4 Discretization of the Revised Volterra Integral Equation

Assume that the input signal, $x(t)$, is W-bandlimited and that the nonlinear element characteristic, $f(z)$, has a polynomial approximation in the form of equation (1a) with degree m. Further assume that the associated linear transfer function $H(f)$ is insignificant for $|f| > B$; i.e.,

the essential bandwidth of the filter is B[1]. Then we may construct a discrete-time realization of the system based on a Picard iteration solution of the Volterra integral equation.

Let us choose the system bandwidth, B, to be related to the input bandwidth W as $B = lW$, where l is an integer. Then the required sampling rate for unaliased discrete-time processing is:

$$R = 2lmW \tag{9}$$

Further, let us choose the difference $t_2 - t_1$ such that:

$$t_2 - t_1 = (2lmW)^{-1} = \Delta t \tag{10}$$

Then, for time values delayed longer than the filter length after the beginning of the input signal, i.e., for $n\Delta t > t_0 + N\Delta t$, equation (8) can be represented approximately as the discrete-time convolution summation expression:

$$y(n\Delta t) = y((n-1)\Delta t) + \Delta t\, a_1 \sum_{k=n-N+1}^{n} h((n-k)\Delta t)x(k\Delta t) - \Delta t\, a_1 \sum_{k=n-N+1}^{n} h((n-k)\Delta t)x((k-1)\Delta t$$

(11)

$$-\Delta t \sum_{k=n-N+1}^{n} [h((n-k+1)\Delta t) - h((n-k)\Delta t)]\left\{\sum_{i=2}^{m} a_i[y((k-1)\Delta t)]^i\right\} - h(0\Delta t)\sum_{i=2}^{m} a_i[y(k\Delta t)]^i$$

---

[1] We assume for convenience that $B > W$. If this is not satisfied, then set $B = W$.

where we have written $g(t)$ as $a_1 h(t)$ in accordance with the results obtained in section 2.5 and an N-term FIR realization of the system's associated linear response has been assumed, i.e., $h(n\Delta t)$ is non-zero only for $0 < n < N\text{-}1$.

Equation (11) indicates that the iteration need be applied only to a single-term summation. We may simplify the appearance of equation (11) and make the relationship more evident by defining the following:

$$u(n\Delta t) = \Delta t\, a_1 \sum_{k=n-N+1}^{n} h((n-k)\Delta t)x(k\Delta t) \tag{12}$$

$$h_\Delta(n\Delta t) = h((n+1)\Delta t) - h(n\Delta t), \quad 0 \le n \le N-1 \tag{13}$$

where $h_\Delta((N\text{-}1)\Delta t) = h((N\text{-}1)\Delta t)$ since $h(N\Delta t) = 0$.

Equation (11) may now be rewritten as:

$$y(n\Delta t) = y((n-1)\Delta t) + u(n\Delta t) - u((n-1)\Delta t) - \Delta t \sum_{k=n-N+1}^{n} h_\Delta((n-k)\Delta t)\left\{\sum_{i=2}^{m} a_i[y((k-1)\Delta t)]^i\right\} - h(0\Delta t)\left\{\sum_{i=2}^{m} a_i[y(k\Delta t)]^i\right\} \tag{14}$$

In equation (14) the values $u(n\Delta t)$ and $u((n\text{-}1)\Delta t)$ represent the present and previous values of the output of a linear filter (defined in equation (12)) operating on the system input. The term $y((n\text{-}1)\Delta t)$ is the previous value of the

nonlinear system response. The only term in equation (14) with a summation indicates a linear filter operating on *past* values only of the nonlinear system response. The Picard iteration need then be performed only with regard to the present sample value of the nonlinear system response. Since it involves only a single sample value it may readily be performed using Newton's method.

6.5 *Computational Complexity Estimate for the Picard Iteration Technique*

The computational effort required to obtain a nonlinear system response using the Picard iteration technique is less sensitive to the order of the nonlinearity than the techniques previously discussed. When a solution is mechanized using equation (14) with an appropriate iteration technique, the computational effort depends more on the essential bandwidth of the associated linear transfer function and the degree of the nonlinear element's polynomial approximation than it does on the order of the nonlinear response. When the lengths of the linear filters dominate the iteration computation required to achieve a convergent solution, the effort may become essentially independent of order. However, the necessarily high processing rate, as indicated in equation (9), may make this technique unattractive for applications which do not require high-order response approximations.

For a response approximation which need be accurate only to the m^{th}-order, it is clearly unnecessary to maintain a filter essential bandwidth greater than mW for an input signal bandwidth W. Accordingly, the required sampling rate is $R = 2m^2W$. Let us assume that the essential duration of the associated linear system impulse response is N' samples expressed with respect to the Nyquist rate for the input signal[2]. Then the number of filter coefficients required at the discrete-time processing rate will be $N = m^2N'$. Two linear filters are required as are a number of auxilliary multiplications, A, in the filtering and iteration procedures. The overall computational magnitude is approximately:

$$C_m \approx 2m^2W\left(m^2N' + A\right) \tag{15}$$

If the auxilliary operations can be neglected, equation (15) reduces to:

$$C_m \approx 2Wm^4N' \tag{16}$$

---

[2] We express the filter length in this manner in order to maintain consistency with the computational estimates derived in earlier chapters so that a comparison can be made for the purpose of selecting the most efficient technique for a particular application.

Comparing equation (16) to equation (5-5) for the computational effort of the serial realization, it is apparent that the Picard iteration approach may require a factor of m more multiplication operations than the serial Volterra series realization approach. This result should be applied cautiously, however, as equation (16) is based on the assumption that the essential bandwidth of the system is at least as wide as the m-fold convolution of the input signal spectrum with itself. This assumption may or may not be valid in any particular case.

CHAPTER 7

A Computational Comparison of Discrete-Time Nonlinear System Responses

In order to assess the capabilities of the computational methods described in the foregoing chapters, the various nonlinear system realizations were constructed and exercised using several test signals. These activities demonstrated both the utility of the techniques for predictive analyses of nonlinear systems via simulation and the relative degrees of difficulty associated with obtaining the system realizations[1]. In presenting these results, we first describe the determination of the specific system realizations. The selection and scaling of the test inputs is described next with estimates of the accuracies expected for each nonlinear system realization. This is followed by the presentation of the results obtained for the test inputs by each system realization.

---

[1] The estimates of computational complexity presented in Chapters 4-6 apply to execution of discrete-time simulations with the particular system realizations. The computational burdens associated with obtaining the coefficients for higher-order Volterra kernels are also substantial and have considerable bearing on the usefulness of such techniques in particular applications.

7.1 *Computational Issues in Volterra Filter Coefficient Determination*

Having obtained explicit representations of the continuous-time Volterra kernels in Chapter 2 for the first-, second-, and third-order terms as in equations (2-55), (2-58), and (2-66/2-70), we still must obtain suitable representations for use in discrete-time processing. Since the corresponding nonlinear transfer functions are not bandlimited, the "appropriate" discrete Volterra filter coefficients are very much dependent on the signal(s) to be processed and their frequency extent relative to the parameters of the system.

For the purpose of developing the example based on the circuit presented in Chapter 2, we have utilized the following parameter values:

Capacitance	C	100 pF	(1.0×10^{-10} F)
Resistance	R	12.5 MΩ	(12.5×10^{7} Ω)
Diode Reverse Saturation Current			
	I_s	1 nA	(1.0×10^{-9} A)
Inverse Thermal Voltage Constant			
	λ	40 V^{-1}	

These parameter values result in a reciprocal time constant, k (defined in equation (2-42)), for the circuit of 1200 sec^{-1}.

Additionally, we have chosen to limit the maximum input signal frequency to 1000 Hertz. At the chosen maximum input frequency, the magnitude of the first-order (linear) filter's frequency response is not insignificant:

$$\frac{\left|H\left(1000\right)\right|}{H\left(0\right)} = 0.188$$

Therefore, some form of aliasing control is needed - for the Volterra filters of all orders - if the sampling rate of the simulation is to be minimized consistent with the maximum input frequency.

As illustrated in Appendix A, frequency-domain windowing of the (nonlinear) transfer functions can be utilized to eliminate aliasing in the Volterra kernels. A rectangular window was utilized in our evaluations to restrict the frequency response to the passband of the input signal. The filter coefficients were then obtained by numerical integration of the inverse Fourier transform integrals, i.e.:

$$h_n(k_1\Delta t,\cdots,k_n\Delta t) = \int_{-W}^{W}\cdots\int H_n\left(f_1,\cdots,f_n\right)\exp\left[j2\pi\left(f_1k_1\Delta t+\cdots+f_nk_n\Delta t\right)\right]df_1\cdots df_n \qquad (1)$$

The character of the single-pole filter in the simple diode circuit used in our example precludes establishing a bound on the absolute error of the time-domain response[2]. Therefore, the error induced by truncation of the discrete-time domain Volterra kernels was estimated by the response energy technique which is described in Appendix A.

7.1.1 Selection of a Numerical Integration Technique

In the one-dimensional, jointly bandlimited and duration-limited response outlined in Appendix A, trapezoidal integration was utilized with an associated explicit error bound calculation based on the second derivative of the integrand. While this is an effective method for obtaining the filter coefficients for a linear filter (a single integral approximation), it does not offer an *a priori* means for controlling the coefficient accuracy since the error estimate is a function of the chosen step size[3]. Furthermore, it can be seen from the coefficients in Appendix A and their associated error bounds that the number of trapezoidal regions required to maintain a specified accuracy (either absolute or relative) varies.

In this specific case, i.e., an inverse Fourier transform integral, as the time instant $n\Delta t$ of the

---

[2] The integral in equation (3-24) is divergent for this choice of $H(f)$.

[3] In fact, the error bound calculation requires more operations than the trapezoidal integration.

coefficient (impulse response sample) being calculated increases, the transform kernel $\exp(j2\pi n\Delta t)$ becomes more oscillatory over the region of integration in the frequency domain. As a result, a trapezoidal approximation to the integral using a fixed step size, Δf, becomes less accurate with increasing $n\Delta t$.

An alternative approach is to utilize an extrapolation technique for numerical integration. One such technique is Romberg integration [31]. This technique produces a succession of approximations to the integral for successively decreased trapezoidal widths and extrapolates the series of approximations to zero step width. Upon achieving two successive extrapolation estimates which differ by less than a pre-specified tolerance, convergence of the procedure is declared to an accuracy equal to the value of the specified tolerance[4].

The Romberg integration technique is implemented in the *Mathcad* software package [32] which was used to obtain the coefficient results reported in this Chapter. The particular implementation of Romberg integration which is utilized by *Mathcad* carries the procedure at least one additional step; two successive extrapolation estimate differences less than the tolerance are required before

---

[4] Typically, the tolerance is specified relative to the magnitude of the most current extrapolation estimate.

results are reported. Thus, the results obtained are significantly more accurate than indicated by the tolerance value.

7.1.2 Comments on the Discrete Fourier Transform as a Numerical Integration Technique

The (inverse) discrete Fourier transform (DFT) may be viewed as a numerical integration (see Appendix B). Therfore, efficient techniques - the class of fast Fourier transform (FFT) algorithms - for calculating the (inverse) DFT may be considered as alternatives for computing the coefficients of Volterra filters. These techniques have not been used here for two reasons.

First, the DFT does not address the accuracy of the numerical integration. As discussed in section 7.1.1, the accuracy of trapezoidal integration improves with decreasing trapezoid width; however, a bound on the accuracy requires a further computation based on the second derivative of the integrand. The DFT does not provide any such assessment, nor is it apparent that such an error bound calculation could be streamlined in any manner similar to the DFT itself. The indications of accuracy obtained in Appendix A for the tapered window example suggest that the number of trapezoidal regions - N in the DFT - may need to be very large to preserve the accuracy of some coefficients. An FFT approach would require that the

largest value of N needed for any coefficient be used in determining all coefficients.

Second, the number of coefficients computed by a block FFT routine is equal to the number of frequency increments (trapezoids) used in the numerical integration. This may be substantially more than the number of coefficients required by the truncation error constraints of a particular system evaluation. Consequently, the expected efficiencies of the FFT algorithms are not likely to be realized when computing filter coefficients.

Because the accuracy of an FFT-based approach to computing the Volterra coefficients is uncertain and the computational efficiency is not expected to offer a significant advantage, the Romberg integration method described above was used to calculate all of the coefficients in this Chapter.

7.1.3 Frequency Domain Window

While the frequency domain windowing which was illustrated in Appendix A may be beneficial in reducing the Gibbs' phenomenon at transition points, it substantially adds to the complexity of obtaining the Volterra filter coefficients in two ways. As a result, a rectangular frequency-domain window was selected for computing the Volterra filter coefficients reported in this Chapter.

The first effect of frequency-domain windowing on coefficient computation is that it increases the density of sampling points required in the time-domain discrete Volterra kernels. If the tapering function, $T(f)$, is applied for $A \leq |f| \leq B$ such that $B = (1+\varepsilon)A$ where A is the maximum input frequency, then the n^{th}-order kernel obtained in this manner will require $(1+\varepsilon)^n$ times as many coefficients as one obtained for a rectangular truncation at $|f| = A$. For instance, in Appendix A, the values of A and B used are $A = 2000$ and $B = 3500$, so that $\varepsilon = 0.75$. Accordingly, this will require approximately $(1.00 + 0.75)^3 = 5.36$ times as many coefficients to represent the third-order kernel as one determined from a rectangular windowed nonlinear transfer function. Given the magnitude of the computational burden required to obtain these coefficients, as well as the corresponding large increase in the discrete-time processing burden, this is a very undesirable increase in effort. In fact, the tapering necessarily extends into the regions which we want to eliminate in order to construct the bandlimited Volterra kernel as described in Chapter 4.

The second source of complexity is the need for separate integrations over the different tapered regions.

In the one-dimensional case, only one additional integration region (due to symmetry) was required, $A \leq |f| \leq B$. In the case of the second-order Volterra kernel, three additonal regions would be required, assuming that the same exploitation of symmetry is obtained: taper in f_1, taper in f_2, and taper in f_1+f_2. The third order Volterra kernel coefficient calculations would require the primary integration plus seven tapered-region integrations. This represents a highly objectionable increase in the computational load.

7.1.4 Coefficient Indexing

Since the result of discrete-time processing with a Volterra series is not affected by the symmetry (or lack thereof) of the kernels obtained for a system, it is advantageous to obtain the unique symmetric kernels, as this reduces the computational load in executing a discrete-time simulation. (There is no significant difference in the computation required to obtain the coefficients of a symmetric kernel as opposed to the asymmetric coefficients.) For the higher-order kernels, it is important to use a structured method for indexing the coefficients to preclude any confusion or double-counting.

For the example system, we have chosen a third-order system realization. Therefore, we will illustrate the

indexing scheme for the third-order Volterra kernel; it may be extended to any order in a straightforward manner (the second-order scheme is even simpler).

Let the third-order, discrete-time symmetric Volterra kernel sample at $t_1 = l\Delta t$, $t_2 = m\Delta t$, $t_3 = n\Delta t$ be written as $h_{3,d}(l, m, n)$. Then for distinct[5] $l, m,$ and n we have:

$$\begin{aligned} h_{3,d}(l, m, n) &= h_{3,d}(m, l, n) \\ &= h_{3,d}(l, n, m) \\ &= h_{3,d}(n, l, m) \\ &= h_{3,d}(m, n, l) \\ &= h_{3,d}(n, m, l) \end{aligned}$$

Since we have assumed a symmetric kernel, only one of these coefficients need be determined. Therefore, let us determine the set of coefficients $h_{3,d}(l, m, n)$ for indices $l, m,$ and n such that $l \le m \le n$. Consequently, if we choose to determine the coefficients of the third-order kernel for $0 \le l, m, n \le 9$, there will be 1000 coefficients within the index bounds so defined; however, only 220 of the coefficients will be unique. Our indexing technique effectively exploits the six-fold symmetry of the

---

[5] When l, m, and n are not all distinct, some of the coefficient index sets will be indistinguishable.

third-order kernel; the reason that the number of coefficients required exceeds one-sixth of the total number of coefficients is that no symmetry reductions can be obtained when $l = m = n$ and only three fold symmetry exists when two of the three indices have the same value.

7.2 Determination of the Volterra Filter Coefficients

Our evaluations considered four discrete-time realizations of the example system presented in Chapter 2. These realizations are:

- Direct realization of the Volterra series (not bandlimited in the sense described in Chapter 4, but truncated to the input signal bandwidth in order to control the effect of filter response aliasing)
- Bandlimited Volterra series (Chapter 4)
- Serial Realization of the Volterra series (Chapter 5)
- Picard Iteration realization (Chapter 6)

In order to construct the four discrete-time realizations of our example system, we require the following Volterra kernels, based on the assumption of an input signal which is bandlimited to 1000 Hertz:

- First-order kernel, bandwidth truncated to 1000 Hertz, sampled at $T = 1/2000$ second; to be used as the first-order kernel of the Bandlimited Volterra Series
- First-order kernel, bandwidth truncated to 1000 Hertz, sampled at $T = 1/6000$ second; to be used as the first-order kernel of the Direct Realization of the Volterra series
- First-order Volterra integral kernel, i.e. $h(n\Delta t)$ which differs from $h_1(n\Delta t)$ by the factor a_1, bandwidth truncated to 3000 Hertz, sampled at $T = 1/6000$ second; to be used as the first-order filter module for the Serial Realization of the Volterra Series
- First-order Volterra integral kernel, i.e. $h(n\Delta t)$ which differs from $h_1(n\Delta t)$ by the factor a_1, bandwidth truncated to 9000 Hertz, sampled at $T = 1/18000$ second; to be used as the first-order filter module for the Picard Iteration Realization
- Second-order kernel, bandwidth truncated (in f_1 and f_2) to 1000 Hertz, bandlimited to 1000 Hertz in f_1+f_2, and sampled at $T = 1/2000$ second; to be used as the second-order kernel of the Bandlimited Volterra Series
- Second-order kernel, bandwidth truncated (in f_1 and f_2) to 1000 Hertz, sampled at $T = 1/6000$ second; to be used as the

second-order kernel of the Direct Realization of the Volterra Series

- Third-order kernel, bandwidth truncated (in f_1, f_2, and f_3) to 1000 Hertz, bandlimited to 1000 Hertz in $f_1+f_2+f_3$, and sampled at $T = 1/2000$ second; to be used as the third-order kernel of the Bandlimited Volterra Series
- Third-order kernel, bandwidth truncated (in f_1, f_2, and f_3) to 1000 Hertz, sampled at $T = 1/6000$ second; to be used as the third-order kernel of the Direct Realization of the Volterra Series

Each of the required discrete-time Volterra filters, i.e. the kernels, was constructed using the Romberg (numerical) integration technique to evaluate equation (1) for the appropriate (non)linear transfer function and transform kernel to obtain a sufficient number of samples (coefficients) to yield a sufficiently accurate representation.

7.2.1 First-Order (Linear) Kernel Determination

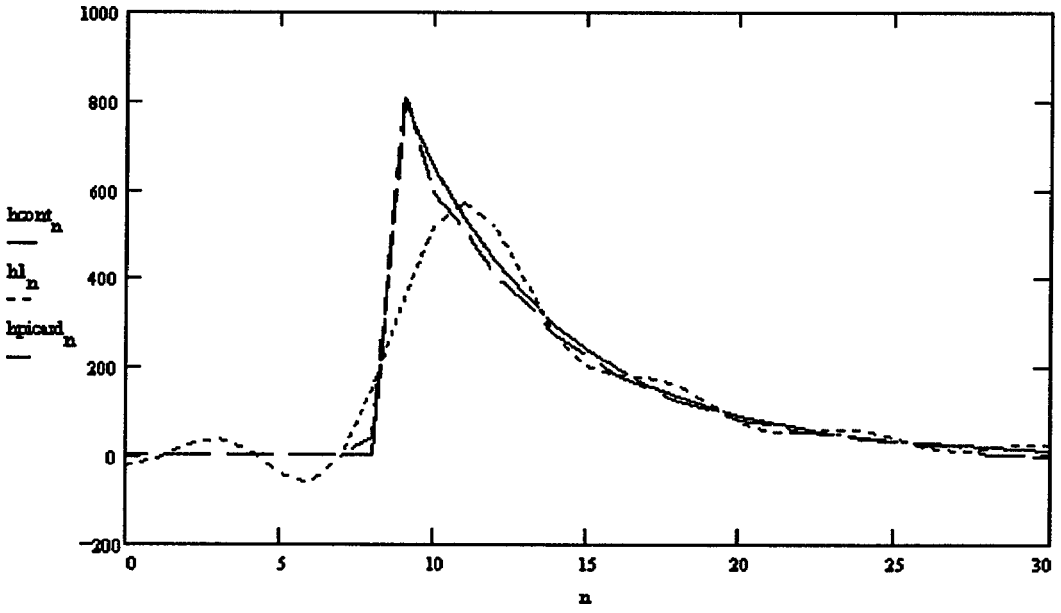
The four sets of coefficients for the first-order kernel realizations are obtained by numerical solutions of the integral:

$$h_{1,d}(n\Delta t) = \frac{1}{RC}\int_{-W}^{W} \frac{1}{k+j2\pi f} \exp\left(j2\pi f n\Delta t\right) df \qquad (2)$$

for values $\Delta t = 1/18000,\ 1/6000,\ 1/2000$ and $W = 1000,\ 3000,\ 9000$ as appropriate. The index parameter, n, defines the individual coefficients of the filters. The number of coefficients required for an acceptable representation is determined by the fraction of the response energy contained by the selected set of coefficients.

The coefficients for the filters which are bandwidth-truncated to 1000 Hertz are identical when the sample-time instants agree, i.e. the first filter coefficient set is a subset of the second. The actual coefficient sets are given in Appendix C. It is more intuitively appealing to examine a graph of the coefficient values compared to the values of the continuous-time response at the same time instants. This is shown in Figure 7-1.

It is clear from the Figure that as the truncation bandwidth is expanded, the approximation to the continuous-time prototype filter characteristic becomes quite good. The most significant difference between the two discrete-time filter characteristics is the slower rise-time of the more severely-bandlimited characteristic.



Solid trace (hcont) represents the continuous-time prototype linear filter response

Dotted trace (h1) represents the first-order Volterra kernel response

Dashed trace (hpicard) represents the filter characteristic used to compute the Picard iteration response (scaled by coefficient a_1 to allow comparison)

Figure 7-1: First-order (linear) filter realizations

The sufficiency of the number of coefficients which are included in each of the discrete-time realizations was assessed by applying the response energy comparison which was developed in Appendix A. By computing the energy of the response from the bandwidth truncated-transfer function, the proportion of the energy which is captured by

a set of N coefficients can readily be determined. For the first-order kernel based on a 1000 Hertz truncation of the transfer function and Nyquist sampling (to be used in the Bandlimited Volterra series realization), the 13 samples identified in Appendix C capture 99.5% of the total energy (the truncation energy error is 23.1 dB down). The 31 samples of the kernel based on 1000 Hertz truncation but sampled at 3 times the Nyquist rate (for use in the Direct Volterra series realization) yield 99.7% of the response energy (error 25.1 dB down). For the kernel based on a 3000 Hertz truncation of the transfer function and Nyquist sampling for the truncation frequency (for the Serial realization), the 31 samples capture slightly greater than 99.9% of the response energy, leaving the response energy error 30.9 dB below the truncated response energy. The 60 samples of the kernel truncated to 9000 Hertz, sampled at Δt = 1/18000 second, include 99.9% of the response energy leaving the error 30.3 dB down. Each of the first-order filter realizations is non-causal; the transfer function truncation in the frequency domain produces an anticipatory time response. Since slightly greater than 0.5% of the response energy occurs for $t<0$, a non-causal realization is necessary to obtain an error greater than 22 dB down.

7.2.2 Second-Order Volterra Kernel Determination

The second-order Volterra kernel coefficients for an input which is bandlimited to 1000 Hertz are found as the solution to the inverse Fourier transform integral:

$$h_2(k_1\Delta t, k_2\Delta t) = \int_{-1000}^{1000}\int_{-1000}^{1000} H_2\left(f_1, f_2\right)\exp\left[j2\pi\left(f_1 k_1\Delta t + f_2 k_2\Delta t\right)\right]df_1 df_2 \qquad (3)$$

where, as determined in Chapter 2:

$$H_2\left(f_1, f_2\right) = a_2 H_1\left(f_1\right)H_1\left(f_2\right)H\left(f_1 + f_2\right) \qquad (4)$$

The set of coefficients is indexed by the parameters k_1 and k_2.

The coefficients for the second-order filters are determinable in a reasonable time, although with substantially more effort than the first-order filter coefficients. Two cases were evaluated.

First, the complete second-order Volterra kernel was determined for a third-order system realization. That is, for a third-order system with a 1000 Hertz input, the maximum response frequency which can occur is 3000 Hertz. Accordingly, the necessary coefficient interval to form a part of the third-order system is 1/6000 second. The resulting filter characteristic which was obtained in our

evaluation is shown in Figure 7-2. The set includes 225 coefficients which collectively represent 92.4% of the response energy; correspondingly, the truncation error is approximately 11.2 dB down.

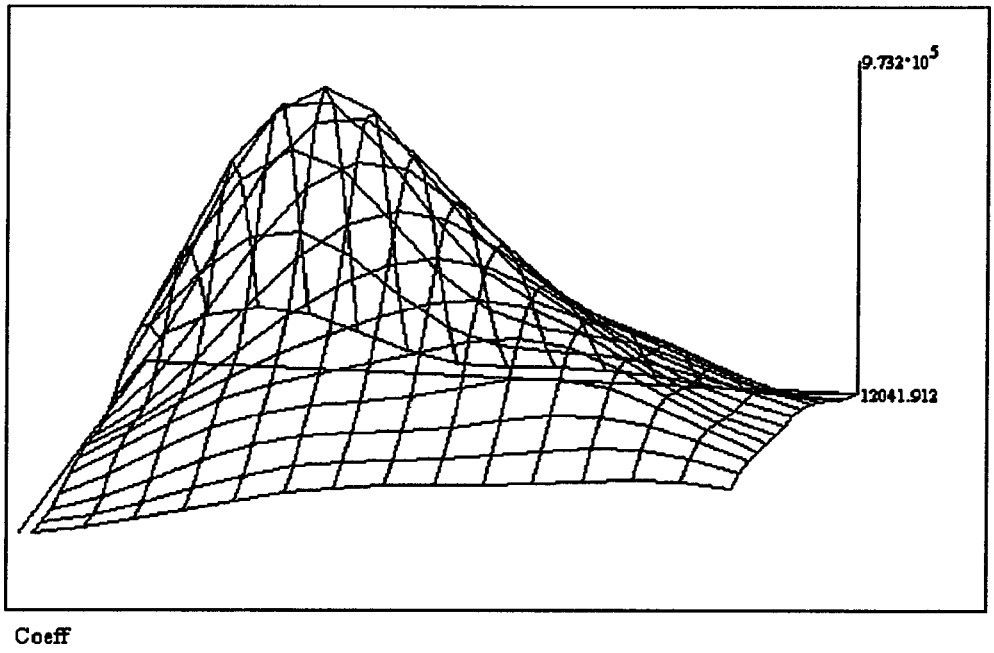


Figure 7-2: The second-order Volterra filter

The surface contour in the Figure shows the amplitude of the second-order Volterra filter at the two-dimensional discrete-time instant in the base plane. The increments for which the amplitudes are given (as indicated by the intersections of the contour lines) are the indices (m, n) for which the Volterra filter coefficients, $h_2(m\Delta t, n\Delta t)$, have

been computed. The value of Δt is 1/6000 second. The origin (i.e. $m, n = 0$) is at the far (into the page) left side of the plot. The peak value shown on the vertical scale bar is the value of the largest coefficient, $h_2(4\Delta t, 4\Delta t)$.

The second representation of the second-order Volterra filter is for the Bandlimited realization which was defined in Chapter 4. The second-order nonlinear transfer function for this filter requires the two-dimensional mask function to be applied to the transfer function given in equation (4). The resulting bandlimited transfer function is:

$$H_{2,BL}(f_1, f_2) = a_2 H_1(f_1) H_1(f_2) H(f_1 + f_2) M_2(f_1, f_2) \qquad (5)$$

where the mask function is defined in equation (4-18). The second-order bandlimited Volterra filter thus obtained is shown in Figure 7-3. In our evaluation, we obtained a 7 by 7 coefficient matrix. The Figure may be interpreted in precisely the same way as Figure 7-2 with the value of Δt set to 1/2000 second.

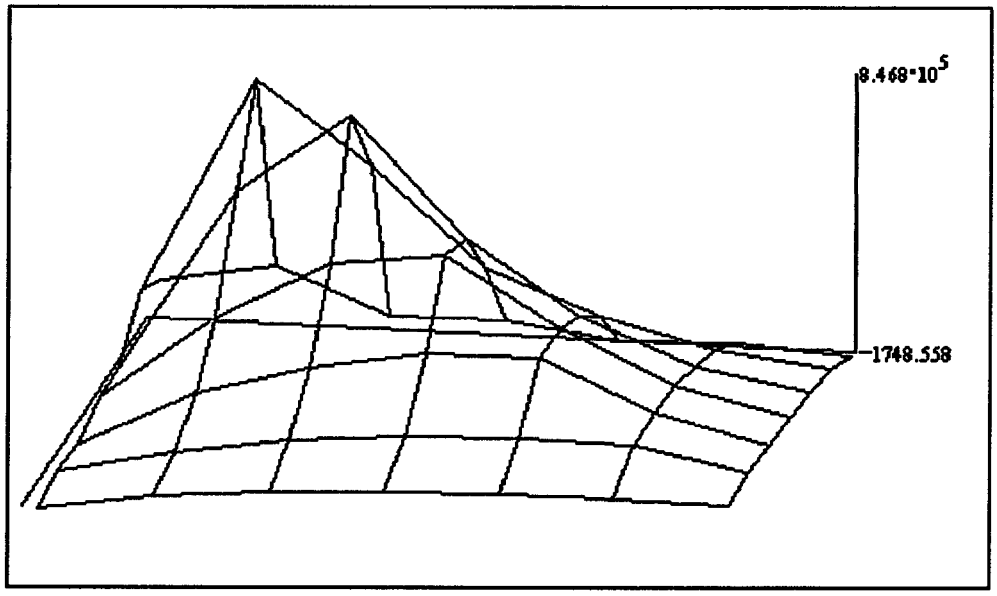


Figure 7-3: Second-order Bandlimited Volterra kernel

The coefficient values obtained for the bandlimited second-order kernel are given in Appendix C. The 49 coefficients obtained captured 98.25% of the response energy. Therefore, the truncation error is 17.6 dB below the computed second-order response component. The truncation error for the coefficients of the second-order bandlimited kernel is significantly less than that for the direct realization second-order kernel primarily because the sampling interval has been decreased threefold for the Bandlimited realization. Nevertheless, it will offer a substantial computational efficiency improvement; the direct realization has 120 unique coefficients in the 15 by

15 array whereas the bandlimited kernel has only 28. Consequently, a greater than four-fold reduction in computational effort can be obtained while delivering an improved representation of the in-band response.

7.2.3 Third-Order Volterra Kernel Determination

Determination of the coefficients of the third-order Volterra kernels introduced an additional dimension to the computational problem. It had been observed that calculation of the second-order filter coefficients was substantially more time-consuming than calculation of the first-order coefficients[6]; whereas the calculation of all the coefficients for the linear filters required a matter of a few minutes, the calculation of the second-order coefficients required nearly 24 hours for the bandlimited realization.

This difference can be easily understood by considering the numerical integration. If a single integration requires N trapezoidal increments in order to achieve a predetermined accuracy, then a double integration of a similarly behaved integrand will require on the order

---

[6] All of the coefficient calculations described here were performed on an Intel 80386/80387 based computer (33 MHz). Clearly, more powerful systems are available, so that it is not particularly meaningful to discuss the tractability of a problem in absolute computation times. However, the relative durations of the calculations for coefficients of different-order Volterra filters provides insight into the computational issues involved.

of N^2 elemental regions in order to achieve a comparable accuracy. Correspondingly, a triple integration, as required to determine the coefficients for the third-order Volterra filters, will require on the order of N^3 elemental regions.

Given the similar growth in the number of coefficients required to "fill" a volume in a three-dimensional time-domain space with a linear dimension comparable to that used for the first- or second-order kernels, it is impractical to compute every coefficient. Therefore, a search for the most significant, i.e. greatest energy contributing, coefficients was conducted. This effectively amounts to a selective truncation of the time domain response rather than a uniform truncation at $\tau_i = T_{max}$.

Since the first- and second-order filters were well-behaved in the sense that their largest coefficients were concentrated around a single large peak value, it was relatively straightforward to calculate coefficient values, moving outward from the peak value found on the primary diagonal of the third-order Volterra kernel. Additional coefficients were computed until a sufficient proportion of the total third-order response energy was obtained. The implementation of this approach resulted in a disproportionate amount of time being spent to determine

the direct realization of the discrete-time Volterra series for use as a reference case.

As for the linear and second-order filters, coefficient values for the third-order Volterra filters were calculated as the numerical inverse Fourier transform integral for the specific sampling instants of interest. The integral to be computed is:

$$h_3(l\Delta t, m\Delta t, n\Delta t) = \tag{6}$$

$$\int_{-1000}^{1000}\int_{-1000}^{1000}\int_{-1000}^{1000} H_3\left(f_1,f_2,f_3\right)M_3\left(f_1,f_2,f_3\right)\exp\left[j2\pi\left(f_1 l\Delta t+f_2 m\Delta t+f_3 n\Delta t\right)\right]df_1\,df_2\,df_3$$

where:

$$H_3\left(f_1,f_2,f_3\right)=a_3H_1\left(f_1\right)H_1\left(f_2\right)H_1\left(f_3\right)H\left(f_1+f_2+f_3\right) \tag{7}$$

$$+a_2H\left(f_1+f_2+f_3\right)\left[H_1\left(f_1\right)H_2\left(f_2,f_3\right)+H_1\left(f_2\right)H_2\left(f_1,f_3\right)+H_1\left(f_3\right)H_2\left(f_1,f_2\right)\right]$$

is the symmetrized third-order nonlinear transfer function and the third-order mask function for the bandlimited kernel is:

$$M_3\left(f_1,f_2,f_3\right)=\begin{cases} 0, & |f_1|>W \\ 0, & |f_2|>W \\ 0, & |f_3|>W \\ 0, & |f_1+f_2+f_3|>W \\ 1, & \text{otherwise} \end{cases} \tag{8}$$

for $W = 1000$.

For the direct implementation of the third-order kernel, the mask is not required, i.e. it is identically 1 everywhere within the region of integration[7].

For the bandlimited third-order Volterra kernel, the sampling interval is $\Delta t = 1/2000$ second; for the direct implementation, the required sampling interval is $\Delta t = 1/6000$ second. Accordingly, it takes twenty-seven times as many coefficients to fill the same volume in three-dimensional time space for the direct implementation kernel as for the bandlimited Volterra kernel realization.

For the direct realization, we constrained the coefficient sampling time indices to: $0 \leq l, m, n \leq 11$. This resulted in a set of 1728 coefficients, only 364 of which (due to symmetry) are unique. For the bandlimited kernel we considered coefficients within the range: $0 \leq l, m, n \leq 5$. While this latter range of indices actually spans a significantly larger time interval in each of the three

---

[7] Actually, the bandwidth truncation which is applied to prevent response aliasing is equivalent to a mask which is 0 for $|f_i| > W$. From an implementation standpoint for the bandlimited kernel, only the $|\Sigma f_i| > W$ aspect of the mask need be explicitly applied separate from the limits of integration. In fact, this condition could be applied through the integration limits; however, this would require partitioning the region of integration and results in a much less compact expression.

temporal dimensions of the Volterra kernel, it contains just 216 coefficients, only 56 of which are unique.

Whereas coefficients for the linear filters could be computed in seconds each and second-order coefficients required, typically, tens of minutes to compute, the third-order coefficients consumed several hours apiece to compute. Accordingly, for the direct realization kernel, only 198 of the 364 unique coefficients within the index range were actually computed. These coefficients represented 858 of the total 1728 coefficients within the defined index range. The calculated coefficients represented only 70.6% of the response energy, however, leaving the truncation energy error 5.3 dB below the energy of the infinite-duration third-order response. While additional coefficients may be calculated to increase the proportion of the total response energy captured by the filter realization, no remaining unique coefficient is expected to deliver more than 0.15% of the total response energy.

For the bandlimited filter realization, 38 unique coefficients representing 132 of the 216 coefficients within the defined index ranges were computed. These coefficients represented 89.0% of the total third-order bandlimited response energy, leaving the truncation error 9.6 dB down.

Visualization of a three-dimensional filter is difficult in a single diagram. We have found it helpful to construct planar slices of the third-order kernels as a means of viewing the characteristic piecewise. The complete characteristics for both the direct realization and bandlimited third-order Volterra kernels are provided in Appendix C. Figures 7-4, 7-5, and 7-6 show the $t = 1/6000$, $t = 5/6000$, and $t = 9/6000$ planes respectively of the direct realization of the third-order Volterra kernel.

Similar to the plots shown for the second-order Volterra filters, Figures 7-4, 7-5, and 7-6 show $h_3(l\Delta t,\ m\Delta t, n\Delta t)$ for fixed l over the m, n plane. Figure 7-4 shows the $l = 1$ plane; Figures 7-5 and 7-6 show the $l = 5$ and $l = 9$ planes, respectively.

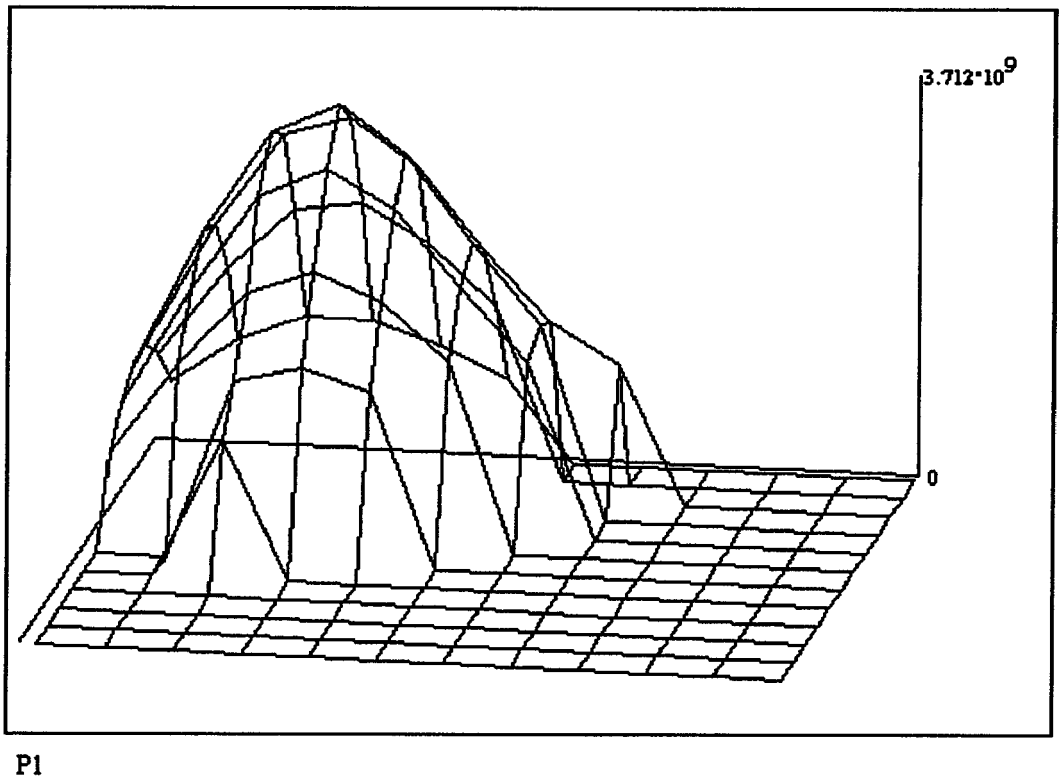


Figure 7-4: The $t = 1/6000$ Plane of the Third-Order Volterra Kernel (Direct Realization)

The sections of the displayed grid which are zero typically indicate coefficients which were not computed; however, these are expected to be less than the adjacent, computed coefficients. The time origin for the two variable dimensions of the coefficient surface is at the upper left hand corner of the plot. Note that the surface plot presentations of the Volterra kernel slices deceptively suggest that the peak coefficient magnitudes are similar from plane to plane; in fact, they vary

substantially. The largest coefficient calculated was the $l=4, m=4, n=4$ coefficient. Its magnitude was more than five times that of any coefficient in the $l=0$ plane, and nearly two and one-half times that of any coefficient in the $l=11$ plane.

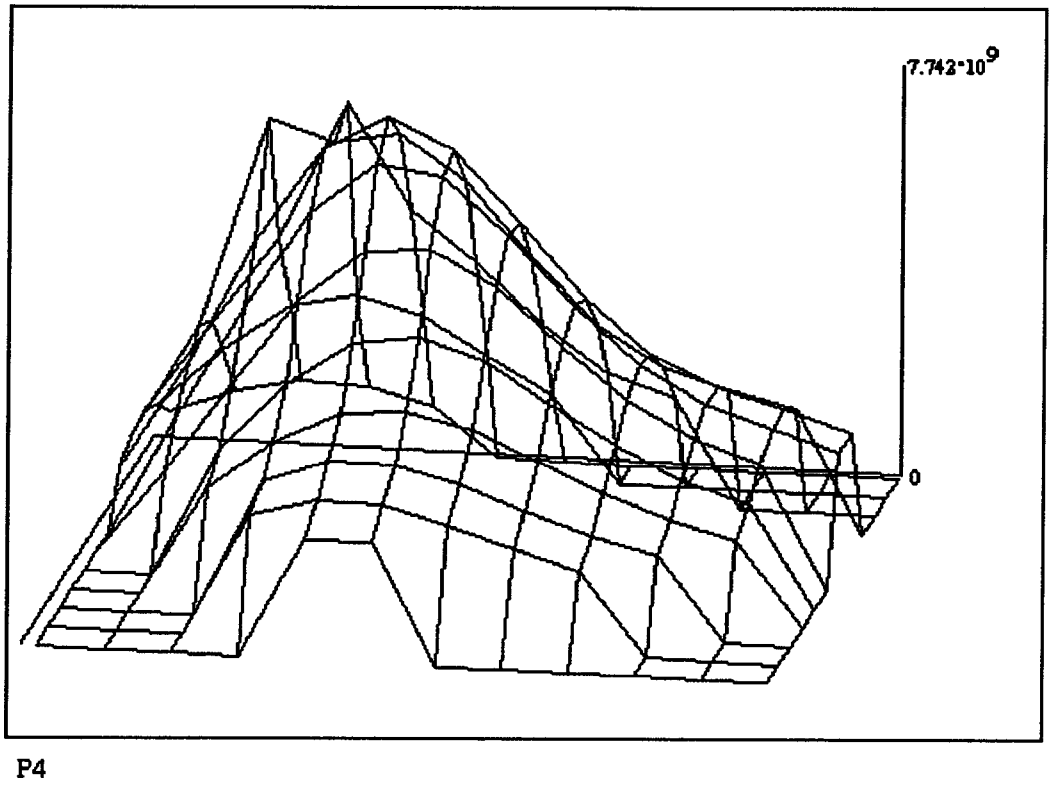


Figure 7-5: The $t=5/6000$ Plane of the Third-Order Volterra Kernel (Direct Realization)

It can be seen in the figures that substantially more of the coefficients in the $l=5$ plane of the direct realization kernel have been computed than were computed

for $l=1$. Due to the generally larger values of the filter coefficients in the $l=5$ plane, it was more advantageous from a truncation error minimization perspective to expend the computational effort to compute these coefficients than to compute additional coefficients on the fringes of planes further removed from the peak coefficient value.

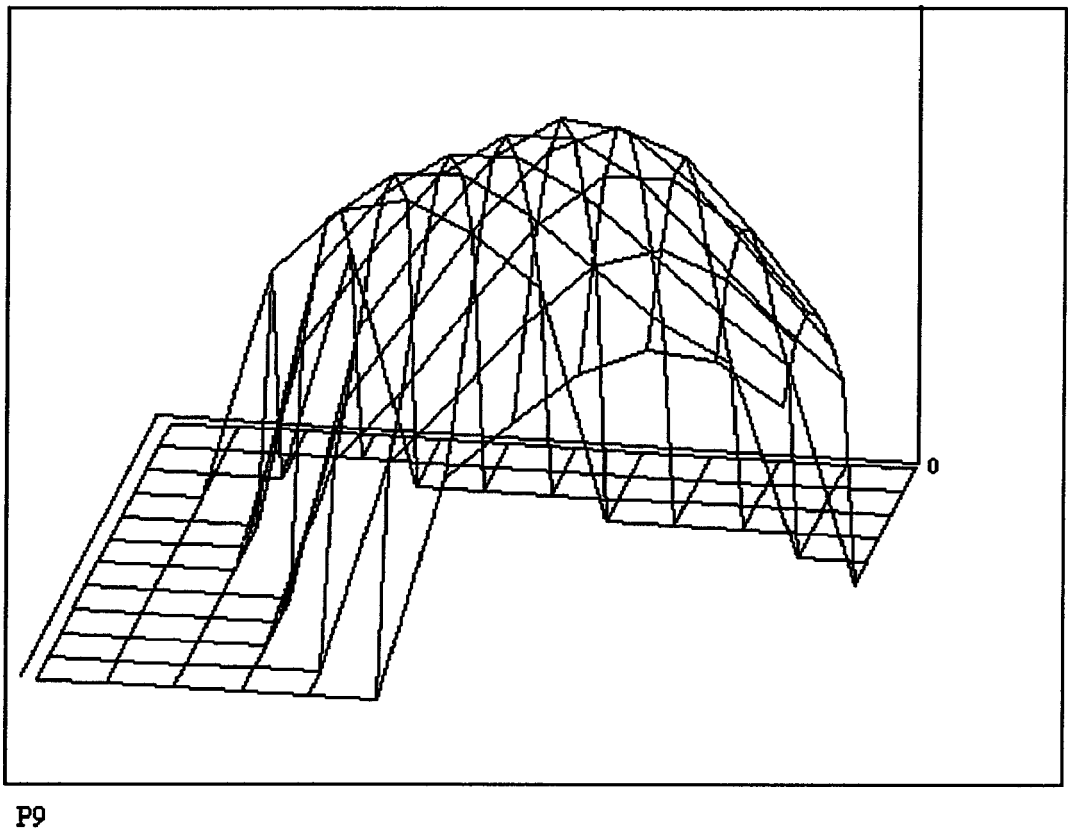


Figure 7-6: The $t=9/6000$ Plane of the Third-Order Volterra Kernel (Direct Realization)

The trend which is evident in Figures 7-4, 7-5, and 7-6 is that the peak value of the Volterra kernel tends to occur when the delay in each of the three time variables, i.e. τ_1, τ_2, τ_3, is relatively equal. The largest values of $h_3(\tau_1, \tau_2, \tau_3)$ occur very close to the principal diagonal: $\tau_1 = \tau_2 = \tau_3$.

The shape of the characteristics exhibited by the planar slices through the third-order bandlimited Volterra kernel are substantially similar to those shown in Figures 7-4, 7-5, and 7-6. Due to the much smaller number of coefficients required for the bandlimited kernel representation, however, there are a generally insufficient number of points in any one plane to give an especially helpful view of the response. The full bandlimited filter coefficient set is given in Appendix C.

7.3 Selection of System Inputs

In order to demonstrate and assess the performance of the various nonlinear system realizations, a set of test signal inputs was constructed. These included single sinusoids and sums of sinusoids. The amplitudes of these signals were carefully controlled to assure that meaningful responses were obtained. For too small an amplitude, the circuit will behave in an essentially linear manner, such

that the nonlinear Volterra series terms have magnitudes smaller than the error of the first-order response representation. On the other hand, for large input amplitudes, the actual system response would be dominated by fourth and higher order terms, rendering a third-order Volterra series model inadequate.

Based on the practical limitations of obtaining a sufficient number of third-order kernel coefficients to make the truncation error for the direct realization discrete-time Volterra kernel arbitrarily small, it was important to choose an input amplitude range which permits a realistic comparitive assessment of the various models. To do this, frequency-domain analysis based on the nonlinear transfer functions obtained by the harmonic probing technique, as illustrated in Chapter 2, was utilized.

7.3.1 Response Estimation

As a basis for estimating the expected response amplitudes, a single complex exponential with an angular frequency, ω, equal to the 3 dB frequency, k, of the associated linear circuit was chosen. Then, the response magnitudes for harmonic excitation at angular frequency ω

were determined as a function of input amplitude, A. We obtained:

$\|Y_1(k)\| = 0.47A$	based on $H_1(k)$
$\|Y_2(2k)\| = 0.66A^2$	based on $H_2(k, k)$
$\|Y_{3,a}(k)\| = 6.58A^3$	based on $H_{3,a}(k, k, -k)$
$\|Y_{3,b}(k)\| = 2.68A^3$	based on $H_{3,b}(k, k, -k)$
$\|Y_{4,a}(2k)\| = 19.63A^4$	based on $H_4(k, k, k, -k)$

We have shown the estimate of the most significant fourth-order term to point out that we must assure that the higher-order terms which are eliminated by truncation of the Volterra series do not dominate the response. The two terms of the third-order response are shown separately to illustrate their relative significance. Depending on the frequency of the excitation, the two terms may add constructively or destructively. By choosing an input signal amplitude of $A = 0.075$ Volt, we obtain:

$\|Y_1(k)\| = 0.035$	
$\|Y_2(2k)\| = 0.0037$	9.8 dB below $\|Y_1(k)\|$
$\|Y_{3,a}(k)\| = 0.0028$	11.0 dB below $\|Y_1(k)\|$
$\|Y_{3,b}(k)\| = 0.0011$	14.9 dB below $\|Y_1(k)\|$
$\|Y_{4,a}(2k)\| = 0.0006$	17.6 dB below $\|Y_1(k)\|$

7.4 *Error Estimates for Discrete Volterra Series Realizations*

Prior to computing results for the various test inputs, error estimates were obtained for each of the discrete-time Volterra series realizations which are described here. These estimates are important to evaluating the validity of results obtained by a particular technique and, accordingly to the selection of a technique for application to a specific problem.

7.4.1 *Direct Volterra Series Response Error Estimate*

Given our determination of the coefficient truncation errors of the second- and third-order direct kernel realizations (11.2 dB for the second-order and 5.3 dB for the third-order), the error of the second-order response realization will be 21.0 dB below the first-order response[8]. Similarly, the error of the third-order response realization will be 16.3 dB below the first-order response. Since the dominant component of the fourth-order response is 17.6 dB below the first-order response, the elimination of the fourth order term by truncating the Volterra series to third-order results in an error which is

[8] This value represents the combination of the magnitude of the true second-order response, expressed relative to the first-order response, and the level of the second-order filter realization error relative to the true second-order response.

below the realization error for the included terms. Since the first-order filter truncation error is 25 dB down, it will not significantly affect the overall accuracy of the model.

7.4.2 Bandlimited Volterra Series Response Error

The increased sampling interval of the bandlimited Volterra series afforded the opportunity to capture a greater proportion of the total response energy in a smaller number of coefficients than did the direct realization. As obtained in Sections 7.2.2 and 7.2.3, the response truncation error for the second-order bandlimited Volterra kernel was 17.6 dB down and that of the third-order bandlimited Volterra kernel was 10.0 dB down. Accordingly, these values place the second-order filter realization error (for a 3-dB frequency input at 75 millivolts) 27.4 dB below the linear response component. The third-order response truncation error will be 21.0 dB below the linear response component. Since the primary fourth-order response component is only 17.5 dB below the linear response term, the Volterra series truncation error becomes the dominant error in this case. The linear (first-order) filter error (-23.1 dB) is not the smallest error in this model, however it should not materially affect the overall accuracy of discrete-time processing with the Bandlimited realization model because it is still

5.6 dB below the Volterra series truncation error and 5.5 dB below the second-order bandlimited Volterra filter truncation error.

7.4.3 Serial Realization Response Error

For the serial realization of the Volterra series, the filter truncation error is due to the error in the determination of the linear filter and the cascaded processing of corrupted intermediate signals.[9] This can readily be estimated based on the filter truncation error already obtained.

Assuming that the first-order filter response truncation error results in a signal error component which is $e_1(t) = \gamma y_1(t)$ where $y_1(t)$ is the intended first order response, the second-order response error must consist of the composite of the result of second-order processing on the first-order error and the second-order filter response truncation error.

The second-order processing effect is the direct result of the signal squaring operation. With reference to Figure 5-3, for the signal plus error term, we obtain:

$$e_2(t) = [y_1(t) + e_1(t)]^2 - [y_1(t)]^2 \qquad (8)$$

---

[9] In the direct and bandlimited Volterra series realizations, there is no subsequent processing of the various response components.

Based on the first-order error expression, $e_1(t) = \gamma y_1(t)$, this is:

$$e_{2,in}(t) = \left(1+\gamma\right)^2 [y_1(t)]^2 - [y_1(t)]^2 = \left(2\gamma+\gamma^2\right)[y_1(t)]^2 \qquad (9)$$

Since the filter used to obtain the second-order response for the serial realization is precisely the filter which was used to obtain the first order response, the proportional error coefficient, γ, is identical to that which pertains to the first-order response. Furthermore, since the filter is linear, it applies equally to the signal and error components of its input. For small values of γ, we may neglect the γ^2 term in equation (9) and the effect of filter error on the error input component. Accordingly, the filter output response can be stated as:

$$e_{2,out}(t) \approx \left(2\gamma+\gamma\right) y_2(t) \qquad (10)$$

Therefore, the relative response error of the second order response will be approximately 4.8 dB poorer than that obtained for the first-order response. Given that a linear filter response truncation error of -30.9 dB can be obtained, the second-order response error will be 26.1 dB below the second-order response. This is appreciably better than the

error obtained for either the direct realization or the bandlimited realization of the discrete-time Volterra kernels.

The same approach can be applied, separately, to the a and b components of the third-order serial realization response. The results of the third-order error analysis are:

$$e_{3,a,out}(t) \approx \left(3\gamma+\gamma\right)y_{3,a}(t) \tag{11a}$$

$$e_{3,b,out}(t) \approx \left(4\gamma+\gamma\right)y_{3,b}(t) \tag{11b}$$

This means that the relative response errors for the a and b components of the third-order Volterra series serial realization will be 6.0 dB and 7.0 dB, respectively, poorer than the first-order response error.

The relative response error deteriorates as the order of a Volterra series term increases. However, for weakly nonlinear systems, the relative significance of each higher order term is expected to decrease. To the extent that the significance of the higher-order terms decreases as fast as the relative error level increases, the error floor of the serial realization model will remain fixed with respect to the magnitude of the linear response component.

7.4.4 Picard Iteration Response Error

In order to estimate the error of the Picard iteration procedure for computing the Volterra series response, we follow essentially the same procedure as for the Serial Realization. With reference to Figure 6-2, it may be seen that the signal computed at the output of the summer following the first nonlinear section is:

$$y_{p,1}(t) = y_1(t) + y_2(t) + y_{3,a}(t) + \cdots + y_{m,a}(t) \tag{12}$$

where m is the degree of the polynomial approximation used to represent the nonlinear constitutive relationship, and only the a components of each higher order response are contained in the first Picard iterate.

The relative error levels of each of the terms are precisely the values computed for the serial realization in section 7.3.4. These error terms become a part of the input to the polynomial operator in the second stage of the Picard iteration procedure. Consequently the relative error levels will be compounded in much the same manner as interest on a savings account. Therefore, the relative error levels after the each subsequent iteration can be expected to be at least 3 dB higher than the levels at the output of the previous nonlinear iteration (except the first-order relative error level which will remain constant

since it is not processed by the nonlinear operator). Unless each successive Picard iteration stage improves the response content by an amount greater than the error increase, it will degrade rather than enhance the overall response accuracy.

7.5 *Discrete-Time Processing Results*

The discrete-time processing models which have been constructed for the example nonlinear circuit were evaluated using two basic input signals. First, each model was exercised for an input consisting of a single sinusoidal signal. This provided an initial assessment of their suitability. Subsequently, two of the models were run for an input consisting of three sinusoidal signals at incommensurable frequencies. In each case, the exact response amplitude at each fundamental, harmonic, and intermodulation frequency was determined from the continuous frequency domain (non)linear transfer functions in order to provide a basis for model accuracy assessment.[10] The responses are examined in the following sections.

---

[10] Determination of the exact response of a nonlinear system to an input consisting of discrete-frequency signals provides a means of calibrating a model. This establishes its validity so that it may be applied with confidence to problems for which the inputs do not contain discrete frequency components and thus cannot be evaluated via a frequency-domain Volterra analysis.

7.5.1 Single Sinusoidal Response Results

In order to assess the fidelity with which each of the models responded to a single sinusoid input, the responses of each were computed for a sinusoidal signal with an angular frequency equal to the 3 dB frequency of the associated linear circuit of our example. Since the 3 dB frequency, approximately 159 Hertz[11], is less than one-third of the input bandlimit selected for our evaluations of the example system, a third-order Volterra series response is completely contained within the input passband. Therefore, all of the models, including the Bandlimited Volterra series, should faithfully deliver the full third-order response. A sinusoidal signal amplitude of 150 millivolts was chosen, consistent with the assessment of section 7.3.1 which was made for a complex-exponential signal having an amplitude of 75 millivolts. Since the sinusoid is the sum of two complex exponentials at complementary frequencies, the magnitudes of the fundamental-frequency sinusoidal response terms contain contributions from both the first and third-order Volterra series terms.

---

[11] The angular frequency of 1200 radians/sec assures that the input will not be sampled at the same sinusoidal phase in each period.

7.5.1.1 *Direct Volterra Series Realization Response to a Single Sinusoid*

It is helpful to examine the first-, second-, and third-order terms of the Volterra series response individually before considering the composite third-order response. Figure 7-7 shows the computed first-order response (broken line trace, $Y1_{nn}$) plotted against the input signal (solid trace, xs_{nn}). The amplitude of the computed

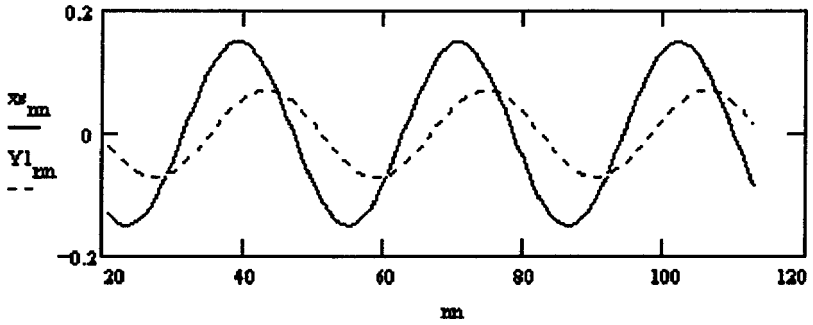


Figure 7-7: The First-Order System Response

response is 0.0711 volt, only 0.0004 volt more than the expected 0.0707 volt value. The expected 45 degree phase shift (delay) is also clearly evident in the Figure. In the Figure, the time scale is marked according to the index of the response sample sequence with each increment equal to the sampling interval, 1/6000 second.

The second-order response component, computed using the direct realization of the Volterra series, is shown in

Figure 7-8. Here, the computed response consists of two components: a DC term and a double-frequency term.

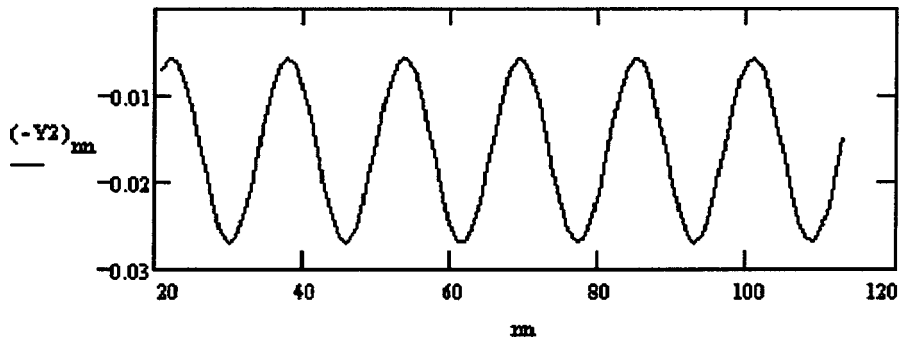


Figure 7-8: The Second-Order Volterra Series Response

The amplitude of the computed double-frequency sinusoid is 0.0105 volt. It rides on a DC term of -0.0165 volt. The corresponding expected values for these two components are 0.0120 volt and -0.0167 volt, respectively. The errors in these two terms are well within the accuracy predicted for the second-order term (error 11.2 dB below the response).

The third-order response component computed via the direct realization of the Volterra series is shown in Figure 7-9. The simultaneous presence of fundamental frequency and third-harmonic terms is evident in the Figure. The filter rolloff at the third harmonic frequency, combined with the three-to-one trigonometric dominance of the fundamental frequency[12], results in the

greater significance of the third-order contribution at the fundamental frequency when compared to the third-order, third-harmonic level. The amplitude of the fundamental-frequency term in the third-order response is 0.0255 volt and the amplitude of the third-harmonic response term is 0.00516 volt. These values may be compared with the expected values of 0.0290 volt and 0.00314 volt, respectively.

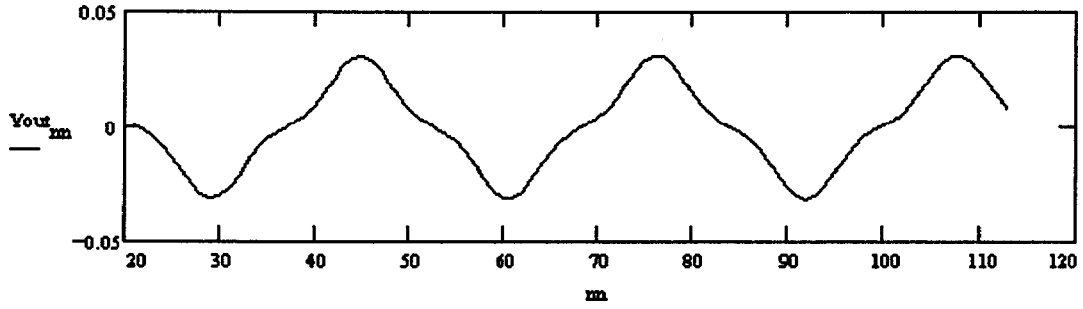


Figure 7-9: Third-Order Volterra Series Response

The smaller amplitudes of the second- and third-order response components than that of the first-order response result in a combined third-order response which roughly retains the appearance of a (distorted) sinusoid at the fundamental input frequency. The combined third-order response is shown in Figure 7-10. The spectrum of the response is shown in Figure 7-11 where the second and third harmonic terms are evident.

---

[12] $\sin^3(\omega t) = 3/4 \sin(\omega t) + 1/4 \sin(3\omega t)$

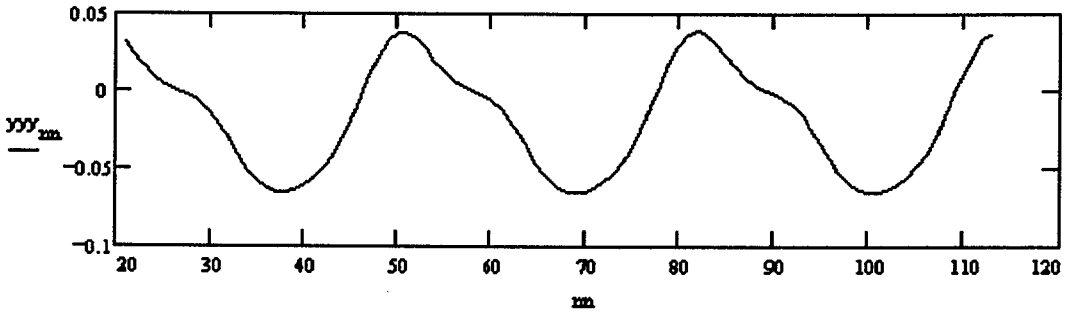


Figure 7-10: Combined Third-Order System Response

The composite sinusoidal response amplitude at the input frequency for the complete response is 0.0465 volt. By the exact analysis using the frequency-domain nonlinear transfer functions, a fundamental frequency response amplitude of 0.0563 volt was expected. Accordingly, the error, 0.0098 volt is approximately 15.2 dB below the expected value. That this combined error is greater than the sum of the first- and third-order contribution errors indicates that a relative phase error is also present between the first and third-order response components. This is consistent with the sparseness of the third-order filter coefficients. The DC, second-harmonic and third-harmonic terms in the complete third-order response are identical to those shown above for the second- and third-order Volterra series terms.

The number of time-domain samples (2400) used to obtain the spectral plot was selected to yield a spectral resolution of 2.5 Hertz. Only the first 1000 Hertz range is shown, as there are no response components for a third-order system beyond that for the particular input used.

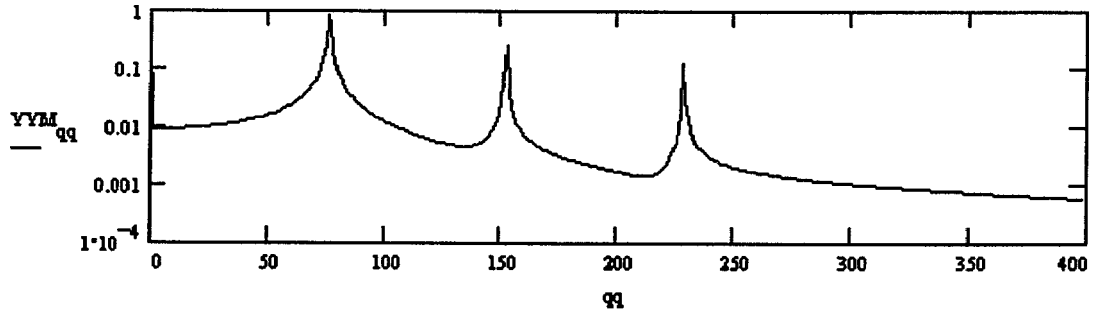


Figure 11: Volterra Series Response Spectrum

For the direct realization, only the third-harmonic frequency response term, as computed by the third-order Volterra filter deviated significantly from the expected response value. On balance, excellent agreement is obtained between the third-order discrete-time direct realization of the Volterra series and the theoretically expected third-order system result for a single sinusoid input. Computationally, however, the direct realization is the most expensive to implement. For the present case, the computation of a response segment of 0.4 second required

approximately 3 hours. Thus, coupled with the fact that determination of the third-order filter coefficients had to be terminated with only about 70% of the response energy obtained, this "brute-force" approach is indeed very unwieldy.

7.5.1.2 Bandlimited Volterra Series Response to a Single Sinusoid

The same input signal as used to evaluate the performance of the direct Volterra series realization was used to exercise the discrete-time, Bandlimited Volterra series model. Again, a 0.4 second response record was obtained, allowing the response spectrum to be computed with the same 2.5 Hertz granularity used for the direct realization.

There is little benefit to showing plots of the computed waveforms for the bandlimited response as, except for a sampling granularity difference[13], there is no visually perceptible difference from those computed by the direct response. The sinusoidal response component amplitudes computed using the Bandlimited Volterra series model are shown in Table 7-1, along with those for the direct realization and the expected values.

---

[13] The sampling interval for the Bandlimited Volterra series model was 1/2000 second.

Overall, the response computed via the Bandlimited model was comparable in accuracy to the direct realization for this particular sinusoidal input. However, the results for the 0.4 second record were obtained in only 8 minutes instead of the 3 hours required by the direct response. Again, the most severe error was encountered for the third-harmonic term where the error was only 3.5 dB below the expected response amplitude.

7.5.1.3 *Serial Volterra Series Realization Response to a Single Sinusoid*

The Serial Volterra series realization was run for the same sinusoidal excitation which was used for the direct and Bandlimited Volterra series realizations. A response record of 0.4 second duration was computed in approximately 34 minutes; this is significantly faster than the direct Volterra series realization, but slower than the Bandlimited Volterra series model. However, the only required preparation for executing the serial realization model is determination of the linear filter coefficients which requires less than 1 minute, whereas the direct and Bandlimited models required hours apiece for computation of each third-order filter coefficient.

The ease with which the Serial Realization can be set up points to a significant advantage where fast-turnaround

results are necessary for a relatively short response computation. The substantial set-up effort required for the Bandlimited Volterra series will place that model at a distinct disadvantage for single short response computations. The shorter run time of the Bandlimited Volterra series will be an advantage, however, where the model must be exercised many times for different input signals or for extended response durations.

The computed response amplitudes for the serial realization are also tabulated in Table 7-1. The errors for the serial realization were less than those obtained with the Bandlimited model at every response frequency. They were less than those of the direct realization for the fundamental and third-harmonic frequencies, essentially equal for the DC component, and somewhat poorer for the second harmonic (9.9 dB below the expected signal amplitude). Overall, the serial realization delivered a better result due to the improvement of the third-order response characteristic.

The serial realization response appears substantially similar to the responses obtained by the direct and Bandlimited Volterra Series realizations, however, the

Frequency	Expected Amplitude	Direct Realization	Bandlimited Realization	Serial Realization
DC	-0.0167 v	-0.0165 v	-0.0174 v	-0.0165 v
fundamental	0.0563 v	0.0465 v	0.0502 v	0.0558 v
2^d harmonic	0.0112 v	0.0105 v	0.0075 v	0.0076 v
3^d	0.00314 v	0.00520 v	0.00445 v	0.00327 v

Table 1: Comparison of Model Results for a Single Sinusoidal Input

improved performance at the fundamental frequency and adjustments to the harmonic amplitudes result in a slight appearance diffence. The computed serial realization response is shown in Figure 12.

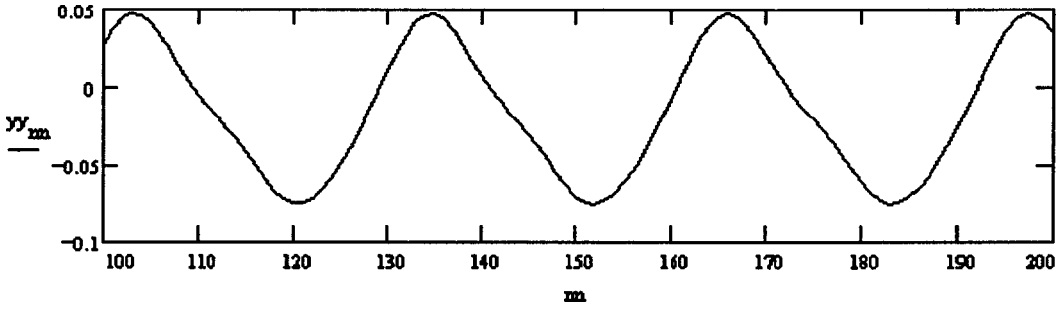


Figure 12: Computed Third-Order Serial Realization Response

7.5.1.4 Picard Iteration Realization Response to a Single Sinusoid

A Picard iteration realization was constructed to be consistent with the third-order Volterra series

realizations. Recognizing that the required sampling rate depends on both the number of iterations and the degree of the polynomial approximation chosen to represent the nonlinear element constitutive relationship, a realization consisting of two iterations and using a third degree polynomial was constructed as shown in Figure 6-2.

For the chosen model, with a maximum input frequency of 1000 Hertz, the maximum response frequency may be as great as 9000 Hertz. Therefore, a sampling interval of 1/18000 second was utilized for the Picard iteration model. This rate will allow parts of the ninth-order Volterra series response to be included in the Picard iteration model output.

The increased sampling rate (three times that of the direct Volterra series and Serial realizations) requires that the filters (all linear, as in the Serial realization) include three times as many coefficients for the same time-domain truncation length. This significantly increases the computational cost of executing the model.

The output of the first Picard iteration stage is the same as the third-order Volterra series response, except for the b term in the third order kernel, i.e., $h_{3,b}(\tau_1,\tau_2,\tau_3)$, equation (2-69). Figure 7-13 shows the first iteration output.

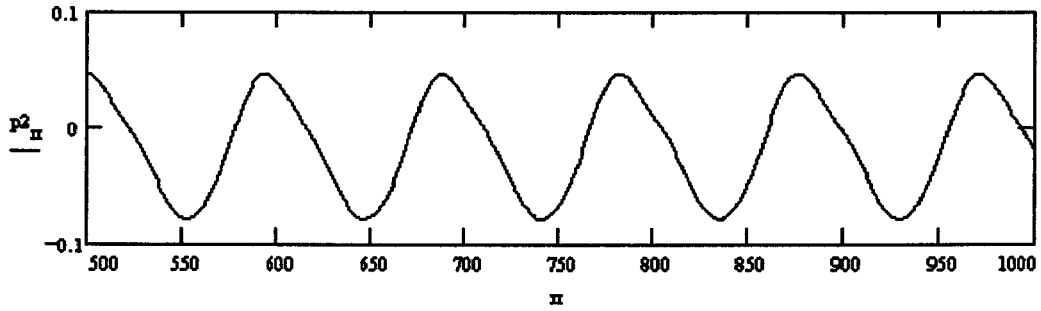


Figure 13: The First Picard Iteration Response

The first iteration response exhibits nearly the same distortion as the serial realization. The b-term of the third-order Volterra kernel is about a factor of 5 smaller than the a-term. Accordingly, its influence on the overall response is small.

The second Picard iteration response is shown in Figure 14. The effect of the fourth- and higher-order terms is such that they tend to cancel some of the distortion components produced by the second- and third-order terms.

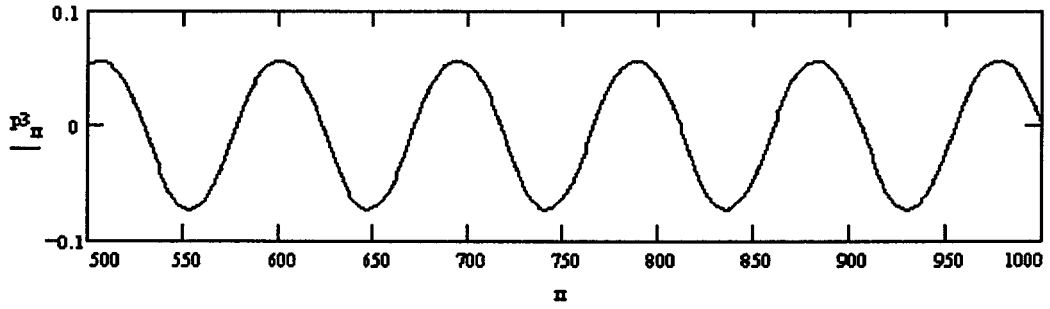


Figure 14: The Second Picard Iteration Response

The spectrum of the second Picard iteration response is shown in Figure 15. Comparison of Figure 15 with Figure 11 shows that the second and third harmonics of the input frequency are significantly lower in the second Picard iteration response. The measured harmonic amplitudes for the second iteration response are:

DC	-0.00527 volt
fundamental	0.06635 volt
2nd harmonic	0.00230 volt
3rd harmonic	0.00162 volt
4th harmonic	0.00060 volt
5th harmonic	0.00021 volt

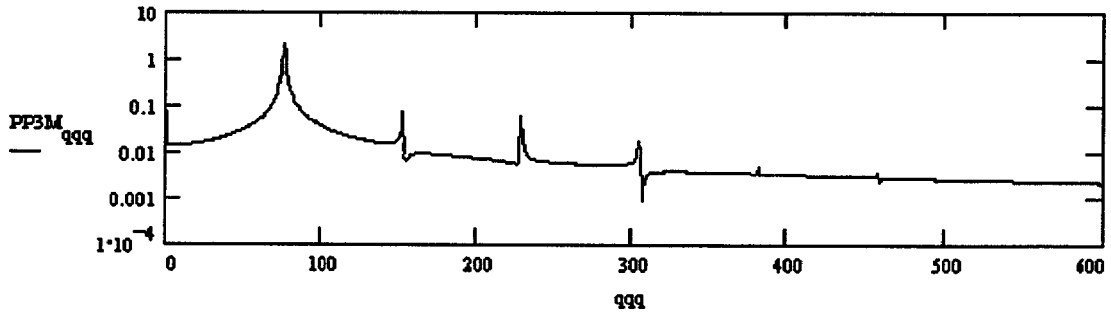


Figure 15: Spectrum of the Second Picard Iterate

A frequency domain analysis for a complete fifth-order Volterra representation of the system provides the following expected values for the DC component and the first five harmonics of the input signal.

DC	-0.07962 volt
fundamental	0.07450 volt
2nd harmonic	0.02402 volt
3rd harmonic	0.01038 volt
4th harmonic	0.00131 volt
5th harmonic	0.000386 volt

The observed values for the amplitudes of the sinusoidal components of the response at the input frequency and its harmonics do not correspond well to the expected values as derived from the first five Volterra series terms. However, it is also clear that some response

contributions up to ninth-order are present, but that only the first three Volterra series terms are fully computed by a two-iteration Picard realization[14]. Consequently, it is difficult to say with certainty that the Picard iteration technique is accurate; it is fair to state that the Picard iteration response to a single sinusoid for the example system does not correspond well to the response predicted by a frequency domain Volterra analysis.

7.5.2 *System Response Calculation for a Multiple Sinusoid Input Signal*

The single sinusoid results presented in Section 7.5.1 do not demonstrate the validity of the models under a variety of conditions, particularly for inputs which would be expected to produce harmonics and intermodulation products outside the input passband. Therefore, a second input signal was constructed to further exercise the models. The signal was constructed of three sinusoids at angular frequencies of: $\omega_1 = 1000$ radians/second (approximately 159 Hertz), $\omega_2 = 2828.43$ radians/second (approximately 450 Hertz), and $\omega_3 = 2\pi 850$ radians/second

---

[14] In principle, the specific part of the Volterra series response to which the Picard iteration response corresponds can be exactly determined. However, the fifth-order nonlinear transfer function contains six separate terms with over fifty argument permutations which for which the various terms must be computed in order to obtain the fifth-order response to a sinusiodal input. Extension to the sixth or higher order is impractical.

(exactly 850 Hertz). The sinusoid amplitudes were all set to 150 millivolts.

Due to the uncertain nature of the Picard iteration model, it was not used for the three-sinusoid input. Similarly, because there is little benefit to the direct Volterra series realization and it requires lengthy computation times, the direct realization was not used for the multiple sinusoid-input case.

7.5.2.1 *Multiple Sinusoid Response Computation Using the Serial Volterra Series Realization*

The incommensurate nature of the three sinusoidal components of the input signal results in an aperiodic input signal. A segment of the input (approximately 17 milliseconds) is shown in Figure 16.

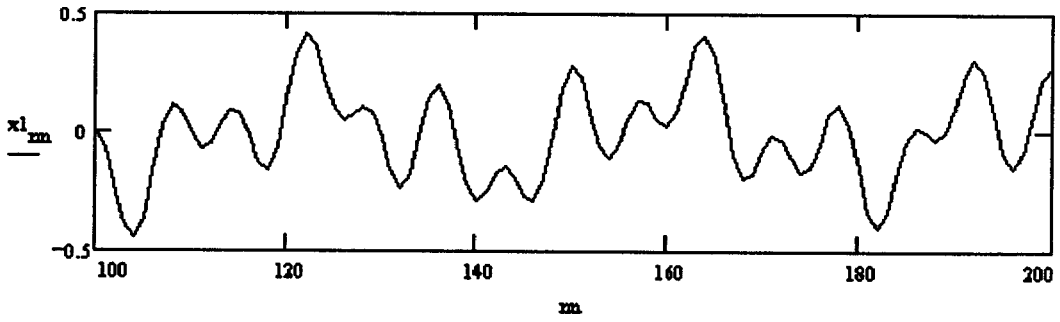


Figure 16: Multitone Input Signal for the Serial Realization

The corresponding segment of the response, as computed using the Serial Volterra series realization is shown in Figure 17.

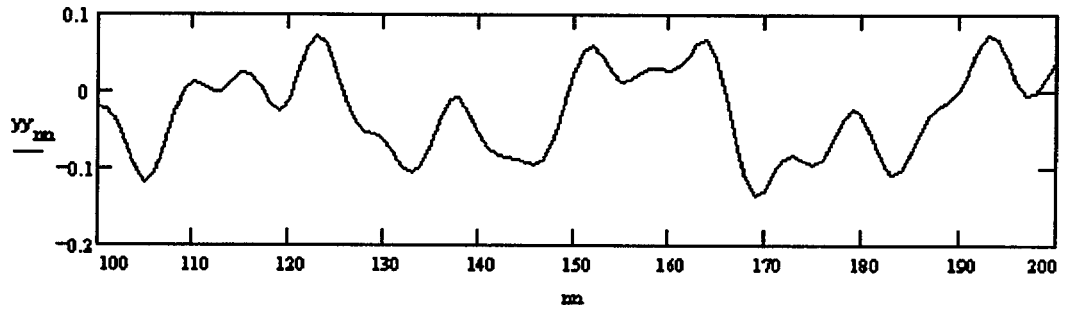


Figure 17: Serial Realization Response

The response spectrum is shown in Figure 18. Table 7-2 identifies the intermodulation frequencies, response amplitudes, and the expected values of the intermodulation response amplitudes as determined by a frequency-domain Volterra analysis.

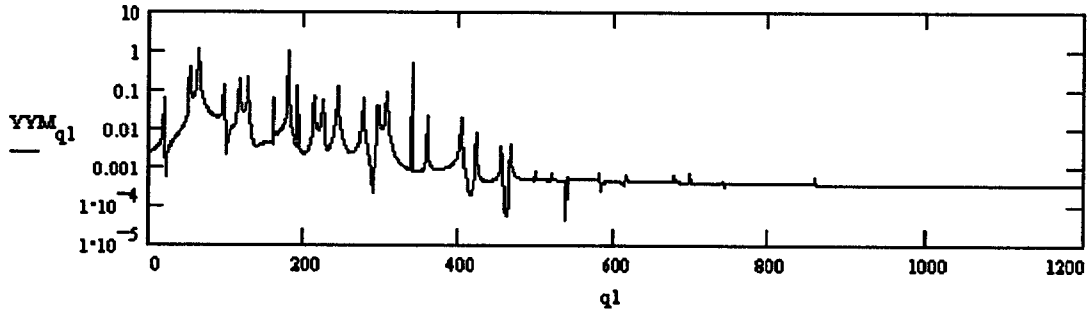


Figure 18: Serial Realization Response Spectrum

Frequency	Response Components	Expected Amplitude	Serial Realization	Bandlimited Realization
DC	$f_i - f_i$	-0.02636 v	-0.02658 v	-0.02375 v
50 Hz	$2f_2 - f_3$	0.00249 v	0.00262 v	0.00147 v
132 Hz	$f_2 - 2f_1$	0.01642 v	0.01737 v	0.00139 v
159 Hz	$f_1; 2f_1 - f_1$	0.05568 v	0.05048 v	0.00244 v
241 Hz	$f_3 - f_1 - f_2$	0.00612 v	0.00641 v	0.00100 v
291 Hz	$f_2 - f_1$	0.01098 v	0.01030 v	0.00121 v
318 Hz	$2f_1$	0.01012 v	0.01007 v	0.00087 v
400 Hz	$f_3 - f_2$	0.00246 v	0.00269 v	0.00257 v
450 Hz	$f_2; 2f_2 - f_2$	0.03821 v	0.04007 v	0.03829 v
477 Hz	$3f_1$	0.00501 v	0.00529 v	0.00128 v
532 Hz	$f_3 - 2f_1$	0.00331 v	0.00323 v	0.00045 v
559 Hz	$f_1 - f_2 + f_3$	0.00333 v	0.00306 v	0.00046 v
609 Hz	$f_1 + f_2$	0.00598 v	0.00581 v	0.00071 v
691 Hz	$f_3 - f_1$	0.00299 v	0.00291 v	0.00059 v
741 Hz	$2f_2 - f_1$	0.00248 v	0.00235 v	0.00078 v
768 Hz	$2f_1 + f_2$	0.00443 v	0.00436 v	0.00263 v
850 Hz	$f_3; 2f_3 - f_3$	0.02187 v	0.02200 v	0.02311 v
900 Hz	$2f_2$	0.00106 v	0.00105 v	0.00107 v
1009 Hz	$f_1 + f_3$	0.00209 v	0.00198 v	
1059 Hz	$2f_2 + f_1$	0.00153 v	0.00153 v	
1141 Hz	$f_2 + f_3 - f_1$	0.00169 v	0.00156 v	
1168 Hz	$2f_1 + f_3$	0.00159 v	0.00155 v	
1250 Hz	$2f_3 - f_2$	0.00022 v	0.00021 v	
1300 Hz	$f_2 + f_3$	0.00083 v	0.00084 v	
1350 Hz	$3f_2$	0.00020 v	0.00021 v	
1459 Hz	$f_1 + f_2 + f_3$	0.00122 v	0.00121 v	
1541 Hz	$2f_3 - f_1$	0.00033 v	0.00031 v	
1700 Hz	$2f_3$	0.00018 v	0.00019 v	
1750 Hz	$2f_2 + f_3$	0.00025 v	0.00027 v	
1859 Hz	$2f_3 + f_1$	0.00026 v	0.00026 v	
2150 Hz	$2f_3 + f_2$	0.00011 v	0.00013 v	
2550 Hz	$3f_3$	0.00002 v	0.00003 v	

Table 2: Multi-sinusoidal Response Characteristics

The correspondance of the computed intermodulation response amplitudes to the expected values is excellent. Most intermodulation products have an error greater than 25 dB below the exact value as computed using a frequency-domain analysis.

7.5.2.2 Multiple Sinusoid Response Computation Using the Bandlimited Volterra Series Realization

The reduced sampling rate which is applied to the Bandlimited Volterra series results in a substantially more coarse appearance of the signal waveforms. Figure 19 shows the three-sinusoid input signal as it was applied to the Bandlimited model. By comparison with Figure 16, it may be seen however, that the inputs are the same.

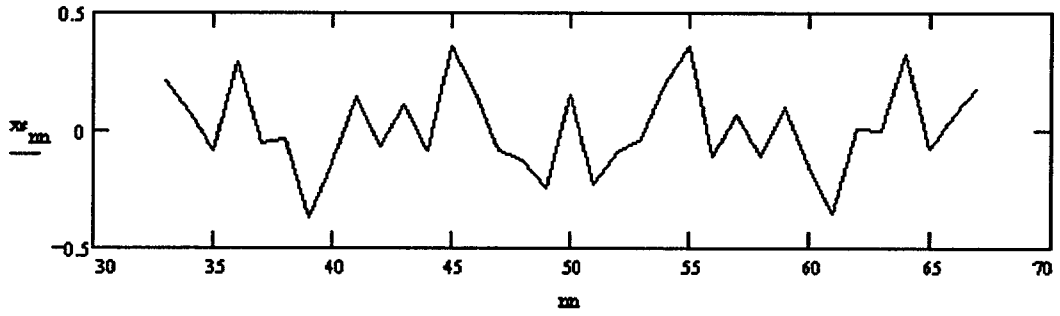


Figure 19: The Multi-sinusoid Input to the Bandlimited Volterra Series Model

The corresponding time interval of the response (chosen to also match the time interval shown in Figures 16

and 17) is shown in Figure 20. The response spectrum is shown in Figure 21 and the intermodulation product amplitudes are tabulated in Table 7-2.

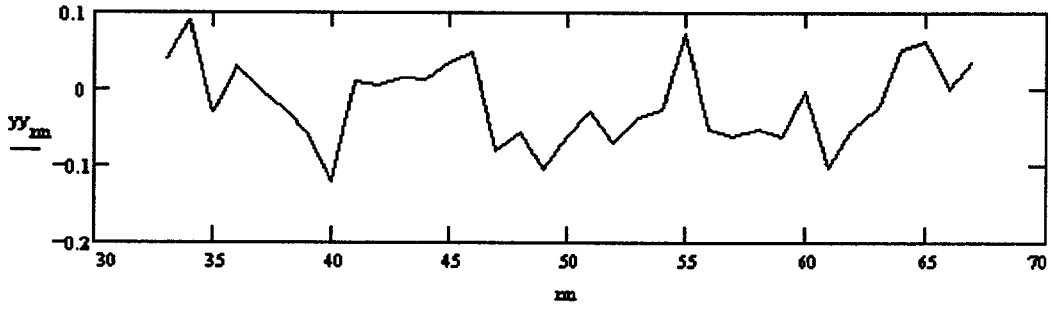


Figure 20: Bandlimited Volterra Series Response to the Multi-sinusoid Input Signal

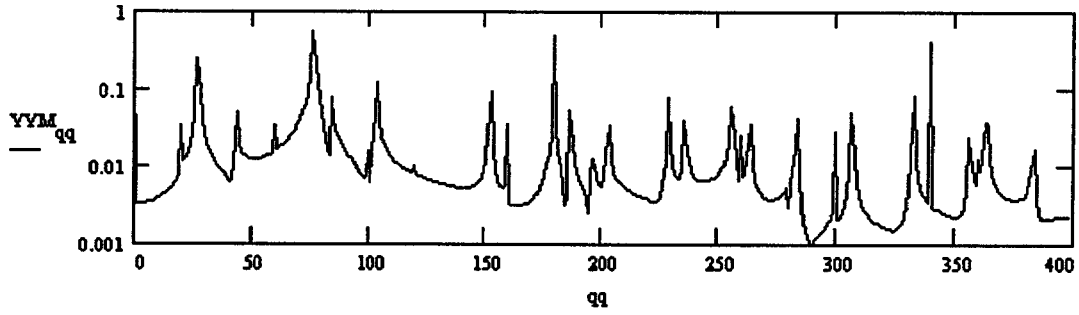


Figure 21: Bandlimited Volterra Series Response Spectrum

A visual comparison of Figure 20 with Figure 17 reveals substantial differences between the two response waveforms. These differences may also be seen in the large errors in the Bandlimited response intermodulation product amplitudes.

7.5.3 Analysis of the Response Calculations

The multi-sinusoidal input examples provide a much clearer picture of the performance of the discrete-time Volterra series models than did the single sinusoid case. The inadequacy of the Bandlimited model, due perhaps to the necessity of restricting the number of third-order discrete kernel coefficients, is most apparent. The serial realization, by contrast, delivered excellent approximations to virtually all of the intermodulation products.

CHAPTER 8
Conclusions

Discrete-time evaluation of nonlinear systems using techniques based on the Volterra series can be effective provided that accuracy is sufficiently controlled. This was demonstrated for the Serial realization technique. On the other hand, we have shown that other approaches to development of efficient discrete-time Volterra models encounter substantial difficulties and suffer from accuracy limitations which may make them unusable in some applications.

8.1 Assessment of the Direct Realization

Discrete-time processing using a direct Volterra series realization demands an enormous computational effort. Determination of the discrete-time filter coefficients for the third-order kernel used in our example required greater than two orders of magnitude more computation than did the execution of the resulting model[1]. Even so, the accuracy of the results obtained is inadequate to merit a recommendation of the technique.

---

[1] The purpose of this dissertation was not to develop the direct realization; however, a model based on the direct realization was used as a basis of comparison for both accuracy and computational-cost assessments of the three efficient techniques presented.

Calculation of the filter coefficients for higher-order Volterra kernels by numerical integration of the inverse Fourier transform integrals presents a formidable task. This necessitates compromise with respect to accuracy, both in the precision of the coefficients determined and their number. The particular example system considered in this dissertation did not allow acceptable accuracy to be obtained by directly sampling the continuous-time Volterra kernel. Other systems may permit the aliasing error to be bounded at an acceptable level when a direct sampling approach is applied.

8.2 Assessment of the Bandlimited Volterra Series

Application of the Bandlimited Volterra series technique to computing the in-band response of a nonlinear system is similar to the direct realization of the Volterra series. It permits a reduction of the sampling rate which may result in a significant computational savings relative to a direct realization; however, the effort required to produce a discrete-time response is still dominated by the determination of the filter coefficients. Unlike the direct Volterra series realization, the Bandlimited technique cannot be applied without computing the inverse Fourier transform integrals for the higher-order filter coefficients. We found that even though a computationally efficient realization (relative to the direct realization)

of the example system could be obtained by the Bandlimited Volterra series technique that, accuracy remained poor. Improvements to the accuracy of the Bandlimited realization will require that more coefficients be computed and that a tighter numerical integration tolerance be applied.

8.3 Assessment of the Serial Realization

Discrete-time computation of the response of the example system to single and multi-sinusoidal inputs with the Serial Realization of the Volterra series produced results which compared very favorably with the exact values as determined by a frequency domain Volterra analysis. The Serial Realization of the discrete-time Volterra series does not escape the sampling rate increase complexity issue; however, it entirely avoids the problems inherent to determining the higher-order filter coefficients. We found that while model execution time was greater than that of the Bandlimited realization, minimal effort was needed to set up the model.

Consequently, the Serial Realization provides an accurate, flexible technique for discrete-time nonlinear signal processing. Although our evaluations considered only a third-order system realization, the Serial Realization can be extended to fourth or higher orders with a far lesser increase in complexity than any of the other techniques presented. While the computational burden of

computing a response to a higher-order system will grow, the dominant contribution to this effort is the increase in sampling rate required to faithfully represent the response. The complexity of the model needed for a higher-order system, in terms of the number of filters required, is directly proportional to the order of the response computed.

The sequential filtering of the signal components within the Serial realization model shows that the higher-order Volterra series terms will tend to be more dispersive than the linear (first-order) component. This further demonstrates why capturing a sufficient proportion of the response energy of a discrete-time Volterra filter in a reasonable number of coefficients is so difficult.

8.4 Assessment of the Picard Iteration Realization

The use of the Picard iteration model for computing a discrete-time response to a nonlinear system produced results which are difficult to compare to any of the Volterra series realizations. The Picard technique computes, in the limit, exactly the same terms as the infinite Volterra series, but it incorporates the terms in a different order. In a finite accuracy realization, however, the Picard iteration approach accumulates error more rapidly than any of the Volterra series techniques. In our evaluations, the accuracy of the Picard iteration

model results was significantly poorer than the results obtained with the Serial Volterra series realization. (The Picard iteration technique bears the greatest structural similarity to the Serial realization.)

For the system considered here, the Picard iteration model required a significantly greater computational effort due to sampling rate considerations: the two-iteration Picard model required a sampling rate nine times the Nyquist rate of the input, whereas the Serial realization required a three-times-Nyquist rate. For a system which exhibits a significantly sharper frequency cutoff, the required sampling rate increase may be less dramatic; however, in the general case, the Picard technique will be less computationally efficient than the Serial realization.

8.5 Summary

We have examined three computationally efficient techniques for implementing a discrete-time Volterra series for a nonlinear system. In contrast to a direct, or brute-force, implementation of a discrete-time Volterra series, each of the techniques offers a substantial improvement in computational effort. With regard to accuracy, however, one technique stands out. The Serial realization of the Volterra series provided excellent representation of the exact response characteristics in our evaluations of multifrequency sinusoidal inputs. This

gives a high degree of confidence that a Serial Volterra series realization can give highly accurate results in discrete-time simulations for inputs which cannot be evaluated by frequency domain techniques.

Appendix A

Error Considerations on the Computational Length of a Discrete-Time Filter

The computational complexity of the response to a discrete-time nonlinear filter is inseparably linked to the degree of error which is acceptable in the response. When a finite impulse response (FIR) realization - or its multidimensional extension - of a filter is chosen, a significant component of the error is due to the truncation of the response. Therefore, control of the error in a response is intimately related to the selection of the quantity and "significance" of the samples which describe the discretized response.

A.1 Bandlimitation of the Filter Response to Prevent Aliasing

Let the response to a linear system be given as:

$$y(t) = x(t) * h(t) \tag{A-1}$$

where $x(t)$ is the input signal, and $h(t)$ is the impulse response of the filter which represents the system. Assume that $x(t)$ is W-bandlimited, i.e., $X(f) = 0$, $|f| > W$, where $X(f)$ is the Fourier transform of $x(t)$.

Since the input is bandlimited, there is no difference between the response as calculated from equation (A-1) and:

$$y(t) = x(t) * h_w(t) \tag{A-2}$$

where:

$$h_w(t) = \int_{-W}^{W} H(f) \exp(j2\pi f t)\, df \tag{A-3}$$

when $H(f)$ is the Fourier transform of $h(t)$. We call $h_w(t)$ the bandlimited impulse response. Failure to properly bandlimit the filter (system) response as in equation (A-3) before discretizing the filter necessarily introduces aliasing into the discrete-time representation.

Since both $x(t)$ and $h_w(t)$ are W-bandlimited, each has a valid sampling expansion. In particular, for the bandlimited filter impulse response:

$$h_w(t) = \sum_{n=-\infty}^{\infty} h_w\left(\frac{n}{2W}\right) \operatorname{sinc}\left[2W\left(t - \frac{n}{2W}\right)\right] \tag{A-4}$$

The input signal, $x(t)$, may be similarly represented, as in equation (3-2).

A.1.1 Frequency Domain Windowing

The choice, $h_w(t)$, for a bandlimited impulse response yields the minimum bandwidth, hence it permits the lowest

possible sampling rate to be applied. However, the abrupt truncation of the frequency response yields discontinuous transitions which will exhibit the overshoot and ripple characteristics of Gibbs' phenomenon [33,34] when the time-domain response is truncated. Consequently, it may be advantageous to apply a multiplicative windowing function in the frequency domain to the continuous-time prototype transfer function, $H(f)$. The result of frequency-domain windowing is the modified transfer function $H_B(f)$ where:

$$H_B(f) = H(f) W_B(f)$$

A window function which will mitigate the Gibbs' phenomemon without compromising the important features of the prototype transfer function satisfies:

$$W(f) = \begin{cases} 1, & |f| \le W \\ T(f), & W \le |f| \le W + \Delta W \\ 0, & |f| > W + \Delta W \end{cases} \qquad \text{(A-5)}$$

where $T(f)$ is a bounded continuous function such that $T(|W|) = 1$ and $T(|W+\Delta W|) = 0$.

The impulse response associated with the transfer function $H_B(f)$ is $h_b(t)$ which may be expressed as:

$$h_b(t) = \int_{-B}^{B} H(f) W_B(f) \exp(j2\pi f t) df \qquad \text{(A-6)}$$

The Nyquist bandwidth of the frequency-domain windowed system response is $B = W + \Delta W$.

A.2 Response Energy

We shall assume that the prototype analog filter impulse response has finite energy:

$$E_H = \int_{-\infty}^{\infty} |h(t)|^2 dt = \int_{-\infty}^{\infty} \left| H(f) \right|^2 df < \infty$$

Then, the W-bandlimited filter impulse response must also have finite energy, which can be expressed in three forms:

$$E_{H,W} = \int_{-W}^{W} \left| H(f) \right|^2 df \qquad \text{(A-7a)}$$

$$= \int_{-\infty}^{\infty} |h_w(t)|^2 dt \qquad \text{(A-7b)}$$

$$= \frac{1}{2W} \sum_{n=-\infty}^{\infty} \left| h_w\left(\frac{n}{2W}\right) \right|^2 \qquad \text{(A-7c)}$$

This final form holds because the sampling basis functions, $\text{sinc}\left[2W\left(t - \frac{n}{2W}\right)\right]$, are orthogonal [27].

In cases where a frequency-domain windowing function is employed, we have:

$$E_{H,B} = \int_{-B}^{B} \left| H\left(f\right) W_B\left(f\right) \right|^2 df \qquad \text{(A-8a)}$$

$$= \int_{-\infty}^{\infty} |h_b(t)|^2 dt \qquad \text{(A-8b)}$$

$$= \frac{1}{2B} \sum_{n=-\infty}^{\infty} \left| h_b\left(\frac{n}{2B}\right) \right|^2 \qquad \text{(A-8c)}$$

In the succeeding sections, we present the developments only for $h_w(t)$; however, one may directly substitute $h_b(t)$ or other appropriate windowed-response form in the event that a frequency-domain window has been applied.

A.3 Response Truncation

In order to utilize the representation of a system which is suggested by equation (A-4) in a computable discrete-time realization, it is necessary to abbreviate the infinite summation of terms. The finite result which we obtain is a truncated sampling expansion[1], i.e.:

$$h_{w,N_1,N_2}(t) = \sum_{n=N_1}^{N_2} h_w\left(\frac{n}{2W}\right) \operatorname{sinc}\left[2W\left(t - \frac{n}{2W}\right)\right] \qquad \text{(A-9)}$$

---

[1] Neither the signal approximation nor the error caused by truncating the sampling expansion is duration limited; consequently, both the truncated sampling expansion and the error are bandlimited to the frequency interval occupied by the bandlimited analog prototype filter response.

Using equations (A-4) and (A-9), we can describe the error inherent to the truncated sampling expansion:

$$h_e(t) = h_w(t) - h_{w,N_1,N_2}(t) = \sum_{\{n \mid n<N_1 \text{ or } n>N_2\}} h_w\left(\frac{n}{2W}\right) \operatorname{sinc}\left[2W\left(t-\frac{n}{2W}\right)\right] \quad \text{(A-10)}$$

The energy associated with the error signal given in equation (A-10) is:

$$E_e = \frac{1}{2W} \sum_{\{n \mid n<N_1 \text{ or } n>N_2\}} \left|h_w\left(\frac{n}{2W}\right)\right|^2 \quad \text{(A-11)}$$

An objective of discrete-time processing is to reduce the error represented by equation (A-11) below some threshold using the smallest practical number of terms in the sampling expansion. Optimization cannot, in general, be obtained in closed form. Nevertheless, equation (A-11) can be used to verify that for a particular number of terms the impulse response error energy is less than a prescribed limit. If the terms are selected based on a knowledge of the specific characteristics of a system, demonstration that the number of terms is the minimum may be possible.

It may also be observed that the abrupt truncation of the sampling expansion of the impulse response may introduce significant distortion into the bandlimited transfer function. Consequently, it may be desirable to apply a time-domain window to the sequence of sample values

as a means of reducing the degradation to the transfer function. When a time-domain window is applied, corresponding adjustments will be required to equations (A-10) and (A-11).

Consider a window function of the form:

$$w_{N_1,N_2}\left(\frac{n}{2W}\right)=\begin{cases} 0, & n<N_1 \\ \xi\left(\frac{n}{2W}\right), & N_1 \le n \le N_2 \\ 0, & n>N_2 \end{cases} \qquad \text{(A-12)}$$

where $\xi(n/2B)$ is an arbitrary, bounded function of n, chosen to minimize the transitions of $h_w(t)$ at $n=N_1, N_2$. The resulting truncation error (compare with equation (A-10)) is:

$$h_e(t)=h_w(t)-\sum_{n=N_1}^{N_2} h_w\left(\frac{n}{2W}\right)\xi\left(\frac{n}{2B}\right)\operatorname{sinc}\left[2W\left(t-\frac{n}{2W}\right)\right] \qquad \text{(A-13)}$$

The energy associated with the error signal in equation (A-13) is:

$$E_e=\frac{1}{2W}\sum_{n=-\infty}^{\infty}\left|\left[1-w_{N_1,N_2}\left(\frac{n}{2W}\right)\right]h_w\left(\frac{n}{2W}\right)\right|^2 \qquad \text{(A-14)}$$

In order to minimize the error energy, it may be desirable to choose a window function, $\xi(n/2B)$, which is identically 1

over a central range of values of n and is tapered to 0 outside this range.

A.4 Absolute Error Bound

Using the truncation error bound given by Papoulis [27], the maximum absolute error in the truncated, bandlimited filter impulse response is bounded by:

$$|h_e(t)| \leq \sqrt{2WE_e} \tag{A-15}$$

A.5 Response Error Bound

Ordinarily, it is more relevant to discuss the error introduced in the system response than the bound on the error in representing the system impulse response. For a linear system described in the form of equation (A-9), assume that the input, $x(t)$, is bounded by some real number, β, such that:

$$|x(t)| \leq \beta \tag{A-16}$$

Stability requires that the system impulse response be absolutely integrable. Let us assume that:

$$\int_{-\infty}^{\infty} |h_w(t)|dt = S < \infty \tag{A-17}$$

Then the response can be decomposed into approximation and error terms:

$$y(t) = \hat{y}(t) + y_e(t) \tag{A-18a}$$

$$y(t) = h_{w,N_1,N_2}(t) * x(t) + h_e(t) * x(t) \tag{A-18b}$$

where $h_{w,N_1,N_2}(t)$ and $h_e(t)$ are defined in equations (A-9) and (A-10), respectively. Since the system impulse response is, by assumption, absolutely integrable, the error component must also be absolutely integrable:

$$\int_{-\infty}^{\infty} |h_e(t)| = S_e < S \tag{A-19}$$

Therefore, the response error term can be bounded as follows:

$$\begin{aligned} |y_e(t)| &= \left| \int_{-\infty}^{\infty} h_e(\tau) x(t-\tau) \right| d\tau \\ &\leq \int_{-\infty}^{\infty} |h_e(\tau)| \, |x(t-\tau)| d\tau \\ &\leq \int_{-\infty}^{\infty} |h_e(\tau)| \beta \, d\tau \\ &= S_e \beta \end{aligned} \tag{A-20}$$

The relative significance of the error signal amplitude is on the order of S_e/S. As a matter of

practicality, however, it may be difficult to explicitly determine the values of either S or S_e since, in general, a closed form expression for $h_w(t)$ may not be obtainable. In this case, the energy bound of the system (impulse) response established in section A.3 may have to be relied upon as a relative indicator of the response error significance.

A.6 An Example of the Application of Bandlimitation and Truncation

It is difficult to work with many analytically described bandlimited transfer functions since their inverse transforms cannot be expressed in closed form. It is illustrative, therefore, to examine the bandlimitation and subsequent sampling-expansion truncation of an ideal lowpass filter (i.e. a brick wall filter). Although there is no processing utility to lowpass filtering a signal of lesser bandwidth, one may consider a filter which introduces a fixed delay in the signal path as might be done as a part of a signal equalization network. For this specially constructed case, it is possible to obtain an inverse transform for the bandlimited filter and the partial energy terms can be expressed in terms of $\mathrm{sinc}(\,)$ functions.

Let $H(f)$ be an ideal lowpass filter with bandwidth C Hertz and delay T_0. The filter transfer function is:

$$H(f) = \begin{cases} \exp(-j2\pi f T_0), & |f| \le C \\ 0, & |f| > C \end{cases} \tag{A-21}$$

Assume that the class of inputs which are of interest for discrete-time processing are bandlimited to A Hertz, such that $A < C$. Let $H(f)$ be windowed by a trapezoidal spectrum mask having the characteristic:

$$W_B(f) = \begin{cases} 1, & |f| \le A \\ \frac{B-|f|}{B-A}, & A \le |f| \le B \\ 0, & |f| \ge B \end{cases} \tag{A-22}$$

where B is any frequency which satisfies $A \le B \le C$. For the limiting case, $A = B$, this window represents abrupt truncation. At the other extreme, $A = C$, the window is so broad that no reduction in sampling rate (without aliasing) is possible. Then,

$$H_B(f) = H(f)W_B(f) = \begin{cases} \exp(-j2\pi f T_0), & |f| \le A \\ \frac{B-|f|}{B-A}\exp(-j2\pi f T_0), & A \le |f| \le B \\ 0, & |f| \ge B \end{cases} \tag{A-23}$$

The transfer function magnitude is shown in Figure A.1.

Taking the inverse Fourier transform of equation (A-23), the impulse response of the bandlimited transfer function is found to be:

$$h_b(t) = (A+B)\{\text{sinc}[(A+B)(t-T_0)]\,\text{sinc}[(A-B)(t-T_0)]\} \qquad \text{(A-24)}$$

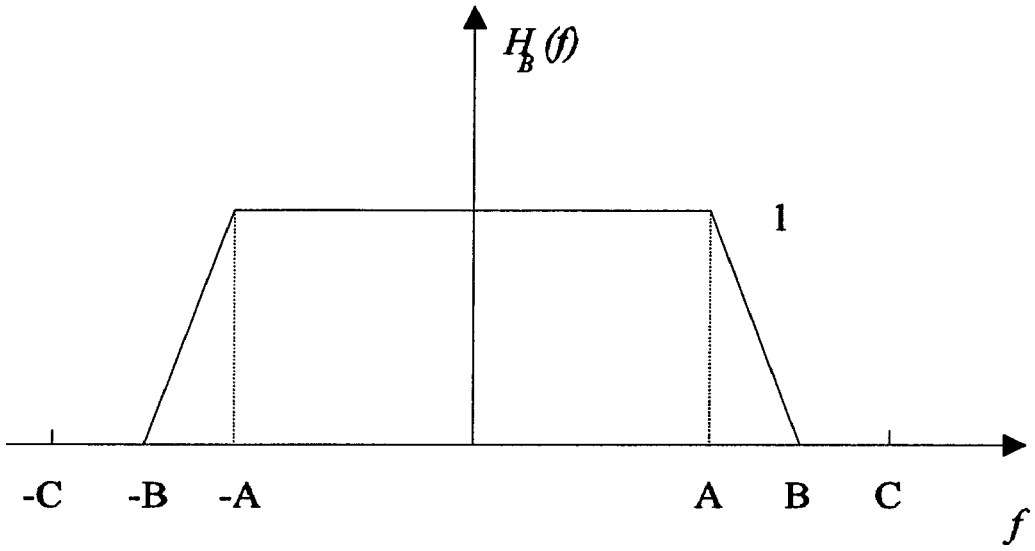


Figure A.1: Magnitude Spectrum of $H_B(f)$

In the limit, as B approaches A, equation (A-24) degenerates to the (delayed) ideal lowpass filter form:

$$\lim_{B\to A} h_b(t) = 2A\,\text{sinc}[2A(t-T_0)] \qquad \text{(A-25)}$$

The windowed, bandlimited impulse response, equation (A-24), can be sampled, without error, at instants $t = n/2B$ to obtain its Nyquist samples:

$$h_b\left(\frac{n}{2B}\right) = (A+B)\,\mathrm{sinc}\left[(A+B)\left(\frac{n}{2B} - T_0\right)\right]\mathrm{sinc}\left[(A-B)\left(\frac{n}{2B} - T_0\right)\right] \qquad \text{(A-26)}$$

The energy of the bandlimited impulse response is found by direct integration after substitution of equation (A-23) into equation (A-8a). We obtain:

$$E_{H,B} = \frac{2}{3}(2A+B) \qquad \text{(A-27)}$$

When A, B, and T_0 are specified, we can determine values of N_1 and N_2 such that the value of the truncation error energy (equation (A-11)) is held below a predetermined threshold. Since the total impulse response energy, equation (A-27), is known and the truncation error is orthogonal to the truncated sampling expansion [27], the truncation error energy, equation (A-11), can be expressed as:

$$E_e = E_{H,B} - \frac{1}{2B}\sum_{n=N_1}^{N_2}\left|h_b\left(\frac{n}{2B}\right)\right|^2 \qquad \text{(A-28)}$$

Assume that we want to reduce E_e to a level 20 dB below $E_{H,B}$. Then we require:

$$E_e \leq 0.01 E_{H,B} \tag{A-29}$$

Using equations (A-27) and (A-29) in equation (A-28), we obtain:

$$\sum_{n=N_1}^{N_2} \left| h_b\left(\frac{n}{2B}\right) \right|^2 \geq 2B(0.99)\frac{2}{3}(2A+B) = (1.32)B(2A+B) \tag{A-30}$$

For this example let us choose $B = 1.5A$ and $T_0 = 0$. Then, choosing $N_1 = -N_2$, since $h_b(t)$ is symmetric for $T_0 = 0$ (or T_0 equal to any integer multiple of the sampling interval), we obtain $N_2 = 3$. The truncation error is $0.0026 E_{H,B}$ or 25.8 dB below the total response energy. Thus, a seven-sample symmetric approximation to $h_b(t)$ suffices to include 99 percent of the impulse response energy[2].

Applying equation (A-15) shows that the absolute error of the impulse response must be bounded by:

$$|h_e(t)| \leq \sqrt{2BE_e} = \frac{2}{3}\sqrt{7}\,B = \sqrt{7}\,A \tag{A-31}$$

---

[2] If an asymmetric approximation is acceptable in this case, a six-sample approximation will meet the 99 percent impulse response energy containment criterion with an error of $0.0085 E_{H,B}$.

The error introduced in the frequency response by truncating the time-domain impulse resonse sampling expansion may be seen by taking the Fourier transform of the seven-sample approximation. This yields:

$$H_{B,-3,3}\left(e^{j\omega}\right) = h_{d,b}(0) + 2h_{d,b}(1)\cos(\omega) + 2h_{d,b}(2)\cos(2\omega) + 2h_{d,b}(3)\cos(3\omega) \qquad \text{(A-32)}$$

where $h_{d,b}(n) = \frac{1}{2B}h_b\left(\frac{n}{2B}\right)$. When the values of the $h_{d,b}(n)$ are computed, equation (A-32) becomes:

$$H_{B,-3,3}\left(e^{j\omega}\right) = \frac{5}{6} + \frac{3}{2\pi^2}\cos(\omega) - \frac{9}{8\pi^2}\cos(2\omega) + \frac{2}{3\pi^2}\cos(3\omega) \qquad \text{(A-33)}$$

The spectrum indicated by equation (A-33) is plotted in figure A-2.

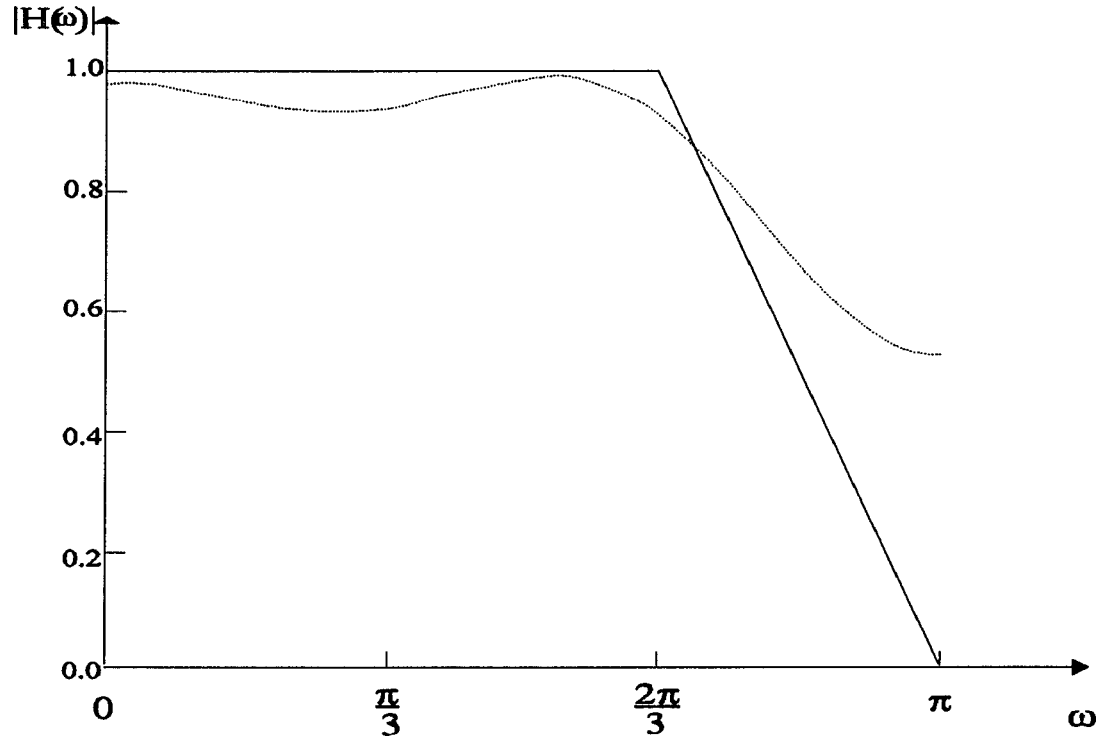


Figure A-2: Seven-sample Approximation Filter Magnitude Spectrum

It may be seen from the figure that the abrupt truncation of the impulse response sampling sequence results in a significantly greater distortion of the frequency response than suggested by the 1 percent loss of impulse response energy. This effect may be mitigated by applying a window to the sampling sequence to effect a smoother transition in the time domain which will, in turn, reduce the distortion to the frequency response.

A.7 *Application of the Truncation Error Control Procedure to the Circuit Example*

In the nonlinear circuit example of section 2.5.2.1, the associated linear circuit had an impulse response given by:

$$h(t) = \frac{1}{C}\exp[-kt]u(t) \qquad \text{(A-34a)}$$

The corresponding transfer function is:

$$H(f) = \frac{1}{C}\frac{1}{j2\pi f + k} \qquad \text{(A-34b)}$$

We will develop the appropriate truncation limits for a bandlimited filter with the same passband characteristic as expressed in equation (A-34b). We will follow the same approach as in the previous section.

Let $H(f)$ be windowed in the frequency domain by the trapezoidal spectrum mask given in equation (A-22). Then, we obtain:

$$H_B(f) = \begin{cases} 0, & f < -B \\ \frac{1}{C}\frac{1}{j2\pi f + k}\frac{f+B}{B-A}, & -B \le f \le -A \\ \frac{1}{C}\frac{1}{j2\pi f + k}, & -A \le f \le A \\ \frac{1}{C}\frac{1}{j2\pi f + k}\frac{B-f}{B-A}, & A \le f \le B \\ 0, & f > B \end{cases} \qquad \text{(A-35)}$$

The bandlimited impulse response energy can be obtained in closed form by integration. We obtain:

$$E_{H,B} = \int_{-B}^{B} \left| H_B(f) \right|^2 df$$

$$= \frac{1}{C^2}\left\{ \tan^{-1}\left(\frac{2\pi A}{k}\right)\left[\frac{1}{\pi k} - \frac{2B^2}{(B-A)^2 2\pi k} + \frac{2k}{(B-A)^2(2\pi)^3}\right] + \tan^{-1}\left(\frac{2\pi B}{k}\right)\left[\frac{2B^2}{(B-A)^2 2\pi k} - \frac{2k}{(B-A)^2(2\pi)^3}\right] \right.$$

$$\left. + \frac{2B}{(B-A)^2(2\pi)^2}\left\{\ln\left[\left(\frac{k}{2\pi}\right)^2 + A^2\right] - \ln\left[\left(\frac{k}{2\pi}\right)^2 + B^2\right]\right\} + \frac{2}{(B-A)(2\pi)^2}\right\} \qquad \text{(A-36)}$$

The impulse response associated with the bandlimited transfer function, as given in equation (A-35) cannot be obtained in closed form. However, since we only require values (samples) of the impulse response at distinct instants of time, e.g. $t = n/2B$, we may obtain these to any desired degree of accuracy by numerical integration.

The inversion integral which defines $h_b(t)$ may be written using equation (A-35) as:

$$h_b(t) = \int_{-B}^{-A} \frac{1}{j2\pi f + k}\frac{f+B}{B-A} e^{j2\pi ft} df + \int_{-A}^{A} \frac{1}{j2\pi f + k} e^{j2\pi ft} df + \int_{A}^{B} \frac{1}{j2\pi f + k}\frac{B-f}{B-A} e^{j2\pi ft} df \qquad \text{(A-37)}$$

Since $H_B(f)$ is antisymmetric, $h_b(t)$ will be real. Accordingly, expansion and combination of the various terms in equation

(A-37) will yield a simplification prior to application of numerical integration. We obtain:

$$h_b(t)=\int_0^A\left[\frac{2k}{C}\frac{\cos\left(2\pi ft\right)}{k^2+\left(2\pi f\right)^2}+\frac{4\pi f}{C}\frac{\sin\left(2\pi ft\right)}{k^2+\left(2\pi f\right)^2}\right]df+\int_A^B\left[\frac{2k}{C}\frac{\cos\left(2\pi ft\right)}{k^2+\left(2\pi f\right)^2}+\frac{4\pi f}{C}\frac{\sin\left(2\pi ft\right)}{k^2+\left(2\pi f\right)^2}\right]\frac{B-f}{B-A}df \qquad \text{(A-38)}$$

Establishing a bound on the error of the Fourier integral inversion by numerical integration using the trapezoidal integration formula requires that we establish a bound on the magnitude of the second derivative(s) of the integrands [35]. Labelling the first and second integrands, respectively, $H_{B,1}(f)$ and $H_{B,2}(f)$, we obtain the following derivatives:

$$H'_{B,1}\left(f\right)=\frac{4\pi}{C}\cos\left(2\pi ft\right)\frac{2\pi\left[k^2\left(2\pi f\right)t+\left(2\pi f\right)^3t-2k\left(2\pi f\right)\right]}{\left[k^2+\left(2\pi f\right)^2\right]^2}+\frac{4\pi}{C}\sin\left(2\pi ft\right)\frac{k^2-k^3t-k\left(2\pi f\right)^2t-\left(2\pi f\right)^2}{\left[k^2+\left(2\pi f\right)^2\right]^2} \qquad \text{(A-39)}$$

and

$$H''_{B,1}\left(f\right)=\frac{4\pi}{C}\cos\left(2\pi ft\right)\left\{\frac{2\pi\left[\left(-2t-kt^2\right)\left(2\pi f\right)^4+\left(6k-2k^3t^2\right)\left(2\pi f\right)^2+2k^4t-2k^3-k^5t^2\right]}{\left[k^2+\left(2\pi f\right)^2\right]^3}\right\} \qquad \text{(A-40)}$$

$$+\frac{8\pi^2\left(2\pi f\right)}{C}\sin\left(2\pi ft\right)\left\{\frac{-t^2\left(2\pi f\right)^4+2\left(1-k^2t^2+2kt\right)\left(2\pi f\right)^2+4k^3t-6k^2-k^4t^2}{\left[k^2+\left(2\pi f\right)^2\right]^3}\right\}$$

Recognizing that the second integrand has the form:

$$H_{B,2}(f) = H_{B,1}(f)T(f) \tag{A-41}$$

where:

$$T(f) = \frac{B-f}{B-A} \tag{A-42}$$

we obtain:

$$H''_{B,2}(f) = H''_{B,1}(f)T(f) + 2H'_{B,1}(f)T'(f) + H_{B,1}(f)T''(f) \tag{A-43}$$

Evaluating the derivatives of $T(f)$ we obtain:

$$T'(f) = \frac{-1}{B-A} \qquad T''(f) = 0 \tag{A-44}$$

Then, equation (A-43) becomes:

$$H''_{B,2}(f) = \frac{B-f}{B-A}H''_{B,1}(f) - \frac{2}{B-A}H'_{B,1}(f) \tag{A-45}$$

For specific sets of the parameters $t, k, f, A, B,$ and $C,$ magnitude bounds on the integrand second derivatives can be obtained in each numerical integration sub-interval. Consequently, for each sample obtained for $h_b(t)$, an associated error bound can also be derived. The resulting

approximations to the bandlimited impulse response samples are given as:

$$\hat{h}_b\left(\frac{n}{2B}\right) = \sum_{l=1}^{L} \frac{\Delta f_l}{2}\left[H_{B,1}\left((l-1)\Delta f_l\right) + H_{B,1}\left(l\Delta f_l\right)\right]_{t=\frac{n}{2B}} + \sum_{m=1}^{M} \frac{\Delta f_m}{2}\left[H_{B,2}\left(A+(m-1)\Delta f_m\right) + H_{B,2}\left(A+m\Delta f_m\right)\right]_{t=\frac{n}{2B}} \quad \text{(A-46)}$$

where $\Delta f_l = A/L$ and $\Delta f_m = (B - A)/M$. Their associated error bounds are:

$$\left|e_b\left(\frac{n}{2B}\right)\right| \le \sum_{l=1}^{L} \left|\frac{\left(\Delta f_l\right)^3}{12} H''_{B,1}\left(\eta_l\right)\right| + \sum_{m=1}^{M} \left|\frac{\left(\Delta f_m\right)^3}{12} H''_{B,2}\left(\eta_m\right)\right| \quad \text{(A-47)}$$

where η_l and η_m are the frequency argument values within the applicable intervals for which the magnitude of the appropriate second derivative is maximized for the specific time instant, $t = n/2B$, at which the sample value of $h_b(t)$ is computed.

The specific values of L and M which are used to define the sub-interval size of the numerical integration may be chosen to preserve a predetermined accuracy in the determination of each sample value. Iteration may be required to obtain the required accuracy; however, the process of determining the filter coefficients is only necessary one time. Once determined, the filter

coefficients remain unchanged for the duration of discrete-time processing.

As an example of the results obtained using equations (A-46) and (A-47), the coeffiecients of a discrete-time filter were obtained for the parameter set:

Frequency break-point	A	2000. Hz
Nyquist frequency	B	3500. Hz
Inverse time constant	k	1200. sec^{-1}
Capacitance	C	normalized 1 f

Note that for the windowed transfer function, the impulse response is no longer strictly causal as the analog prototype, equation (A-34a) was. Table A-1 shows the coefficient values, their respective numerical integration error bounds, and the cumulative fraction of the total bandlimited impulse response energy represented by the coefficients. The number of steps in the numerical integration procedure was set at $L = M = 10{,}000$.

Coefficient Number	Coefficient Value	Coefficient Error Bound	Cumulative Coeff. Energy
- 2	-0.0069030	3.096471 E-6	1.717230 E-5
- 1	-0.0473697	2.906999 E-6	8.264408 E-4
0	0.4773606	2.057313 E-7	0.0830199
1	0.9024777	4.058704 E-6	0.3767941
2	0.7110701	6.225556 E-6	0.5591677
3	0.5844567	6.414444 E-6	0.6823760
4	0.5094609	7.672586 E-6	0.7759927
5	0.4269101	8.052886 E-6	0.8417284
6	0.3542025	9.070353 E-6	0.8869791
7	0.3009782	9.831162 E-6	0.9196519
8	0.2557755	1.054984 E-5	0.9432473
9	0.2130949	1.160481 E-5	0.9596246
10	0.1790633	1.216838 E-5	0.9711883
11	0.1525623	1.332188 E-5	0.9795822
12	0.1282104	1.394066 E-5	0.9855100
13	0.1069255	1.497328 E-5	0.9896327
14	0.0907252	1.580297 E-5	0.9926006
15	0.0769961	1.662661 E-5	0.9947380
16	0.0641472	1.770609 E-5	0.9962214
17	0.0539005	1.837307 E-5	0.9972687

18	0.0460186	1.954778 E-5	0.9980319
19	0.0386485	2.025542 E-5	0.9985701
20	0.0321061	2.137101 E-5	0.9989414
21	0.0273404	2.229762 E-5	0.9992105
22	0.0232769	2.322558 E-5	0.9994056
23	0.0192786	2.438909 E-5	0.9995393
24	0.0161628	2.516876 E-5	0.9996333
25	0.0139382	2.642932 E-5	0.9997030
26	0.0116672	2.724396 E-5	0.9997519
27	0.0095944	2.845249 E-5	0.9997849
28	0.0082360	2.947604 E-5	0.9998093
29	0.0070862	3.050417 E-5	0.9998271
30	0.0057673	3.176054 E-5	0.9998390
31	0.0048219	3.264365 E-5	0.9998473

Table A-1: Bandlimited Filter Design Example

Since the thrust of our determination of the required filter coefficients is to bound the error, the energy contribution of each coefficient was determined on a minimum basis. That is, the energy contribution of each coefficient was determined by reducing the calculated magnitude by the amount of the calculated coefficient error

bound. While the number of numerical integration steps chosen was large, this is inevitable for applications which require high accuracy.

The required number of coefficients for any desired accuracy can quickly be determined. (If a causal approximation is desired, the cumulative energy of the $n=-1$ and $n=-2$ coefficients should be subtracted from the cumulative energy value shown for the appropriate coefficient.) The $n=-1$ and $n=-2$ coefficients are small; however, if an application limits truncation error to less than approximately 0.4% of the impulse response energy, it may be advantageous to include the $n=-1$ term before the $n=18$ and subsequent terms. If it is required that truncation error energy be greater than 31 dB down, it will be necessary to include one or more terms for $n<0$.

APPENDIX B

Viewing the Discrete Fourier Transform as a Numerical Integration

The discrete Fourier transform is equivalent to a trapezoidal numerical integration procedure under very mild assumptions regarding the form of the integrand. We show this below for the inverse Fourier transform, which was used (see Chapter 7) to obtain the Volterra kernels from frequency-domain windowed transfer functions.

Let a desired continuous-time function be given as the inverse Fourier transform of a bandlimited frequency domain transfer function:

$$x(t) = \int_{-W}^{W} X(f) \exp(j2\pi f t)\, df \qquad \text{(B-1)}$$

Since the time-domain function is bandlimited, if only because the transform is truncated at $|f| = W$, it can be completely described by the values of its samples at instants of time, $t = n/2W$. At these instants of time, using a trapezoidal approximation to the integral with N increments, we obtain:

$$x\left(\frac{n}{2W}\right) \approx \Delta f \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} \frac{1}{2}\left\{X[k\Delta f] \exp\left[j2\pi k \Delta f \frac{n}{2W}\right] + X[(k+1)\Delta f] \exp\left[j2\pi (k+1)\Delta f \frac{n}{2W}\right]\right\} \qquad \text{(B-2)}$$

where $\Delta f = \frac{2W}{N}$.

Since the second term of the k^{th} summand is the first term of the $k+1^{st}$ summand (except the first term of the first summand and the second term of the last summand), we may write equation (B-2) as:

$$x\left(\frac{n}{2W}\right) \approx \frac{2W}{N}\frac{1}{2}X(-W)\exp\left(-j\pi n\right) + \frac{2W}{N}\sum_{k=-\frac{N}{2}+1}^{\frac{N}{2}-1} X\left(\frac{2kW}{N}\right)\exp\left(j\frac{2\pi}{N}kn\right)$$

$$+\frac{2W}{N}\frac{1}{2}X(W)\exp\left(j\pi n\right) \qquad \text{(B-3)}$$

Recognizing that $\exp\left(-j\pi n\right) = \exp\left(j\pi n\right)$, if $X(-W) = X(W)$, then equation (B-3) reduces to:

$$x\left(\frac{n}{2W}\right) \approx \frac{2W}{N}\sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} X\left(\frac{2kW}{N}\right)\exp\left(j\frac{2\pi}{N}kn\right) \qquad \text{(B-4)}$$

This assumption is clearly valid if we force $X(f) = 0$ at $|f| = W$. In the event that this assumption is not satisfied, the error will diminished as N increases.

If we further assume that $X(f)$ is periodic in $2W$,then we may also write in place of equation (B-4):

$$x\left(\frac{n}{2W}\right) \approx \frac{2W}{N}\sum_{k=0}^{N-1} X\left(\frac{2kW}{N}\right)\exp\left(j\frac{2\pi}{N}kn\right) \qquad \text{(B-5)}$$

Defining: $\tilde{x}(n) = x\left(\frac{n}{2W}\right)$ and $\hat{X}(k) = X\left(\frac{2kW}{N}\right)$ equation (B-5), when expressed as an equality is the inverse discrete Fourier transform multiplied by the constant $2W$.

APPENDIX C

Volterra Filter Coefficients

The calculated filter coefficients for the linear (first-order) filter of the Bandlimited Volterra series realization used to obtain the results in Chapter 7 are listed in Table 1. The bandwidth of the filter is 1000 Hertz and the sampling interval is 1/2000 second.

Coeff. Number	Coefficient Value
-3	-24.69
-2	35.83
-1	-64.49
0	351.95
1	514.41
2	201.37
3	158.78
4	52.68
5	55.72
6	8.64
7	23.31
8	-3.31
9	12.39

Table 1: Bandlimited Volterra Series Realization First-Order Filter Coefficients

All of the linear filter coefficients were computed with the Romberg integration tolerance set to 0.001 times the final extrapolation estimate.

The calculated filter coefficients for the linear (first-order) filters used in the Direct and Serial Volterra series realizations are listed in Table 2. The bandwidth of the Direct realization filter is 1000 Hertz and the sampling interval is 1/6000 second. The bandwidth of the Serial realization filter is 3000 Hertz and the sampling interval is 1/6000 second. Each filter has 31 coefficients. The seven order-of-magnitude difference between the coefficient values is due to the inclusion of the a1 factor in the direct realization filter; the Serial realization filter is utilized in three separate parts of the Serial realization model where it is multiplied by appropriate (but different) scaling factors.

Coefficient Number	Direct Realization Coefficient Value	Serial Realization Coefficient Value
-9	- 24.69	-1.114 E+8
-8	- 6.58	1.251 E+8
-7	24.10	-1.428 E+8
-6	35.83	1.662 E+8
-5	8.00	-1.987 E+8
-4	- 42.79	2.470 E+8

-3	- 64.49	-3.257 E+8
-2	- 3.67	4.760 E+8
-1	151.21	-8.690 E+8
0	351.95	4.800 E+9
1	514.88	9.103 E+9
2	572.41	6.211 E+9
3	514.41	5.822 E+9
4	390.04	4.242 E+9
5	270.35	3.881 E+9
6	201.37	2.844 E+9
7	181.93	2.610 E+9
8	178.31	1.893 E+9
9	158.78	1.765 E+9
10	118.24	1.252 E+9
11	75.83	1.200 E+9
12	52.68	8.229 E+8
13	52.10	8.205 E+8
14	58.97	5.359 E+8
15	55.72	5.653 E+8
16	38.37	3.444 E+8
17	18.07	3.932 E+8
18	8.64	2.171 E+8
19	13.16	2.769 E+8
20	22.07	1.326 E+8
21	23.31	1.981 E+8

Table 2: First-Order Volterra Filter Coefficients for the Direct and Serial Realizations

The calculated filter coefficients for the linear filter used in the Picard iteration realization of the Volterra series are listed in Table 3. The bandwidth of the 9000 Hertz and the sampling interval is 1/18000 second.

Coeff. Number	Coefficient Value	Coeff. Number	Coefficient Value	Coeff. Number	Coefficient Value
-4	2.494 E8	16	3.378 E9	36	8.790 E8
-3	-3.294 E8	17	3.279 E9	37	8.761 E8
-2	4.829 E8	18	2.956 E9	38	7.673 E8
-1	-8.868 E8	19	2.871 E9	39	7.687 E8
0	4.932 E9	20	2.585 E9	40	6.695 E8
1	1.026E10	21	2.514 E9	41	6.747 E8
2	8.263 E9	22	2.261 E9	42	5.840 E8
3	8.519 E9	23	2.202 E9	43	5.924 E8
4	7.408 E9	24	1.977 E9	44	5.092 E8
5	7.367 E9	25	1.929 E9	45	5.204 E8
6	6.535 E9	26	1.728 E9	46	4.437 E8
7	6.415 E9	27	1.691 E9	47	4.573 E8
8	5.740 E9	28	1.510 E9	48	3.865 E8
9	5.601 E9	29	1.482 E9	49	4.020 E8
10	5.033 E9	30	1.320 E9	50	3.365 E8
11	4.895 E9	31	1.299 E9	51	3.536 E8
12	4.409 E9	32	1.153 E9	52	2.927 E8
13	4.281 E9	33	1.139 E9	53	3.112 E8
14	3.860 E9	34	1.007 E9	54	2.545 E8
15	3.746 E9	35	9.987 E8	55	2.740 E8

Table 3: Picard Iteration Filter Coefficients

All of the second-order Bandlimited filter coefficients were computed with the Romberg integration tolerance set to 0.001 times the final extrapolation estimate.

The second-order coefficients for the Bandlimited Volterra series realization are shown in Table 4. The Table is structured in an array format which may be used for computation (although since the i,j term is identical to the j,i term, this is not the most computationally efficient method). The top left element of the array is the $h_2(0,0)$ coefficient.

$$h2 = \begin{bmatrix} 1.264\cdot10^5 & 1.898\cdot10^5 & 5.04\cdot10^4 & 5.627\cdot10^4 & 9.256\cdot10^3 & 2.048\cdot10^4 & -1.749\cdot10^3 \\ 1.899\cdot10^5 & 8.468\cdot10^5 & 5.985\cdot10^5 & 3.093\cdot10^5 & 1.789\cdot10^5 & 9.498\cdot10^4 & 5.422\cdot10^4 \\ 5.04\cdot10^4 & 5.985\cdot10^5 & 8.443\cdot10^5 & 4.897\cdot10^5 & 2.714\cdot10^5 & 1.43\cdot10^5 & 8.303\cdot10^4 \\ 5.627\cdot10^4 & 3.093\cdot10^5 & 4.911\cdot10^5 & 5.335\cdot10^5 & 3.317\cdot10^5 & 1.774\cdot10^5 & 1.003\cdot10^5 \\ 9.256\cdot10^3 & 1.789\cdot10^5 & 2.714\cdot10^5 & 3.317\cdot10^5 & 3.391\cdot10^5 & 1.874\cdot10^5 & 1.069\cdot10^5 \\ 2.048\cdot10^4 & 9.498\cdot10^4 & 1.43\cdot10^5 & 1.774\cdot10^5 & 1.879\cdot10^5 & 1.838\cdot10^5 & 1.154\cdot10^5 \\ -1.749\cdot10^3 & 5.422\cdot10^4 & 8.303\cdot10^4 & 1.003\cdot10^5 & 1.069\cdot10^5 & 1.154\cdot10^5 & 1.124\cdot10^5 \end{bmatrix}$$

Table 4: Second-Order Bandlimited Filter Coefficients

The second-order coefficients for the Direct Volterra series realization are shown in Table 5. The table is broken into two parts, the first part including the first

eight columns of the $h_2(i,j)$ array and the second part including the remaining seven columns. All of the second-order coefficients were computed with a Romberg integration relative tolerance of 0.001.

$$h2 = \begin{bmatrix}
9.451\cdot10^4 & 1.605\cdot10^5 & 1.93\cdot10^5 & 1.799\cdot10^5 & 1.347\cdot10^5 & 8.573\cdot10^4 & 5.622\cdot10^4 & 5.065\cdot10^4 \\
1.605\cdot10^5 & 3.17\cdot10^5 & 4.292\cdot10^5 & 4.517\cdot10^5 & 3.874\cdot10^5 & 2.823\cdot10^5 & 1.92\cdot10^5 & 1.465\cdot10^5 \\
1.93\cdot10^5 & 4.292\cdot10^5 & 6.319\cdot10^5 & 7.203\cdot10^5 & 6.726\cdot10^5 & 5.346\cdot10^5 & 3.849\cdot10^5 & 2.834\cdot10^5 \\
1.799\cdot10^5 & 4.517\cdot10^5 & 7.203\cdot10^5 & 8.842\cdot10^5 & 8.918\cdot10^5 & 7.678\cdot10^5 & 5.91\cdot10^5 & 4.406\cdot10^5 \\
1.347\cdot10^5 & 3.874\cdot10^5 & 6.726\cdot10^5 & 8.918\cdot10^5 & 9.732\cdot10^5 & 9.095\cdot10^5 & 7.552\cdot10^5 & 5.872\cdot10^5 \\
8.573\cdot10^4 & 2.823\cdot10^5 & 5.346\cdot10^5 & 7.678\cdot10^5 & 9.095\cdot10^5 & 9.26\cdot10^5 & 8.358\cdot10^5 & 6.926\cdot10^5 \\
5.622\cdot10^4 & 1.92\cdot10^5 & 3.849\cdot10^5 & 5.91\cdot10^5 & 7.552\cdot10^5 & 8.358\cdot10^5 & 8.225\cdot10^5 & 7.367\cdot10^5 \\
5.065\cdot10^4 & 1.465\cdot10^5 & 2.834\cdot10^5 & 4.406\cdot10^5 & 5.872\cdot10^5 & 6.926\cdot10^5 & 7.367\cdot10^5 & 7.163\cdot10^5 \\
5.612\cdot10^4 & 1.379\cdot10^5 & 2.409\cdot10^5 & 3.521\cdot10^5 & 4.594\cdot10^5 & 5.516\cdot10^5 & 6.171\cdot10^5 & 6.443\cdot10^5 \\
5.603\cdot10^4 & 1.367\cdot10^5 & 2.27\cdot10^5 & 3.106\cdot10^5 & 3.816\cdot10^5 & 4.433\cdot10^5 & 4.991\cdot10^5 & 5.441\cdot10^5 \\
4.377\cdot10^4 & 1.196\cdot10^5 & 2.049\cdot10^5 & 2.786\cdot10^5 & 3.309\cdot10^5 & 3.676\cdot10^5 & 4.018\cdot10^5 & 4.403\cdot10^5 \\
2.576\cdot10^4 & 8.62\cdot10^4 & 1.617\cdot10^5 & 2.315\cdot10^5 & 2.808\cdot10^5 & 3.089\cdot10^5 & 3.277\cdot10^5 & 3.508\cdot10^5 \\
1.337\cdot10^4 & 5.377\cdot10^4 & 1.119\cdot10^5 & 1.736\cdot10^5 & 2.236\cdot10^5 & 2.545\cdot10^5 & 2.702\cdot10^5 & 2.824\cdot10^5 \\
1.204\cdot10^4 & 3.82\cdot10^4 & 7.818\cdot10^4 & 1.255\cdot10^5 & 1.7\cdot10^5 & 2.031\cdot10^5 & 2.225\cdot10^5 & 2.329\cdot10^5 \\
1.739\cdot10^4 & 3.988\cdot10^4 & 6.879\cdot10^4 & 1.012\cdot10^5 & 1.332\cdot10^5 & 1.611\cdot10^5 & 1.818\cdot10^5 & 1.949\cdot10^5
\end{bmatrix}$$

Table 5, Part I: Second-Order Volterra Kernel Coefficients for the Direct Realization (Columns 0 - 7)

$$h2 = \begin{bmatrix}
5.612\cdot10^{4} & 5.603\cdot10^{4} & 4.377\cdot10^{4} & 2.576\cdot10^{4} & 1.337\cdot10^{4} & 1.204\cdot10^{4} & 1.739\cdot10^{4} \\
1.379\cdot10^{5} & 1.367\cdot10^{5} & 1.196\cdot10^{5} & 8.62\cdot10^{4} & 5.377\cdot10^{4} & 3.82\cdot10^{4} & 3.988\cdot10^{4} \\
2.409\cdot10^{5} & 2.27\cdot10^{5} & 2.049\cdot10^{5} & 1.617\cdot10^{5} & 1.119\cdot10^{5} & 7.818\cdot10^{4} & 6.879\cdot10^{4} \\
3.521\cdot10^{5} & 3.106\cdot10^{5} & 2.786\cdot10^{5} & 2.315\cdot10^{5} & 1.736\cdot10^{5} & 1.255\cdot10^{5} & 1.012\cdot10^{5} \\
4.594\cdot10^{5} & 3.816\cdot10^{5} & 3.309\cdot10^{5} & 2.808\cdot10^{5} & 2.236\cdot10^{5} & 1.7\cdot10^{5} & 1.332\cdot10^{5} \\
5.516\cdot10^{5} & 4.433\cdot10^{5} & 3.676\cdot10^{5} & 3.089\cdot10^{5} & 2.545\cdot10^{5} & 2.031\cdot10^{5} & 1.611\cdot10^{5} \\
6.171\cdot10^{5} & 4.991\cdot10^{5} & 4.018\cdot10^{5} & 3.277\cdot10^{5} & 2.702\cdot10^{5} & 2.225\cdot10^{5} & 1.818\cdot10^{5} \\
6.443\cdot10^{5} & 5.441\cdot10^{5} & 4.403\cdot10^{5} & 3.508\cdot10^{5} & 2.824\cdot10^{5} & 2.329\cdot10^{5} & 1.949\cdot10^{5} \\
6.258\cdot10^{5} & 5.646\cdot10^{5} & 4.758\cdot10^{5} & 3.813\cdot10^{5} & 3.007\cdot10^{5} & 2.424\cdot10^{5} & 2.028\cdot10^{5} \\
5.646\cdot10^{5} & 5.474\cdot10^{5} & 4.909\cdot10^{5} & 4.093\cdot10^{5} & 3.25\cdot10^{5} & 2.567\cdot10^{5} & 2.1\cdot10^{5} \\
4.758\cdot10^{5} & 4.909\cdot10^{5} & 4.717\cdot10^{5} & 4.182\cdot10^{5} & 3.455\cdot10^{5} & 2.746\cdot10^{5} & 2.195\cdot10^{5} \\
3.813\cdot10^{5} & 4.093\cdot10^{5} & 4.182\cdot10^{5} & 3.975\cdot10^{5} & 3.496\cdot10^{5} & 2.888\cdot10^{5} & 2.313\cdot10^{5} \\
3.007\cdot10^{5} & 3.25\cdot10^{5} & 3.455\cdot10^{5} & 3.496\cdot10^{5} & 3.3\cdot10^{5} & 2.9\cdot10^{5} & 2.406\cdot10^{5} \\
2.424\cdot10^{5} & 2.567\cdot10^{5} & 2.746\cdot10^{5} & 2.888\cdot10^{5} & 2.9\cdot10^{5} & 2.733\cdot10^{5} & 2.409\cdot10^{5} \\
2.028\cdot10^{5} & 2.1\cdot10^{5} & 2.195\cdot10^{5} & 2.313\cdot10^{5} & 2.406\cdot10^{5} & 2.409\cdot10^{5} & 2.276\cdot10^{5}
\end{bmatrix}$$

Table 5, Part II: Second-Order Volterra Kernel Coefficients for the Direct Realization (Columns 8 -14)

Table 6 contains a listing of the coefficients for the third-order Bandlimited Volterra filter. Only the unique coefficients, $h_3(l\Delta t, m\Delta t, n\Delta t)$, for l, m, n defined as in section 7.1.4 are listed ($\Delta t = 1/2000$ sec). The Romberg integration tolerance for determining the third-order coefficients was 0.05 times the final extrapolation estimate; however, as described in Section 7.1.1, the actual errors are significantly less than suggested by the tolerance value. For several coefficients computed at varying tolerances, the actual error associated with a 0.05 relative tolerance appear to be on the order of 0.0005.

Coefficient Indices l, m, n	$h_3(l\Delta t, m\Delta t, n\Delta t)$	Coefficient Indices l, m, n	$h_3(l\Delta t, m\Delta t, n\Delta t)$
0, 0, 0	5.614 E8	2, 2, 2	8.062 E8
0, 0, 1	1.007 E9	2, 2, 3	4.711 E9
0, 0, 2	4.051 E8	2, 2, 4	2.684 E9
0, 1, 1	1.700 E9	2, 2, 5	1.385 E9
0, 1, 2	5.974 E8	2, 3, 3	3.713 E9
0, 2, 2	1.364 E8	2, 3, 4	2.014 E9
1, 1, 1	7.361 E9	2, 3, 5	1.125 E9
1, 1, 2	4.966 E9	2, 4, 4	1.426 E9
1, 1, 3	2.653 E9	2, 4, 5	6.154 E8
1, 1, 4	1.467 E9	3, 3, 3	5.044 E9
1, 1, 5	8.139 E8	3, 3, 4	3.097 E9
1, 2, 2	4.736 E9	3, 3, 5	1.685 E9
1, 2, 3	2.395 E9	3, 4, 4	2.546 E9
1, 2, 4	1.144 E9	3, 4, 5	1.347 E9
1, 2, 5	7.001 E8	3, 5, 5	9.546 E8
1, 3, 3	1.607 E9	4, 4, 4	1.873 E9
1, 3, 4	9.989 E9	4, 4, 5	1.873 E9
1, 3, 5	5.104 E8	4, 5, 5	1.412 E9
1, 4, 4	8.733 E8	5, 5, 5	1.735 E9

Table 6: Third-Order Bandlimited Volterra Filter Coefficients

Filter coefficients for the third-order direct realization Volterra filter are listed in Table 7. Only the unique coefficients are listed for $\Delta t = 1/6000$ second. The Romberg integration tolerance was set at 0.05 times the final extrapolation estimate.

Coefficient Indices l, m, n	$h_3(l\Delta t, m\Delta t, n\Delta t)$	Coefficient Indices l, m, n	$h_3(l\Delta t, m\Delta t, n\Delta t)$
0, 0, 0	3.465 E8	0, 4, 4	1.025 E9
0, 0, 1	5.936 E8	1, 1, 1	1.950 E9
0, 0, 2	7.394 E8	1, 1, 2	2.574 E9
0, 0, 3	7.343 E8	1, 1, 3	2.678 E9
0, 0, 4	6.097 E8	1, 1, 4	2.305 E9
0, 1, 1	1.041 E9	1, 1, 5	1.722 E9
0, 1, 2	1.312 E9	1, 2, 2	3.503 E9
0, 1, 3	1.295 E9	1, 2, 3	3.739 E9
0, 1, 4	1.082 E9	1, 2, 4	3.293 E9
0, 1, 5	7.835 E8	1, 2, 5	2.499 E9
0, 2, 2	1.659 E9	1, 2, 6	1.783 E9
0, 2, 3	1.638 E9	1, 2, 7	1.378 E9
0, 2, 4	1.352 E9	1, 3, 3	4.075 E9
0, 2, 5	9.571 E8	1, 3, 4	3.653 E9
0, 3, 3	1.629 E9	1, 3, 5	2.810 E9
0, 3, 4	1.311 E9	1, 3, 6	2.003 E9
0, 3, 5	9.009 E8	1, 4, 4	3.326 E9

Table 7, Part I: Direct 3rd-Order Volterra Coefficients

Coefficient Indices l, m, n	$h_3(l\Delta t, m\Delta t, n\Delta t)$	Coefficient Indices l, m, n	$h_3(l\Delta t, m\Delta t, n\Delta t)$
1, 4, 5	2.588 E9	2, 5, 6	3.356 E9
1, 4, 6	1.849 E9	2, 5, 7	2.519 E9
1, 5, 5	2.036 E9	2, 5, 8	2.032 E9
2, 2, 2	5.003 E9	2, 6, 6	2.656 E9
2, 2, 3	5.575 E9	2, 6, 7	2.034 E9
2, 2, 4	5.118 E9	2, 6, 8	1.650 E9
2, 2, 5	4.038 E9	2, 7, 7	1.618 E9
2, 2, 6	2.937 E9	3, 3, 3	7.824 E9
2, 2, 7	2.227 E9	3, 3, 4	7.800 E9
2, 2, 8	1.936 E9	3, 3, 5	6.644 E9
2, 3, 3	6.442 E9	3, 3, 6	5.084 E9
2, 3, 4	9.709 E9	3, 3, 7	3.815 E9
2, 3, 5	4.984 E9	3, 3, 8	3.104 E9
2, 3, 6	3.691 E9	3, 3, 9	2.774 E9
2, 3, 7	2.768 E9	3, 3,10	2.481 E9
2, 3, 8	2.331 E9	3, 4, 4	8.131 E9
2, 3, 9	3.849 E9	3, 4, 5	7.224 E9
2, 4, 4	6.011 E9	3, 4, 6	5.715 E9
2, 4, 5	5.038 E9	3, 4, 7	4.323 E9
2, 4, 6	3.805 E9	3, 4, 8	3.431 E9
2, 4, 7	2.841 E9	3, 4, 9	2.976 E9
2, 4, 8	2.330 E9	3, 4,10	2.651 E9
2, 5, 5	4.346 E9	3, 5, 5	6.685 E9

Table 7, Part II: Direct 3rd-Order Volterra Coefficients

Coefficient Indices l, m, n	$h_3(l\Delta t, m\Delta t, n\Delta t)$	Coefficient Indices l, m, n	$h_3(l\Delta t, m\Delta t, n\Delta t)$
3, 5, 6	5.473 E9	4, 5, 7	5.692 E9
3, 5, 7	4.205 E9	4, 5, 8	4.431 E9
3, 5, 8	3.294 E9	4, 5, 9	3.607 E9
3, 5, 9	2.778 E9	4, 5,10	3.100 E9
3, 5,10	2.446 E9	4, 5,11	2.667 E9
3, 6, 6	4.633 E9	4, 6, 6	6.471 E9
3, 6, 7	3.647 E9	4, 6, 7	5.314 E9
3, 6, 8	2.864 E9	4, 6, 8	4.182 E9
3, 6, 9	2.376 E9	4, 6, 9	3.352 E9
3, 6,10	2.060 E9	4, 6,10	2.813 E9
3, 7, 7	2.957 E9	4, 7, 7	4.514 E9
3, 7, 8	2.383 E9	4, 7, 8	3.641 E9
3, 8, 8	2.004 E9	4, 7, 9	2.930 E9
4, 4, 4	8.962 E9	4, 7,10	2.426 E9
4, 4, 5	8.398 E9	4, 8, 8	3.035 E9
4, 4, 6	6.949 E9	4, 8, 9	2.512 E9
4, 4, 7	5.368 E9	4, 8,10	2.101 E9
4, 4, 8	4.191 E9	4, 9, 9	2.169 E9
4, 4, 9	3.510 E9	5, 5, 5	8.664 E9
4, 4,10	3.082 E9	5, 5, 6	7.917 E9
4, 4,11	2.634 E9	5, 5, 7	6.543 E9
4, 5, 5	8.279 E9	5, 5, 8	5.145 E9
4, 5, 6	7.164 E9	5, 5, 9	4.097 E9

Table 7, Part III: Direct 3rd-Order Volterra Coefficients

Coefficient Indices l, m, n	$h_3(l\Delta t, m\Delta t, n\Delta t)$	Coefficient Indices l, m, n	$h_3(l\Delta t, m\Delta t, n\Delta t)$
5, 5,10	3.423 E9	6, 6,11	3.083 E9
5, 5,11	2.935 E9	6, 7, 7	6.750 E9
5, 6, 6	7.617 E9	6, 7, 8	5.847 E9
5, 6, 7	6.571 E9	6, 7, 9	4.742 E9
5, 6, 8	5.285 E9	6, 7,10	3.759 E9
5, 6, 9	4.179 E9	6, 7,11	3.039 E9
5, 6,10	3.404 E9	6, 8, 8	5.272 E9
5, 6,11	2.876 E9	6, 8, 9	4.407 E9
5, 7, 7	5.908 E9	6, 8,10	3.528 E9
5, 7, 8	4.904 E9	6, 8,11	2.817 E9
5, 7, 9	3.915 E9	6, 9, 9	3.806 E9
5, 7,10	3.146 E9	6, 9,10	3.125 E9
5, 8, 8	4.208 E9	6, 9,11	2.512 E9
5, 8, 9	3.445 E9	6,10,10	2.651 E9
5, 8,10	2.784 E9	7, 7, 7	6.829 E9
5, 8,11	2.277 E9	7, 7, 8	6.212 E9
5, 9, 9	2.917 E9	7, 7, 9	5.195 E9
5, 9,10	2.424 E9	7, 7,10	4.137 E9
6, 6, 6	7.781 E9	7, 7,11	3.283 E9
6, 6, 7	7.070 E9	7, 8, 8	5.927 E9
6, 6, 8	5.882 E9	7, 8, 9	5.151 E9
6, 6, 9	4.675 E9	7, 8,10	4.182 E9
6, 6,10	3.731 E9	7, 8,11	3.301 E9

Table 7, Part IV: Direct 3rd-Order Volterra Coefficients

Coefficient Indices l, m, n	$h_3(l\Delta t, m\Delta t, n\Delta t)$	Coefficient Indices l, m, n	$h_3(l\Delta t, m\Delta t, n\Delta t)$
7, 9, 9	4.653 E9	9, 9, 9	5.214 E9
7, 9,10	3.890 E9	9, 9,10	4.731 E9
7, 9,11	3.102 E9	9, 9,11	3.936 E9
7,10,10	3.359 E9	9,10,10	4.489 E9
7,10,11	2.747 E9	9,10,11	3.874 E9
7,11,11	2.323 E9	9,11,11	3.473 E9
8, 8, 8	5.977 E9	10,10,10	4.485 E9
8, 8, 9	5.442 E9	10,10,11	4.045 E9
8, 8,10	4.549 E9	10,11,11	3.815 E9
8, 8,11	3.608 E9	11,11,11	3.787 E9
8, 9, 9	5.188 E9		
8, 9,10	4.501 E9		
8, 9,11	3.638 E9		
8,10,10	4.054 E9		
8,10,11	3.375 E9		
8,11,11	2.904 E9		

Table 7, Part V: Direct 3rd-Order Volterra Coefficients

References

[1] B. Leon and D. Schaefer, "Volterra series and Picard iteration for nonlinear circuits and systems", *IEEE Trans. Circuits and Systems*, Vol. CAS-25, pp. 789-793, September 1978.

[2] K. Kim and E. Powers, "A digital method of modeling quadratically nonlinear systems with a general random input", *IEEE Trans. Acoustics Speech and Signal Processing*, ASSP-36, pp. 1758-1769, November 1988.

[3] K. Shanmugam and M. Lal, "Analysis and synthesis of a class of nonlinear systems", *IEEE Trans. Circuits and Systems*, vol. CAS-23, pp. 17-25, January 1976.

[4] V. Volterra, "Sopra le funzioni che dipendono de altre funzioni", *Rend. R. Accademia dei Lincei 2°Sem,* 1887.

[5] N. Wiener, "Response of a nonlinear device to noise", M.I.T. Radiation Lab., Report No. 129, April, 1942.

[6] N. Wiener, *Nonlinear Problems in Random Theory*, The Technology Press, M.I.T. and John Wiley & Sons, New York, 1958.

[7] M. Schetzen, *The Volterra and Wiener Theories of Nonlinear Systems*, John Wiley & Sons, New York, 1980.

[8] S. Boyd, "Volterra series: engineering fundamentals", Ph.D. dissertation, University of California, Berkeley, 1985.

[9] S. Boyd, Y. Tang, and L. Chua, "Measuring Volterra kernels", *IEEE Trans. Circuits and Systems*, CAS-30, pp. 571-577, August 1983.

[10] Y. L. Kuo, "Frequency-domain analysis of weakly nonlinear networks", *IEEE Circuits and Systems Magazine*, pp 2-8, August 1977.

[11] J. Bussgang, L. Ehrman, and J. Graham, "Analysis of nonlinear systems with multiple inputs", *Proc. IEEE*, Vol 62, pp. 1088-1119, August 1974.

[12] D. Weiner and J. Spina, *Sinusoidal Analysis and Modeling of Weakly Nonlinear Circuits*, Van Nostrand Reinhold, New York, 1980.

[13] G. Lambrianou and C. Aitchison, "Optimization of third-order intermodulation product and output power from an X-band MESFET amplifier using Volterra series analysis", *IEEE Trans. Microwave Theory and Techniques*, MTT-33, pp. 1395-1403, December 1985.

[14] T. Koh and E. Powers, "Second-order Volterra filtering and its application to nonlinear system identification", *IEEE Trans. Acoustics Speech and Signal Processing*, ASSP-33, pp. 1445-1455, December 1985.

[15] D. Hummels and R. Gitchell, "Equivalent low-pass representation for bandpass Volterra systems", *IEEE Trans. Communications*, COM-28, pp 140-142, January 1980.

[16] M. Maqusi, "Performance of baseband digital data transmission in nonlinear channels with memory", *IEEE Trans. Communications*, COM-33, pp 715-719, July 1985.

[17] M. Steer and P. Khan, "An algebraic formula for the output of a system with large-signal, multifrequency excitation", *Proc. IEEE*, pp 177-179, January 1983.

[18] E. Bedrosian and S. Rice, "The output properties of Volterra systems (nonlinear systems with memory) driven by harmonic and Gaussian inputs", *Proc. IEEE*, vol. 59, pp. 1688-1707, December 1971.

[19] M. Jeruchim, "Techniques for estimating the bit error rate in the simulation of digital communication systems", *IEEE J. Selected Areas in Comm.*, Vol. SAC-2, pp. 153-170, January 1984.

[20] A. Jerri, "The Shannon sampling theorem - its various extensions and applications: a tutorial review", *Proc. IEEE*, Vol. 65, pp. 1565-1696, November 1977.

[21] A. Papoulis, "Error analysis in sampling theory", *Proc. IEEE*, Vol. 54, pp. 947-955, July 1966.

[22] J. Brown, Jr., "On mean-square aliasing error in the cardinal series expansion of random processes", *IEEE Trans. Information Theory*, vol. IT-24, pp. 254-256, Mar. 1978.

[23] P. Weiss, "An estimate of the error arising from misapplication of the sampling theorem", *Amer. Math. Soc. Notices*, No. 10.351, (abstract No. 601-54), 1963.

[24] B. Tsybakov and V. Iakovlev, "On the accuracy of restoring a function with a finite number of terms of Kotel'nikov series", *Radio Eng. Electron. (Phys.)*, Vol. 4, pp. 274-275, March 1959.

[25] H. Helms and J. Thomas, "Truncation error of sampling theorem expansion", Proc. IRE, Vol. 50, pp. 179-184, February 1962.

[26] H. Urkowitz, *Signal Theory and Random Processes*, Artech House, Dedham, MA, 1983.

[27] A. Papoulis, *Signal Analysis*, McGraw-Hill, New York, 1977.

[28] Z. Kowalczuk, "On discretization of continuous-time state-space models: a stable-normal approach", *IEEE Trans. Circuits and Systems*, Vol. CAS-38, pp. 1460-1477, December 1991.

[29] A. Oppenheim and R. Schafer, *Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1975.

[30] A. Schneider, J. Kaneshige, and F. Groutage, "Higher order s-to-z mapping functions and their application in digitizing continuous-time filters", *Proc. IEEE*, Vol. 79, pp. 1661-1674, November 1991.

[31] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*, Springer-Verlag, New York, 1980.

[32] *Mathcad 3.1 Users Guide*, MathSoft Inc., Cambridge, MA, 1992.

[33] L. Rabiner and B. Gold, *Theory and Application of Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1975.

[34] M. Greenberg, *Advanced Engineering Mathematics*, Prentice-Hall, Englewood Cliffs, NJ, 1988.

[35] L. Johnson and R. Riess, *Numerical Analysis*, Addison-Wesley, Reading, MA, 1977.

Vita

Carl David Garthwaite was born March 20, 1954 in Newark, New Jersey to David and Arlene Garthwaite. He received the Bachelor of Science Degree in Electrical Engineering with Honors from Lehigh University, Bethlehem, Pennsylvania, in 1976 and the Master of Science in Electrical Engineering from Drexel University, Philadelphia, Pennsylvania, in 1985.

From 1976 to 1980, Carl Garthwaite served in the United States Army, and from 1980 until 1982, he was employed by the Burroughs Corporation. In 1982, he joined the General Electric Company where he is a senior systems engineer working in the area of high data rate communications. He is a member of the Institute of Electrical and Electronics Engineers.